Solaris のシステム管理 (資源管理とネットワークサービス)

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

Part No: 816–6237–10
2002 年 9 月

原典: *System Administration Guide: Resource Management and Network Services*

Part No: 816-4882-10

Revision A

020808@2851

目次

表目次

図目次

例目次

はじめに

本書『*Solaris* のシステム管理 *(*資源管理とネットワークサービス*)*』は、Solaris™ システム管理に関する重要な情報を提供する、システム管理マニュアルセットの 1 冊です。このマニュアルでは、 SunOS™ 5.9 オペレーティングシステムがインストール済みであることを前提としています。 また、使用する予定のネットワークソフトウェアも設定しておく必要があります。 Solaris 9 製品ファミリには、SunOS 5.9 オペレーティングシステムのほか、Solaris 共通デスクトップ環境 (CDE) をはじめとする多くの機能が含まれています。

注 – Solaris オペレーティング環境は、SPARC™ と IA の 2 種類のハードウェア (プラットフォーム) 上で動作します。また、Solaris オペレーティング環境は、64 ビットと 32 ビットの両方のアドレス空間で動作します。このマニュアルで説明する情報は、特に明記しないかぎり、両方のプラットフォームおよび両方のアドレス空間に 適用されます。

対象読者

このマニュアルは、Solaris 9 リリースが稼働しているシステムの管理者を対象としています。このマニュアルを活用するには、1、2 年程度の UNIX® システムの管理経験が必要です。UNIX システム管理のトレーニングコースに参加することも知識の習得に役立ちます。

Solaris システム管理マニュアルセットの構成

システム管理マニュアルセットに含まれる各マニュアルとその内容は、次のとおりです。

マニュアル名	内容
『*Solaris* のシステム管理 *(*基本編*)*』	ユーザーアカウントとグループの管理、サーバーとクライアントの管理、システムのシャットダウンとブート、取り外し可能な媒体の管理、ソフトウェア管理 (パッケージとパッチ)、ディスクとデバイスの管理、ファイルシステムの管理、およびデータのバックアップと復元
『*Solaris* のシステム管理 *(*上級編*)*』	印刷サービスの管理、端末とモデムの設定、システム資源の管理 (ディスク割り当て、アカウンティング、および crontab ファイルの管理)、システムプロセスの管理、 および Solaris ソフトウェアの障害追跡
『*Solaris* のシステム管理 *(IP* サービス*)*』	TCP/IP ネットワークの管理、IPv4 から IPv6 への移行と IPv6 の管理、DHCP の管理、IP セキュリティの管理、モバイル IP の管理、および IP ネットワークマルチパスの管理
『*Solaris* のシステム管理 *(*ネーミングとディレクトリサービス *: DNS, NIS, LDAP* 編*)*』	DNS、NIS、および LDAP のネーミングとディレクトリサービスの管理
『*Solaris* のシステム管理 *(*ネーミングとディレクトリサービス *: FNS, NIS+* 編*)*』	FNS および NIS+ のネーミングとディレクトリサービスの管理
『*Solaris* のシステム管理 *(*資源管理とネットワークサービス*)*』	システム資源、リモートファイルシステム、メールサービス、SLP、および PPP の管理
『*Solaris* のシステム管理 *(*セキュリティサービス*)*』	監査、PAM、RBAC、および SEAM の管理

関連マニュアル

このマニュアルでは、以下の関連マニュアルと関連書籍を参照します。

- 『*Solaris* のシステム管理 *(*上級編*)*』
- 『*Solaris* のシステム管理 *(*基本編*)*』
- 『*Solaris* のシステム管理 *(IP* サービス*)*』

- 『*Solaris* のシステム管理 *(*ネーミングとディレクトリサービス *: DNS, NIS, LDAP* 編*)*』
- 『*Solaris* のシステム管理 *(*ネーミングとディレクトリサービス *: FNS, NIS+* 編*)*』
- 『*Solaris* のシステム管理 *(*資源管理とネットワークサービス*)*』
- 『*Solaris* のシステム管理 *(*セキュリティサービス*)*』
- 『*UNIX Communications*』Anderson、Bart、Bryan Costales、Harry Henderson 著、Howard W. Sams & Company 発行、1987
- 『*sendmail, Second Edition* 』Costales、Bryan 著、O'Reilly & Associates, Inc. 発行、1997
- 『*A Directory of Electronic Mail Addressing and Networks*』Frey、Donnalyn、Rick Adams 著、O'Reilly & Associates, Inc. 発行、1993
- 『*The Whole Internet User's Guide and Catalog*』Krol、Ed 著、O'Reilly & Associates, Inc. 発行、1993
- 『*Managing UUCP and Usenet*』O' Reilly、Tim、Grace Todino 著、O'Reilly & Associates, Inc. 発行、1992

関連情報

PPPoE の使用許諾権については、以下の各ファイルを参照してください。

/var/sadm/pkg/SUNWpppd/install/copyright

/var/sadm/pkg/SUNWpppdu/install/copyright

/var/sadm/pkg/SUNWpppg/install/copyright

Sun のオンラインマニュアル

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

表記上の規則

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

表 **P–1** 表記上の規則

字体または記号	意味	例
AaBbCc123	コマンド名、ファイル名、ディレクトリ名、画面上のコンピュータ出力、コード例を示します。	.login ファイルを編集します。 ls -a を使用してすべてのファイルを表示します。 system%
AaBbCc123	ユーザーが入力する文字を、画面上のコンピュータ出力と区別して示します。	system% **su** password:
AaBbCc123	変数を示します。実際に使用する特定の名前または値で置き換えます。	ファイルを削除するには、rm *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、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超える場合に、継続を示します。	sun% **grep '^#define ** **XV_VERSION_STRING'**

コード例は次のように表示されます。

- C シェル

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

- C シェルのスーパーユーザー

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

- Bourne シェルおよび Korn シェル

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

- Bourne シェルおよび Korn シェルのスーパーユーザー

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

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

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

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

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

一般規則

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

第 1 章

システム資源の管理とネットワークサービス (概要)

Solaris 9 リリースのトピック

このマニュアルでは、次のサービスとユーティリティについて説明します。

第 5 章	システム資源の割り当て、監視、および制御を容易にするシステム資源管理機能について説明します。
第 14 章	リモートホストからファイルシステムへのアクセスを可能にする NFS プロトコルについて説明します。
第 18 章	動的サービス発見プロトコルである SLP について説明します。
第 24 章	ネットワークに応じたルーティングにより、だれにでもメッセージを送信できるメールサービス機能について説明します。
第 29 章	リモートホスト間にポイントツーポイント接続を提供する PPP プロトコルについて説明します。
第 38 章	ホスト間のファイル交換を可能にする UUCP について説明します。
第 42 章	リモートシステムからファイルにアクセスするために使用するコマンド、`ftp`、`rlogin`、および `rcp` について説明します。
第 2 章	Web ページのキャッシングにより、Web サーバーのパフォーマンスを向上する NCA について説明します。
42 ページの「Perl 5」	システム管理作業を簡略化するためのスクリプトを生成するのに使用する Perl (Practical Extraction and Report Language) について説明します。

Perl 5

この Solaris リリースには、Perl (Practical Extraction and Report Language) 5.6.1 が付属しています。この強力な汎用プログラミング言語は、一般にフリーソフトウェアとして入手可能なツールです。Perl はプロセス、ファイル、およびテキスト処理機能に優れ、複雑なシステム管理作業を行う際の標準的な開発ツールとして広く使用されています。

Perl 5 には、動的にロード可能なモジュールフレームワークが含まれています。このモジュールフレームワークを使用すると、特定の作業に新しい機能を追加することができます。多くのモジュールが、Comprehensive Perl Archive Network (CPAN) のホームページ (http://www.cpan.org) から自由に入手できます。

Perl ドキュメントへのアクセス

この Solaris リリースには、Perl に関する情報ソースも含まれています。以下に同じ情報へアクセスするための 2 通りの方法を示します。

MANPATH 環境変数に `/usr/perl5/man` を設定すると、マニュアルページにアクセスできます。次の例は Perl の概要を表示します。

```
% setenv MANPATH "${MANPATH} : /usr/perl5/man"
% man perl
```

追加ドキュメントには、`perldoc` ユーティリティを使用してアクセスします。次の例も同じ概要を表示します。

```
% /usr/perl5/bin/perldoc perl
```

`perl` 概要ページには、このリリースに含まれているすべてのドキュメントの一覧が示されています。

Perl の互換性について

一般に、バージョン 5.6.1 の Perl は以前のバージョンと互換性があるため、スクリプトの再作成や再コンパイルは必要ありません。 ただし、XSUB ベースのモジュール (.xs) はすべて、再コンパイルおよび再インストールする必要があります。

Solaris 9 リリースでは、`/usr/perl5/5.00503/bin/perl` と入力することにより、古いバージョンの Perl にアクセスできます。古いバージョンは将来のリリースではサポートされない可能性があるため、このバージョンは新しいモジュールを再構築するまでの一時的な代用ツールとして使用してください。

Solaris 版 Perl の変更点

Solaris 版 Perl は 64 ビット整数、malloc システムコール、および大規模ファイルをサポートするようにコンパイルされています。また、必要なパッチも適用済みです。すべての構成情報の一覧については、次のコマンドの出力を参照してください。

```
% /usr/perl5/bin/perlbug -dv
---
Flags:
    category=
    severity=
---
Site configuration information for perl v5.6.1:
 .
 .
```

`perl -V` と入力すると、構成の要約リストを生成できます。

第 2 章

Web キャッシュサーバーの管理

この章では、Solaris NCA (ネットワークキャッシュとアクセラレータ) の概要について説明します。また、NCA を使用するための手順と、NCA に関する参考資料も示します。

NCA (ネットワークキャッシュとアクセラレータ) (概要)

Solaris NCA (ネットワークキャッシュとアクセラレータ) は、HTTP 要求時にアクセスされる Web ページのカーネル内キャッシュを保持することにより、Web サーバーのパフォーマンスを向上します。このカーネル内キャッシュはシステムメモリーを使用するため、通常は Web サーバーによって処理される HTTP 要求のパフォーマンスを大幅に向上します。HTTP 要求時に Web ページがシステムメモリー内に保持されているため、カーネルと Web サーバー間のオーバーヘッドが減少し、Web サーバーのパフォーマンスが向上します。NCA にはソケットインタフェースが用意されており、どのような Web サーバーでも最小限の変更で NCA と通信できます。

要求されたページがカーネル内キャッシュから取得された場合 (キャッシュヒット時) は、パフォーマンスが飛躍的にに向上します。要求されたページがキャッシュ内になく、Web サーバーから取得する必要がある場合 (キャッシュミス時) でも、パフォーマンスは大幅に改善されます。

NCA は、専用の Web サーバー上で実行するようにします。NCA が動作するサーバー上で他の大きいプロセスを実行すると、問題が起きることがあります。

NCA はすべてのキャッシュヒットを記録するロギング機能を提供します。ログはパフォーマンスを向上させるためにバイナリ形式で格納されます。`ncab2clf` コマンドを使用すると、バイナリ形式のログを共通ログ形式 (CLF) に変換できます。

Solaris 9 リリースには、次のような機能強化が実施されています。

- ソケットインタフェースの提供
- AF_NCA サポートを可能にするベクトル化 sendfile システムコールの提供。詳細は `sendfilev(3EXT)` のマニュアルページを参照する
- `ncab2clf` コマンド用の 2 つの新しいオプション、具体的には、選択された日付以前のレコードをスキップするための `-s` オプションと、指定された数のレコードを処理するための `-n` オプションの追加
- `ncalogd.conf` ファイル内の `logd_path_name` を用いて raw デバイス、ファイル、または両者の組み合わせを指定可能

Web キャッシュサーバーの管理 (作業マップ)

次の表に、NCA を使用するために必要な手順を示します。

作業	説明	参照先
NCA の利用を計画する	NCA を使用するための要件をリストする。NCA を構成する前にすべての要件を満たしているか再検討する	47 ページの「NCA を使用するためのシステム要件」
NCA を有効にする	Web サーバー上の Web ページのカーネル内キャッシュを有効にするための手順を実行する	47 ページの「Web ページのキャッシングを有効にする方法」
NCA を無効にする	Web サーバー上の Web ページのカーネル内キャッシュを無効にするための手順を実行する	49 ページの「Web ページのキャッシングを無効にする方法」
NCA ロギングを管理する	NCA ロギング処理を有効または無効にするための手順を実行する	50 ページの「NCA ロギングを有効または無効にする方法」
NCA ソケットライブラリをロードする	AF_NCA ソケットがサポートされていない場合に NCA を使用するための手順を実行する	50 ページの「NCA ソケットユーティリティライブラリのロード方法」

Web ページのキャッシュ管理 (手順)

以下に、NCA を使用するために必要なシステム要件と、サービスを有効または無効にするための手順を示します。

NCA を使用するためのシステム要件

NCA をサポートするには、システムは次の要件を満たす必要があります。

- 256M バイトの RAM がインストールされている
- Solaris 9 リリース、または Solaris 8 アップグレードリリースのいずれかがインストールされている
- Apache がサポートされている。Solaris 9 と Solaris 8 アップグレードリリースでは、Apache がサポートされている

この製品は、専用の Web サーバー上で実行するようにします。NCA を実行しているサーバー上で別の大きいプロセスを実行すると、問題が生じることがあります。

▼ Web ページのキャッシングを有効にする方法

1. スーパーユーザーになります。
2. インタフェースを登録します。

   `/etc/nca/nca.if` ファイルに各物理インタフェースの名前を指定します。詳細は、`nca.if(4)` のマニュアルページを参照してください。

   ```
   # cat /etc/nca/nca.if
   hme0
   hme1
   ```

   インタフェースごとに、対応する `hostname.`*interface-name* ファイルが必要です。また、`/etc/hosts` ファイル内に `hostname.`*interface-name* の内容と一致するエントリが必要です。 すべてのインタフェースで NCA 機能を使用可能にするには、`nca.if` ファイル内でアスタリスク (`*`) を指定します。
3. `ncakmod` カーネルモジュールを有効にします。

   `/etc/nca/ncakmod.conf` 内の `status` エントリを `enabled` に変更します。

   ```
   # cat /etc/nca/ncakmod.conf
   #
   # NCA Kernel Module Configuration File
   #
   status=enabled
   httpd_door_path=/var/run/nca_httpd_1.door
   ```

```
nca_active=disabled
```

詳細は、ncakmod.conf(4) のマニュアルページを参照してください。

4. **NCA** ロギングを有効にします。

   /etc/nca/ncalogd.conf 内の status エントリを enabled に変更します。

```
# cat /etc/nca/ncalogd.conf
#
# NCA Logging Configuration File
#
status=enabled
logd_path_name="/var/nca/log"
logd_file_size=1000000
```

   logd_path_name エントリに示されているパスを変更すると、ログファイルの格納場所を変更できます。ログファイルには raw デバイスとファイルのどちらでも指定できます。NCA ログファイルのパス例については、次項の例を参照してください。構成ファイルの詳細は、ncalogd.conf(4) のマニュアルページを参照してください。

5. **IA** のみ**:** 仮想メモリーサイズを増やします。

   eeprom コマンドを使用して、システムの kernelbase を設定します。

```
# eeprom kernelbase=0x90000000
# eeprom kernelbase
kernelbase=0x90000000
```

   2 行目の eeprom コマンドを実行すると、パラメータが設定済みかどうかを確認できます。

注 – kernelbase を設定すると、ユーザープロセスが使用できる仮想メモリー領域が 3G バイト未満に減少します。 このため、システムは ABI に準拠しなくなります。 システムをブートすると、そのことを警告するメッセージがコンソールに表示されます。ほとんどのプログラムは、実際には 3G バイトの仮想アドレス空間を必要としません。3G バイト以上の仮想アドレス空間を必要とするプログラムは、NCA を無効に設定したシステム上で実行する必要があります。

---

6. サーバーを再起動します。

例 - NCA ログファイルとして raw デバイスを使用する

ncalogd.conf ファイル内の logd_path_name 文字列で、NCA ログファイルの格納先として raw デバイスを指定できます。raw デバイスを使用する利点としては、アクセス時のオーバーヘッドが小さいため、サービスを高速に実行できることが挙げられます。

NCA サービスはファイル内に記述されているすべての raw デバイスに対して、対応するファイルシステムがないことを確認します。このテストは、アクティブなファイルシステムを誤って上書きしてしまわないように実行されます。

このテストでファイルシステムが検出されないようにするには、以下のコマンドを実行して、ファイルシステムとして構成されているすべてのディスクパーティション上のファイルシステム部分を破棄します。この例では、/dev/rdsk/c0t0d0s7 が古いファイルシステムを持つ raw デバイスです。

```
# dd if=/dev/zero of=/dev/rdsk/c0t0d0s7 bs=1024 count=1
```

dd コマンドを実行すると、ncalogd.conf ファイルに raw デバイスを追加できるようになります。

```
# cat /etc/nca/ncalogd.conf
#
# NCA Logging Configuration File
#
status=enabled
logd_path_name="/dev/rdsk/c0t0d0s7"
logd_file_size=1000000
```

例 - NCA ロギング用に複数のファイルを使用する

ncalogd.conf ファイル内の logd_path_name 文字列で、NCA ログファイルの格納先として複数のファイルを指定できます。 最初のファイルが満杯になると、2 番目のファイルが使用されます。以下の例では、最初に /var/nca/log ファイルを書き込みに使用し、次に raw パーティションを使用する方法を示します。

```
# cat /etc/nca/ncalogd.conf
#
# NCA Logging Configuration File
#
status=enabled
logd_path_name="/var/nca/log /dev/rdsk/c0t0d0s7"
logd_file_size=1000000
```

▼ Web ページのキャッシングを無効にする方法

1. スーパーユーザーになります。
2. ncakmod カーネルモジュールを無効にします。

   /etc/nca/ncakmod.conf 内の status エントリを disabled に変更します。

   ```
   # cat /etc/nca/ncakmod.conf
   # NCA Kernel Module Configuration File
   #
   status=disabled
   httpd_door_path=/var/run/nca_httpd_1.door
   nca_active=disabled
   ```

   詳細は、ncakmod.conf(4) のマニュアルページを参照してください。
3. **NCA** ロギングを無効にします。

/etc/nca/ncalogd.conf 内の status エントリを disabled に変更します。

```
# cat /etc/nca/ncalogd.conf
#
# NCA Logging Configuration File
#
status=disabled
logd_path_name="/var/nca/log"
logd_file_size=1000000
```

詳細は、ncalogd.conf(4) のマニュアルページを参照してください。

4. サーバーを再起動します。

▼ NCA ロギングを有効または無効にする方法

NCA が有効になっていると必要に応じて NCA のログ処理のオン/オフを切り換えることができます。詳細は 47 ページの「Web ページのキャッシングを有効にする方法」 を参照してください。

1. スーパーユーザーになります。
2. **NCA** ロギングのオン**/**オフを切り換えます。
   ロギングを恒久的に無効にする場合は、/etc/nca/ncalogd.conf 内の status を disabled に変更し、システムをリブートする必要があります。詳細は、ncalogd.conf(4) のマニュアルページを参照してください。

   a. ロギングを停止します。

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

   b. ロギングを開始します。

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

▼ NCA ソケットユーティリティライブラリのロード方法

この手順は、AF_NCA ソケットを直接にサポートしていない Web サーバーに対してのみ使用します。

Web サーバーの起動スクリプトに、ライブラリをプリロードするための 1 行を追加します。次のような行を追加します。

```
LD_PRELOAD=/usr/lib/ncad_addr.so /usr/bin/httpd
```

Web ページのキャッシング (リファレンス)

この節では、NCA を使用するために必要なファイルとコンポーネントについて説明します。 また、NCA が Web サーバーと通信する方法についても説明します。

NCA ファイル

NCA 機能をサポートするにはいくつかのファイルが必要です。ほとんどのファイルは ASCII 形式ですが、バイナリ形式のファイルもあります。次の表に必要なファイルの一覧を示します。

表 **2–1** NCA ファイル

ファイル名	機能
`/etc/hostname.*`	サーバー上で構成されているすべての物理インタフェースについてホスト名が記述されているファイル
`/etc/hosts`	サーバーに対応付けられるすべてのホスト名が記述されているファイル。 NCA が機能するには、このファイルの各エントリが、対応する `/etc/hostname.*` ファイル内のエントリと一致していなければならない
`/etc/init.d/ncakmod`	NCA サーバーを起動するスクリプト。このスクリプトは、サーバーのブート時に実行される
`/etc/init.d/ncalogd`	NCA ロギングを開始するスクリプト。 このスクリプトは、サーバーのブート時に実行される
`/etc/nca/nca.if`	NCA が実行されるすべてのインタフェースが記述されているファイル。詳細は `nca.if(4)` のマニュアルページを参照
`/etc/nca/ncakmod.conf`	NCA 用のすべての構成パラメータが記述されているファイル。 詳細は、`ncakmod.conf(4)` のマニュアルページを参照
`/etc/nca/ncalogd.conf`	NCA ロギング用のすべての構成パラメータが記述されているファイル。詳細は、`ncalogd.conf(4)` のマニュアルページを参照
`/usr/bin/ncab2clf`	ログファイル内のデータを共通ログ形式に変換するために使用されるコマンド。詳細は `ncab2clf(1)` のマニュアルページを参照

表 **2–1** NCA ファイル　(続き)

ファイル名	機能
/usr/lib/net/ncaconfd	ブート時に複数のインタフェース上で NCA が実行するように設定するために使用されるコマンド。詳細は ncaconfd(1M) のマニュアルページを参照
/usr/lib/nca_addr.so	AF_INET ソケットの代わりに AF_NCA ソケットを使用するライブラリ。このライブラリは AF_INET ソケットを使用する Web サーバー上で使用する。詳細は ncad_addr(4) のマニュアルページを参照
/var/nca/log	ログファイルのデータを保持するファイル。バイナリ形式のファイルなので編集できない

NCA アーキテクチャ

NCA が機能するためには、次のコンポーネントが必要です。

- カーネルモジュール: ncakmod
- Web サーバー: httpd

カーネルモジュール ncakmod は、Web ページのキャッシュをシステムメモリー内に保持します。このモジュールは、ソケットインタフェース (ファミリータイプは PF_NCA) を介して Web サーバー httpd と通信します。

また、カーネルモジュールは、すべての HTTP キャッシュヒットを記録するログ機能も備えています。NCA ロギングは、HTTP データをバイナリ形式でディスクに書き込みます。NCA には、バイナリログファイルを共通ログ形式 (CLF) に変換するユーティリティが用意されています。

次の図に、通常のデータフローと、NCA が有効になっている場合のデータフローを示します。

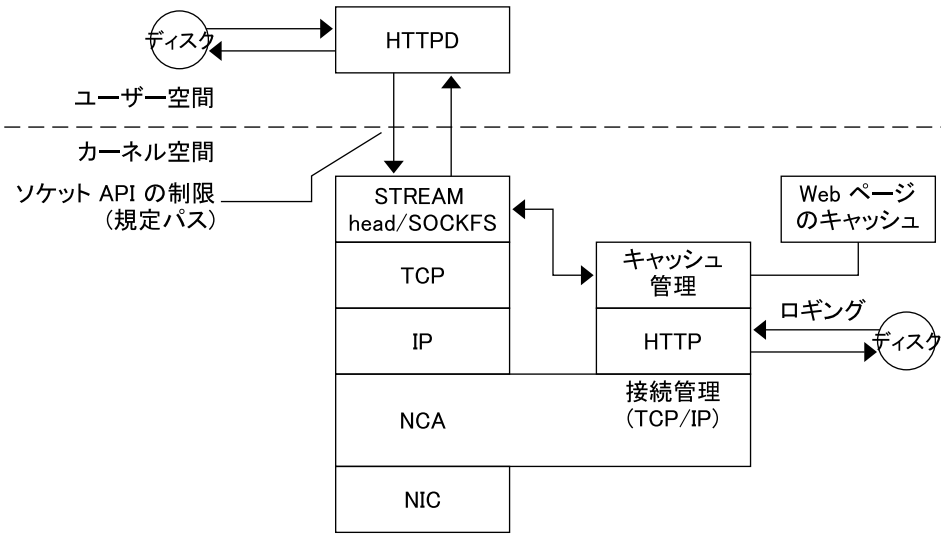


図 **2–1** NCA サービスのデータフロー

NCA から httpd への要求フロー

以下に、クライアントと Web サーバー間の要求フローを示します。

1. クライアントから Web サーバーに対して HTTP 要求が発行されます。
2. ページがキャッシュ内にある場合は、カーネル内キャッシュの Web ページが返されます。
3. ページがキャッシュ内にない場合は、Web サーバーに要求が送信され、ページが取得または更新されます。
4. ページがキャッシュされているかどうか、またクライアントに返されるかどうかは、HTTP 応答で使用される HTTP プロトコルのセマンティクスによって異なります。 HTTP 要求に「Pragma:No-cache」ヘッダーが含まれている場合、ページはキャッシュされません。

ライブラリ置き換えによる door サーバーデーモンのサポート

多くの Web サーバーは AF_INET ソケットを使用します。デフォルトでは、NCA は AF_NCA ソケットを使用します。この状況に対応するために、置き換え用のライブラリが用意されています。新しいライブラリは標準ソケットライブラリ `libsocket.so` の前にロードされます。`bind()` ライブラリ呼び出しは新しいライブラリ ncad_addr.so の用意するものが呼び出されます。

`/etc/nca/ncakmod.conf` ファイル内で状態が有効に設定されていれば、Solaris 9 リリースに付属している Apache は、このライブラリを呼び出すように設定されています。IWS または Netscape サーバーで新しいライブラリを使用する場合は、50 ページの「NCA ソケットユーティリティライブラリのロード方法」 を参照してください。

第 3 章

システムの時刻関連サービス

多くのデータベースと認証サービスでは、ネットワーク内でシステムクロックを同期させる必要があります。この章の内容は次のとおりです。

時刻の同期 (概要)

Solaris 2.6 以降、Solaris ソフトウェアには Delaware 大学の NTP (Network Time Protocol) 公開ドメインソフトウェアが添付されています。`xntpd` デーモンは、UNIX システムの時刻をインターネット標準時刻サーバーの時刻と合うように調整し、保守します。`xntpd` デーモンは、RFC 1305 に規定されている NTP バージョン 3 標準に完全に準拠して実装されています。

`xntpd` デーモンは、システムの起動時に `/etc/inet/ntp.conf` ファイルを読み込みます。構成オプションの詳細は、`xntpd(1M)` のマニュアルページを参照してください。

ネットワーク内で NTP を使用するときには、次のことを考慮してください。

- `xntpd` デーモンは最小限のシステム資源しか使用しない
- NTP クライアントは起動時に、自動的に NTP サーバーと同期を取る。クライアントは同期の取れていない状態になった場合、タイムサーバーと通信したときに再同期を取る

`cron` を使用して `rdate` コマンドを実行することにより、時刻の同期を取ることもできます。

NTP の管理 (作業)

NTP サービスを設定および使用するための手順を示します。

▼ NTP サーバーを設定する方法

1. スーパーユーザーになります。
2. ntp.conf ファイルを作成します。

   xntpd デーモンを正しく実行するには、最初に ntp.conf ファイルを作成する必要があります。 ntp.server ファイルをテンプレートとして使用できます。

   ```
   # cd /etc/inet
   # cp ntp.server ntp.conf
   ```

3. xntpd デーモンを起動します。

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

▼ NTP クライアントを設定する方法

1. スーパーユーザーになります。
2. ntp.conf ファイルを作成します。

   xntpd デーモンを有効にするには、最初に ntp.conf ファイルを作成する必要があります。

   ```
   # cd /etc/inet
   # cp ntp.client ntp.conf
   ```

3. xntpd デーモンを起動します。

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

他の時刻関連コマンドの使用 (作業)

▼ 他のシステムの日時と同期させる方法

1. スーパーユーザーになります。

2. rdate コマンドを使用して、日付と時刻を設定し直し、他のシステムとの同期を取ります。

```
# rdate another-system
```

another-system　　他のシステム名前

3. date コマンドを使用して、システムの日時が正しく設定し直されていることを確認してください

出力は、指定したシステムと同じ日付と時刻を示します。

例 — 他のシステムの日時と同期させる方法

次の例は、rdate を使用してシステムの日時を別のシステムの日時と同期させる方法を示します。次の例は、時刻が数時間遅れているシステム earth の日付と時刻をサーバー starbug の日付と時刻に一致させます。

```
earth# date
Tue Jun  5 11:08:27 MDT 2001
earth# rdate starbug
Tue Jun  5 14:06:37 2001
earth# date
Tue Jun  5 14:06:40 MDT 2001
```

NTP (リファレンス)

NTP サービスを実行するには、次のファイルが必要です

表 **3–1** NTP ファイル

ファイル名	機能
/etc/inet/ntp.conf	NTP 用のすべての構成オプションが記述されているファイル
/etc/inet/ntp.client	NTP クライアント用のサンプル構成ファイル
/etc/inet/ntp.server	NTP サーバー用のサンプル構成ファイル
/etc/inet/ntp.drift	NTP サーバー上で初期周波数オフセットを設定するファイル
/etc/inet/ntp.keys	NTP サーバー用のサンプル構成ファイル
/etc/init.d/xntpd	ホストの起動時に実行される NTP 起動スクリプト

表 **3–1** NTP ファイル　(続き)

ファイル名	機能
/usr/lib/inet/xntpd	NTP デーモン。詳細は xntpd(1M) のマニュアルページを参照
/usr/sbin/ntpdate	NTP に基づいてローカルな日付と時刻を設定するユーティリティ。詳細は ntpdate(1M) のマニュアルページを参照
/usr/sbin/ntpq	NTP 照会プログラム。詳細は ntpq(1M) のマニュアルページを参照
/usr/sbin/ntptrace	マスターの NTP サーバーまで NTP ホストを追跡するプログラム。詳細は ntptrace(1M) のマニュアルページを参照
/usr/sbin/xntpdc	xntp デーモン用の NTP 照会プログラム。詳細は xntpdc (1M) のマニュアルページを参照
/var/ntp/ntpstats	NTP の統計情報を保持するディレクトリ

第 4 章

Solaris 9 リソースマネージャ (トピック)

この章と以下に示す章では、Solaris オペレーティング環境における資源管理について説明します。

第 5 章	資源管理の概要について説明し、システムに備わっている機能の利用方法について検討します。
第 6 章	プロジェクトおよびタスク機能について解説し、作業負荷にラベル付けを行い、区別する方法について説明します。
第 7 章	容量計画や課金についての詳細な資源消費統計情報を取得するのに使用する拡張アカウンティング機能について説明します。
第 8 章	システム上で実行されるアプリケーションが使用する資源量に制限を設けるための資源制御機能について説明します。
第 9 章	シェアを使用して、システム上で実行されるプロセスに割り当てる CPU 時間を指定するフェアシェアスケジューラについて説明します。
第 10 章	システム資源を分割し、システム上で実行される特定の作業負荷に対して常に一定量の資源の使用を保証する資源プールについて説明します。
第 11 章	仮想的なサーバー統合プロジェクトについて説明します。
第 12 章	Solaris 管理コンソールで利用できる資源管理機能について説明します。

第 5 章

Solaris 9 リソースマネージャの紹介

資源管理機能を利用すると、アプリケーションが利用可能なシステム資源をどのように使用するかを制御できます。次のような制御が可能になります。

- プロセッサ時間などのコンピュータ資源を割り当てる
- 割り当てた資源の使用状況を監視し、必要に応じて調整する
- 分析、課金、および容量計画のために拡張アカウンティング情報を生成する

概要

今日のコンピューティング環境では、システム上で実行されるアプリケーションによって生成されるさまざまな作業負荷に柔軟に対応できる能力が求められます。資源管理機能を使用しない場合、Solaris オペレーティング環境は、新しいアプリケーションの要求に動的に適応することによって作業負荷の要求に対応しています。このデフォルトの動作は、通常、システムのすべてのアクティビティに対して資源へのアクセスを同等に与えることを意味します。Solaris の資源管理機能を使用すると、作業負荷を個別に扱うことができるようになります。次のような制御が可能になります。

- 特定の資源へのアクセスを制限する
- 優先順位に基づいて作業負荷に資源を提供する
- 作業負荷を互いに分離する

作業負荷が相互に影響し合うことによるパフォーマンスの低下を最小限に抑える機能と、資源の使用状況や使用率を監視する機能を総称して「資源管理機能」といいます。資源管理機能は、いくつかのアルゴリズムの集合として実装されます。 これらのアルゴリズムは、アプリケーションがその実行過程で発行する一連の資源要求を処理します。

資源管理機能を使用すると、さまざまな作業負荷に対してオペレーティングシステムのデフォルトの動作を変更できます。この場合の「動作」とは、主に、アプリケーションが 1 つ以上の資源要求をシステムに発行したときに、オペレーティングシステムのアルゴリズムが行う一連の決定処理のことです。 資源管理機能は、次の目的で使用できます。

- あるアプリケーションに対して、資源の割り当てを拒否したり、他のアプリケーションに許可されているよりも大きい割り当てを与えたりする
- 特定の割り当てを個別にではなく一括して行う

次のような目的を達成したいときは、資源管理機能を使用できるようにシステムを構成します。次のような制御が可能になります。

- アプリケーションが際限なく資源を浪費するのを防止する
- 外部イベントに基づいてアプリケーションの優先順位を変更する
- 一連のアプリケーションに資源の使用を保証する一方で、システムの使用率を最大限に高める

資源管理機能をシステム構成に組み込むときは、次のような作業が事前に必要です。

- システム上で競合する作業負荷を特定する
- 競合しない作業負荷と、主要な作業負荷に影響を与えるようなパフォーマンス要件を伴った作業負荷を区別する

競合しない作業負荷と競合する作業負荷を特定したら、システムの能力が許す範囲内で、業務サービスへの影響を最小限に抑えた資源構成を構築できます。

効率のよい資源管理を実現するために、Solaris 環境には制御メカニズム、通知メカニズム、および監視メカニズムが用意されています。制御メカニズムについては、63 ページの「資源管理の制御メカニズム」で説明します。 通知および監視メカニズムについては、第 8 章で説明します。これらの機能の多くは、`proc(4)` ファイルシステム、プロセッサセット、スケジューリングクラスなどの既存メカニズムの拡張機能として提供されます。その他の機能は資源管理に固有です。これらの機能については、以降の章で説明します。

資源の分類

資源は、アプリケーションの動作を変更する目的で操作されるコンピューティングシステムのあらゆる側面を意味します。つまり、資源は、アプリケーションが暗黙的または明示的に要求する機能です。 この機能が拒否または制限された場合は、堅固に作成されて いるアプリケーションの実行速度が低下します。

資源の分類は、資源の識別とは対照的に、多くの基準に基づいて行うことができます。たとえば、資源は、暗黙的な要求か明示的な要求か、時間ベース (CPU 時間など) の要求か時間に無関係な要求 (割り当てられた CPU シェアなど) か、などを基準に行うことができます。

通常、スケジューラベースの資源管理は、アプリケーションが暗黙的に要求する資源に適用されます。たとえば、実行を継続するときは、アプリケーションは暗黙的に追加の CPU 時間を要求します。 ネットワークソケットにデータを書き込むときは、アプリケーションは暗黙的に帯域幅を要求します。 暗黙的に要求される資源の総使用量に対して制約を設けることができます。

帯域幅または CPU サービスのレベルを明示的に折衝できるように、インタフェースを追加することもできます。追加スレッドの要求のように明示的に要求される資源は、制約によって管理できます。

資源管理の制御メカニズム

Solaris オペレーティング環境では、制約、スケジューリング、パーティション分割の 3 種類の制御メカニズムを使用できます。

制約

制約を使用すると、管理者やアプリケーション開発者は、作業負荷が使用する特定の資源の消費にいくつかの制限を設定できます。 制限を設定すると、資源の消費シナリオを簡単にモデル化できます。また、制限を設定することにより、無秩序に資源を要求してシステムのパフォーマンスや可用性に悪影響を及ぼす可能性がある悪質なアプリケーションを制御できます。

制約は、アプリケーションに制限を課します。アプリケーションとシステムの関係は、アプリケーションが動作できないところまで悪化してしまう可能性があります。そのような事態を回避する方法の 1 つは、資源に関する動作が不明なアプリケーションに対する制約を徐々に強めていくことです。第 8 章で説明する資源制御機能は、制約メカニズムを提供します。新たに作成するアプリケーションであれば、資源の制約をアプリケーションが認識するようにすることもできます。ただし、すべての開発者がこの機能を使用するとは限りません。

スケジューリング

スケジューリングとは、一定の間隔で割り当てを決定することです。この決定は、予測可能なアルゴリズムに基づいて行われます。現在割り当てられている必要としないアプリケーションは、他のアプリケーションが使用できるように、その資源を解放します。スケジューリングに基づいて資源管理を行うと、資源に余裕がある構成の場合は使用率を最大限にできると同時に、資源が限界まで、あるいは過剰に使用されている場合には、割り当てを制御できます。スケジューリングのアルゴリズムにより、「制御」という用語の意味が決まります。場合によっては、スケジューリングアルゴリズムは、すべてのアプリケーションが資源にある程度アクセスできることを保証します。第 9 章で説明するフェアシェアスケジューラ (FSS) は、アプリケーションが制御された方法で CPU 資源にアクセスするように管理します。

パーティション分割

パーティション分割は、作業負荷をシステム上で使用可能な資源のサブセットに結合 (バインド) するために使用されます。資源と結合することにより、作業負荷は常に一定量の資源を使用できることが保証されます。第 10 章で説明する資源プール機能は、マシンの特定のサブセットに結合する作業負荷を制限します。パーティション分割を使用する構成では、システム全体が過剰使用されるのを防ぐことができます。ただし、この方法では、高い使用率の達成は難しくなります。 予約済みの資源 (プロセッサなど) に結合されている作業負荷がアイドル状態になっている場合でも、別の作業負荷がその資源を使用することはできないためです。

資源管理構成

資源管理構成の一部をネットワークのネームサービスに置くことができます。この機能により、管理者は、資源管理制約をマシンごとに排他的に適用するのではなく、複数のマシンに対して一括して適用できます。関連する作業は共通識別子を共有でき、その作業の使用状況はアカウンティングデータに基づいて表形式で表すことができます。

資源管理構成と作業負荷識別子の詳細については、第 6 章を参照してください。これらの識別子をアプリケーションの資源使用状況と結び付ける拡張アカウンティング機能については、第 7 章を参照してください。

資源管理機能の効率的な使用

資源管理機能を使用して、アプリケーションが必要な応答時間を確保できるようにします。

また、資源管理機能により、資源の使用率を向上することができます。使用状況を分類して優先付けすることにより、オフピーク時に余った資源を効率よく使用でき、処理能力を追加する必要がなくなります。また、負荷の変動が原因で資源を無駄にすることもなくなります。

サーバーを統合する場合

資源管理機能は、多くのアプリケーションを 1 台のサーバー上で統合できる環境で使用すると最も高い効果を発揮します。

多数のマシンの管理は複雑でコストがかかるため、より大規模で拡張性の高いサーバーにアプリケーションを統合することが望まれます。個々の作業負荷を別々のシステムで実行して、そのシステムの資源へのフルアクセスを与える代わりに、資源管理

ソフトウェアを使用すれば、システム内の作業負荷を分離できます。資源管理機能を使用すると、1 つの Solaris システムで複数の異なるアプリケーションを実行して制御することにより、システムの総保有コスト (TCO) を低減することができます。

インターネットサービスやアプリケーションサービスを提供する場合は、資源管理を使用すると、次のことが可能になります。

- 1 台のマシンに複数の Web サーバーを配置する。各 Web サイトの資源消費を制御し、各サイトを他のサイトで起こる可能性のある過剰使用から保護する
- 欠陥のある CGI (Common Gateway Interface) スクリプトが CPU 資源を浪費するのを防止する
- 不正な動作をするアプリケーションによって引き起こされる、仮想メモリーのリークを防止する
- 顧客のアプリケーションが、同じサイトで実行されている別の顧客のアプリケーションの影響を受けないようにする
- 同一マシン上で異なるレベルまたはクラスのサービスを提供する
- 課金目的でアカウンティング情報を取得する

大規模で多様なユーザーが利用するシステムをサポートする場合

資源管理機能は、特に、教育機関のように大規模で多様なユーザーが利用するシステムでその効果を発揮します。さまざまな作業負荷が混在している場合は、特定のプロジェクトに高い優先順位を与えるようにソフトウェアを構成できます。

たとえば、大きな証券会社のトレーダは、データベースのクエリーや計算を実行するために、一時的に高速なアクセスが必要になる場合があります。一方、他のユーザーの作業負荷は、一定しています。トレーダのプロジェクトに、作業量に応じてより高い処理能力を割り当てれば、トレーダは必要とする応答性を確保できます。

また、資源管理機能は、シン (thin) クライアントシステムをサポートするのにも適しています。これらのプラットフォームは、スマートカードのようなフレームバッファーと入力デバイスを持つステートレスなコンソールを備えています。実際の処理は共有サーバー上で行われるため、タイムシェアリング型の環境とみなすことができます。資源管理機能を使ってサーバー上のユーザーを分離してください。こうすることで、過剰な負荷を引き起こしたユーザーがハードウェア資源を占有し、システムを使用する他のユーザーに重大な影響を与えることがなくなります。

資源管理の設定 (作業マップ)

次の作業マップに、システム上で資源管理機能を設定する際に必要となる作業の概要を示します。

作業	説明	参照先
システム上の作業負荷を特定する	/etc/project データベースファイル、または NIS マップか LDAP ディレクトリサービスでプロジェクトエントリを確認する	71 ページの「project データベース」
システム上の作業負荷に優先順位を付ける	どのアプリケーションが重要かを判定する。重要な作業負荷には資源への優先的なアクセスが必要になる場合がある	サービスの目的を考慮
システム上で実際のアクティビティを監視する	パフォーマンスツールを使用して、システムで実行されている作業負荷の現在の資源消費量を表示する。その上で、特定の資源へのアクセスを制限する必要があるかどうか、あるいは作業負荷を互いに分離する必要があるかどうかを判定できる	134 ページの「システム単位の監視」、cpustat(1M)、 iostat(1M)、 mpstat(1M)、 prstat(1M)、 sar(1)、 および vmstat(1M)
システムで実行されている作業負荷を一時的に変更する	変更可能な設定値を決めるには、Solaris 環境で使用できる資源制御を参照する。タスクまたはプロセスが実行している間は、コマンド行から値を更新できる	90 ページの「使用可能な資源制御」、92 ページの「資源制御値に対応付けられたアクション」、95 ページの「動作中のシステム上の資源制御値を一時的に更新する」、rctladm(1M)、および prctl(1)

作業	説明	参照先
project データベースまたはネームサービスプロジェクトテーブル内のプロジェクトエントリごとに資源制御属性を設定する	/etc/project データベースまたはネームサービスプロジェクトテーブル内の各プロジェクトエントリには、資源制御を 1 つ以上含めることができる。 これらの資源制御は、そのプロジェクトに属するタスクとプロセスを制約する。 資源制御で指定する各しきい値に対しては、その値に達したときに行われるアクションを 1 つ以上対応付けることができる。 資源制御は、コマンド行インタフェースまたは Solaris 管理コンソールを使って設定できる。多数のシステムの構成パラメータを設定するときは、Solaris 管理コンソールを使用する	71 ページの「project データベース」、72 ページの「ローカルの project ファイルの形式」、90 ページの「使用可能な資源制御」、92 ページの「資源制御値に対応付けられたアクション」、および第 9 章
資源プール構成を作成する	資源プールは、プロセッサなどのシステム資源をパーティション分割する手段を提供し、再起動時にもそのパーティションを保持する。/etc/project データベースの各エントリに project.pool 属性を追加できる	117 ページの「プール構成の作成」
フェアシェアスケジューラ (FSS) をデフォルトのシステムスケジューラとして設定する	単一の CPU システムまたはプロセッサセット内のすべてのユーザープロセスが同じスケジューリングクラスに属すようにする	110 ページの「FSS の構成例」および dispadmin(1M)
拡張アカウンティング機能を起動し、タスクまたはプロセスベースで資源消費を監視して記録する	拡張アカウンティングデータを使って現在の資源制御を評価し、将来の作業負荷のための容量要件を計画する。システム全体の総使用状況を追跡できる。複数のシステムに渡って相互に関連しあう作業負荷について完全な使用統計を取得するために、プロジェクト名は複数のマシンで共有できる	85 ページの「プロセス、タスク、およびフローの拡張アカウンティングを起動する方法」および acctadm(1M)
(省略可能) 構成をさらに調整する必要があると判断した場合、タスクまたはプロセスが実行している間は、引き続きコマンド行から値を変更できる	既存のタスクに対しては、プロジェクトを再起動しなくても、変更を一時的に適用できる。満足のいくパフォーマンスが得られるまで値を調整する。次に、/etc/project データベースまたはネームサービスプロジェクトテーブルで現在の値を更新する	95 ページの「動作中のシステム上の資源制御値を一時的に更新する」、rctladm(1M)、および prctl(1)

作業	説明	参照先
(省略可能) 拡張アカウンティングデータを取得する	アクティブなプロセスおよびタスクの拡張アカウンティングレコードを書き込む。作成されるファイルは、計画、チャージバック、および課金のために使用できる	`wracct(1M)`

第 6 章

プロジェクトとタスク

この章では、Solaris の資源管理機能のうち、プロジェクトおよびタスク機能について説明します。プロジェクトとタスクは、作業負荷にラベル付けを行い、他の作業負荷と区別するために使用されます。プロジェクトは、関連した作業に対してネットワーク全体で適用される管理識別子を与えます。タスクは、プロセスのグループを、作業負荷コンポーネントを表す管理しやすいエンティティにまとめます。

概要

作業負荷の応答性を最適化するには、まず分析対象のシステム上で実行中の作業負荷を特定できなければなりません。この情報は、プロセス指向の手法とユーザー指向の手法のどちらか一方だけを使用して取得できるものではありません。Solaris 環境では、作業負荷を区別して特定するための 2 つの追加機能、プロジェクトとタスクを利用できます。

実行中のプロセスは、そのプロセスのプロジェクトメンバーシップまたはタスクメンバーシップに基づいて、Solaris の標準コマンドを使って操作できます。 拡張アカウンティング機能は、プロセスとタスクの両方の使用状況についてレポートを作成し、各レコードに管理用プロジェクト識別子のタグを付けることができます。この処理により、オフラインで行う作業負荷分析作業をオンラインでの監視作業と関連付けることができます。プロジェクト識別子は、`project` ネームサービスデータベースを介して複数のマシンで共有できます。したがって、最終的には、複数のマシン上で実行される (つまり複数のマシンにわたる) 関連した作業負荷の資源消費をすべてのマシンについて分析できます。

プロジェクト

プロジェクト識別子は、関連する作業を特定するために使用される管理識別子です。プロジェクト識別子は、ユーザー識別子やグループ識別子と同様な、作業負荷に付けられているタグと考えることができます。ユーザーまたはグループは 1 つ以上のプロジェクトに所属できます。プロジェクトは、ユーザーまたはユーザーグループが参加することを許可されている作業負荷を表すために使用します。このメンバーシップは、たとえば、使用状況や初期資源割り当てに基づくチャージバックの根拠となります。ユーザーにはデフォルトのプロジェクトを割り当てる必要がありますが、ユーザーが起動するプロセスは、ユーザーが属しているプロジェクトであれば、どのプロジェクトにでも関連付けることができます。

ユーザーのデフォルトプロジェクトの判定

システムにログインするには、そのユーザーにデフォルトのプロジェクトが割り当てられている必要があります。

システム上の各プロセスはプロジェクトのメンバーシップを所有しているため、ログインやその他の初期処理にデフォルトのプロジェクトを割り当てるアルゴリズムが必要です。デフォルトプロジェクトを判定するアルゴリズムは、4 つの手順から成ります。デフォルトプロジェクトが見つからない場合、ユーザーのログインまたはプロセスの開始要求は拒否されます。

システムは、次の手順でユーザーのデフォルトプロジェクトを判定します。

1. ユーザーが、`/etc/user_attr` 拡張ユーザー属性データベースで定義されている `project` 属性のエントリを持っている場合は、その `project` 属性の値がデフォルトプロジェクトになります (`user_attr(4)` 参照)。
2. `user.`*user-id* という名前のプロジェクトが `project(4)` データベースにある場合は、そのプロジェクトがデフォルトプロジェクトになります。
3. `group.`*group-name* というプロジェクトが `project` データベースにあり、*group-name* がユーザーのデフォルトのグループ名 (`passwd(4)` で指定されている) である場合は、そのプロジェクトがデフォルトプロジェクトになります。
4. `project` データベースに `default` という特別なプロジェクトがある場合は、そのプロジェクトがデフォルトプロジェクトになります。

このロジックは `getdefaultproj()` ライブラリ関数が提供します (`getprojent(3PROJECT)` 参照)。

project データベース

プロジェクトのデータは、ローカルファイル、ネットワーク情報サービス (NIS) のプロジェクトマップ、または LDAP (Lightweight Directory Access Protocol) ディレクトリサービスに保存できます。/etc/project データベースまたはネームサービスは、ログイン時、および PAM (Pluggable Authentication Module) によるアカウント管理に対する要求があったときに使用され、ユーザーをデフォルトのプロジェクトに結合します。

注 – プロジェクトデータベース内のエントリに対する更新は、/etc/project ファイルまたはネットワークネームサービスのデータベース表現のどちらに対するものであっても、現在アクティブなプロジェクトには適用されません。 更新内容は、login(1) または newtask(1) が使用されたときに、プロジェクトに新たに参加するタスクに適用されます。

PAM サブシステム

アカウント情報の変更や設定を行う操作には、システムへのログイン、rcp または rsh コマンドの呼び出し、ftp の使用、su の使用などがあります。 アカウント情報の変更や設定が必要な場合は、設定可能なモジュール群を使用して、認証、アカウント管理、資格管理、セッション管理などを行います。

プロジェクト用のアカウント管理 PAM モジュールについては、pam_projects(5) のマニュアルページを参照してください。PAM システムについては、pam(3PAM)、pam.conf(4) および pam_unix(5) のマニュアルページを参照してください。

ネームサービス構成

資源管理は、ネームサービス project データベースをサポートします。project データベースが格納されている場所は、/etc/nsswitch.conf で指定されます。デフォルトでは files が最初に指定されていますが、ソースは任意の順序で指定できます。

```
project: files [nis] [ldap]
```

プロジェクト情報に対して複数のソースが指定されている場合、ルーチンは最初に指定されているソースで情報を検索し、次にその後に続くデータベースを検索します。

/etc/nsswitch.conf の詳細については、『*Solaris* のシステム管理 (ネーミングとディレクトリサービス *: DNS*、*NIS*、*LDAP* 編)』の「pam_unix」および nsswitch.conf(4) のマニュアルページを参照してください。

ローカルの project ファイルの形式

nsswitch.conf で project データベースとして files を選択した場合、ログインプロセスはプロジェクト情報を /etc/project ファイルで検索します (projects (1) および project(4) 参照)。project ファイルには、システムによって認識されるプロジェクトごとにエントリが 1 行ずつ、次の形式で記述されています。

```
projname:projid:comment:user-list:group-list:attributes
```

フィールドの定義は次のとおりです。

projname プロジェクト名。英数字、下線 (_)、ハイフン (-) から成る文字列を指定します。最初の文字は英字でなければなりません。*projname* に、ピリオド (.)、コロン (:)、または改行文字を使用することはできません。

projid システムでプロジェクトに割り当てられる一意の数値 ID (PROJID)。*projid* フィールドの最大値は UID_MAX (2147483647) です。

comment プロジェクトの説明です。

user-list プロジェクトへの参加を許可されたユーザーをコンマで区切ったリストです。

このフィールドではワイルドカードを使用できます。 アスタリスク (*) を指定した場合、すべてのユーザーにプロジェクトへの参加を許可します。感嘆符に続けてアスタリスクを指定した場合 (!*)、すべてのユーザーのプロジェクトへの参加を拒否します。 感嘆符 (!) に続けてユーザー名を指定した場合、指定されたユーザーのプロジェクトへの参加を拒否します。

group-list プロジェクトへの参加を許可されたユーザーのグループをコンマで区切ったリストです。

このフィールドではワイルドカードを使用できます。 アスタリスク (*) を指定した場合、すべてのグループにプロジェクトへの参加を許可します。感嘆符に続けてアスタリスクを指定した場合 (!*)、すべてのグループのプロジェクトへの参加を拒否します。 感嘆符 (!) に続けてグループ名を指定した場合、指定されたグループのプロジェクトへの参加を拒否します。

attributes 名前と値の組をセミコロンで区切ったリストです (第 8 章参照)。*name* は、オブジェクトに関連する属性を指定する任意の文字列です。また *value* はその属性のオプション値です。

```
name[=value]
```

名前と値の組で、名前に使用できるのは、文字、数字、下線、ピリオドに制限されます。rctl のカテゴリとサブカテゴリの区切り文字にはピリオドを使用します。属性名の最初の文字は英字にする必要があります。名前については大文字と小文字は区別されます。

値を、コンマと括弧を使用して構造化することにより、優先順位を設定できます。セミコロンは、名前と値の組を区切るために使用されるので、値の定義には使用できません。コロンは、プロジェクトフィールドを区切る

ために使用されるので、値の定義には使用できません。

注 – このファイルを読み取るルーチンは、無効なエントリを検出すると停止します。このため、無効なエントリの後に指定されているプロジェクトの割り当ては実行されません。

次に、デフォルトの /etc/project ファイルの例を示します。

```
system:0:System:::
user.root:1:Super-User:::
noproject:2:No Project:::
default:3::::
group.staff:10::::
```

次に、デフォルトの/etc/project ファイルの最後にプロジェクトエントリを追加した例を示します。

```
system:0:System:::
user.root:1:Super-User:::
noproject:2:No Project:::
default:3::::
group.staff:10::::
user.ml:2424:Lyle Personal:::
booksite:4113:Book Auction Project:ml,mp,jtd,kjh::
```

/etc/project ファイルに資源制御を追加する方法については、96 ページの「資源制御の使用」を参照してください。

NIS のネームサービス構成

NIS を使用している場合は、NIS マップ内でプロジェクトを検索するように、/etc/nsswitch.conf ファイルで指定できます。

```
project: nis files
```

NIS マップの project.byname と project.bynumber はどちらも、次に示すように /etc/project ファイルと同じ形式です。

```
projname:projid:comment:user-list:group-list:attributes
```

詳細は、『*Solaris* のシステム管理 *(*ネーミングとディレクトリサービス *: DNS*、*NIS*、*LDAP* 編*)*』を参照してください。

LDAP のディレクトリサービス構成

LDAP を使用している場合は、LDAP エントリでプロジェクトを検索するように、`/etc/nsswitch.conf` ファイルで指定できます。

```
project: ldap files
```

LDAP データベースにおけるプロジェクトエントリのスキーマなどの詳細については、『*Solaris* のシステム管理 *(*ネーミングとディレクトリサービス *: DNS*、*NIS*、*LDAP* 編*)*』の「一般的なリファレンス」を参照してください。

タスク

プロジェクトへのログインが成功するたびに、ログインプロセスを含む新しい「タスク」が作成されます。タスクは、時間をかけて行われる一連の作業を表すプロセスです。また、タスクは作業負荷コンポーネントと考えることもできます。

各プロセスは、1 つのタスクのメンバーであり、各タスクは 1 つのプロジェクトに関連付けられています。

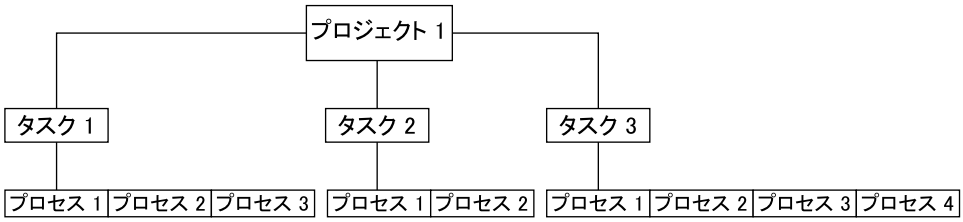


図 6–1 プロジェクトとタスクのツリー

シグナル送信のようなセッション上のすべての操作も、タスクでサポートされています。タスクをプロセッサセットに結合して、スケジューリングの優先順位とクラスを設定することにより、タスク内の現在のプロセスとそれに続くすべてのプロセスを変更することもできます。

タスクは、ログイン時に作成されるほか (`login(1)` 参照)、`cron(1M)`、`newtask(1)`、および `setproject(3PROJECT)` によっても作成されます。

拡張アカウンティング機能は、タスクレベルで集計されたプロセスのアカウンティングデータを提供できます。

プロジェクトとタスクの管理に使用するコマンド

コマンド名	説明
projects(1)	ユーザーのプロジェクトメンバーシップを表示する
newtask(1)	ユーザーのデフォルトのシェルまたは指定されたコマンドを実行し、指定されたプロジェクトが所有する新しいタスクに実行コマンドを配置する。また、newtask は、実行中のプロセスに結合するタスクとプロジェクトを変更するためにも使用できる
projadd(1M)	/etc/project ファイルに新しいプロジェクトエントリを追加する。projadd は、ローカルシステム上にだけプロジェクトエントリを作成する。projadd は、ネットワークネームサービスから提供される情報は変更できない
projmod(1M)	ローカルシステム上のプロジェクトの情報を変更する。projmod は、ネットワークネームサービスから提供される情報は変更できない。ただし、このコマンドは、外部のネームサービスに対してプロジェクト名とプロジェクト ID の一意性を確認する
projdel(1M)	ローカルシステムからプロジェクトを削除する。projdel は、ネットワークネームサービスから提供される情報は変更できない

プロジェクトとタスクで使用するコマンドオプション

ps

タスクおよびプロジェクトの ID を表示するには、ps -o を使用します。たとえば、プロジェクト ID を表示するには、次のように入力します。

```
# ps -o user,pid,uid,projid
USER PID   UID  PROJID
jtd  89430 124  4113
```

id

ユーザーおよびグループ ID に加えて、現在のプロジェクト ID を表示するには、`id -p` を使用します。*user* オペランドを指定した場合、そのユーザーの通常のログインに関連付けられたプロジェクトが表示されます。

```
#  id -p
uid=124(jtd) gid=10(staff) projid=4113(booksite)
```

pgrep と pkill

特定のリスト内のプロジェクト ID と一致するプロセスだけを表示するには、次のように入力します。

```
# pgrep -J projidlist
# pkill -J projidlist
```

特定のリスト内のタスク ID と一致するプロセスだけを表示するには、次のように入力します。

```
# pgrep -T taskidlist
# pkill -T taskidlist
```

prstat

システムで現在実行中のプロセスとプロジェクトのさまざまな統計情報を表示するには、次のように入力します。

```
% prstat -J
   PID USERNAME  SIZE   RSS STATE  PRI NICE      TIME  CPU PROCESS/NLWP
 21634 jtd      5512K 4848K cpu0    44    0   0:00.00 0.3% prstat/1
   324 root       29M   75M sleep   59    0   0:08.27 0.2% Xsun/1
 15497 jtd        48M   41M sleep   49    0   0:08.26 0.1% adeptedit/1
   328 root     2856K 2600K sleep   58    0   0:00.00 0.0% mibiisa/11
  1979 jtd      1568K 1352K sleep   49    0   0:00.00 0.0% csh/1
  1977 jtd      7256K 5512K sleep   49    0   0:00.00 0.0% dtterm/1
   192 root     3680K 2856K sleep   58    0   0:00.36 0.0% automountd/5
  1845 jtd        24M   22M sleep   49    0   0:00.29 0.0% dtmail/11
  1009 jtd      9864K 8384K sleep   49    0   0:00.59 0.0% dtwm/8
   114 root     1640K  704K sleep   58    0   0:01.16 0.0% in.routed/1
   180 daemon   2704K 1944K sleep   58    0   0:00.00 0.0% statd/4
   145 root     2120K 1520K sleep   58    0   0:00.00 0.0% ypbind/1
   181 root     1864K 1336K sleep   51    0   0:00.00 0.0% lockd/1
   173 root     2584K 2136K sleep   58    0   0:00.00 0.0% inetd/1
   135 root     2960K 1424K sleep    0    0   0:00.00 0.0% keyserv/4
PROJID    NPROC  SIZE   RSS MEMORY      TIME  CPU PROJECT
    10       52  400M  271M    68%   0:11.45 0.4% booksite
     0       35  113M  129M    32%   0:10.46 0.2% system

Total: 87 processes, 205 lwps, load averages: 0.05, 0.02, 0.02
```

システムで現在実行中のプロセスとタスクのさまざまな統計情報を表示するには、次のように入力します。

```
% prstat -T
   PID USERNAME  SIZE   RSS STATE  PRI NICE      TIME  CPU PROCESS/NLWP
 23023 root       26M   20M sleep   59    0   0:03:18 0.6% Xsun/1
 23476 jtd        51M   45M sleep   49    0   0:04:31 0.5% adeptedit/1
 23432 jtd      6928K 5064K sleep   59    0   0:00:00 0.1% dtterm/1
 28959 jtd        26M   18M sleep   49    0   0:00:18 0.0% .netscape.bin/1
 23116 jtd      9232K 8104K sleep   59    0   0:00:27 0.0% dtwm/5
 29010 jtd      5144K 4664K cpu0    59    0   0:00:00 0.0% prstat/1
   200 root     3096K 1024K sleep   59    0   0:00:00 0.0% lpsched/1
   161 root     2120K 1600K sleep   59    0   0:00:00 0.0% lockd/2
   170 root     5888K 4248K sleep   59    0   0:03:10 0.0% automountd/3
   132 root     2120K 1408K sleep   59    0   0:00:00 0.0% ypbind/1
   162 daemon   2504K 1936K sleep   59    0   0:00:00 0.0% statd/2
   146 root     2560K 2008K sleep   59    0   0:00:00 0.0% inetd/1
   122 root     2336K 1264K sleep   59    0   0:00:00 0.0% keyserv/2
   119 root     2336K 1496K sleep   59    0   0:00:02 0.0% rpcbind/1
   104 root     1664K  672K sleep   59    0   0:00:03 0.0% in.rdisc/1
TASKID    NPROC  SIZE   RSS MEMORY      TIME  CPU PROJECT
   222       30  229M  161M    44%   0:05:54 0.6% group.staff
   223        1   26M   20M   5.3%   0:03:18 0.6% group.staff
    12        1   61M   33M   8.9%   0:00:31 0.0% group.staff
     1       33   85M   53M    14%   0:03:33 0.0% system

Total: 65 processes, 154 lwps, load averages: 0.04, 0.05, 0.06
```

注 – -J オプションと -T オプションを一緒に使用することはできません。

プロジェクトとタスクでの cron と su の使用

cron

cron コマンドは、settaskid を発行し、実行を要求したユーザーの適切なデフォルトプロジェクトを使用して、cron、at、および batch の各ジョブが別のタスクで実行されるようにします。また、at および batch コマンドは、現在のプロジェクト ID を取得して at ジョブを実行するときにプロジェクト ID が復元されるようにします。

su

ユーザーのデフォルトプロジェクトを切り替え、その結果として新しいタスクを作成する (ログインのシミュレーションの一環として) には、次のように入力します。

```
# su - user
```

呼び出し側のプロジェクト ID を保持するには、- フラグを付けずに su コマンドを発行します。

```
# su user
```

プロジェクト管理の例

▼ プロジェクトを定義して現在のプロジェクトを表示する方法

次の例は、projadd および projmod コマンドを使用する方法を示します。

1. スーパーユーザーになります。
2. システムのデフォルトの /etc/project ファイルを表示します。

```
# cat /etc/project
system:0::::
user.root:1::::
noproject:2::::
default:3::::
group.staff:10::::
```

3. *booksite* というプロジェクトを追加し、追加したプロジェクトを *mark* という名前のユーザーにプロジェクト **ID** 番号 *4113* で割り当てます。

```
# projadd -U mark -p 4113 booksite
```

4. 再度 /etc/project ファイルを表示し、プロジェクトが追加されていることを確認します。

```
# cat /etc/project
system:0::::
user.root:1::::
noproject:2::::
default:3::::
group.staff:10::::
booksite:4113::mark::
```

5. **comment** フィールドにプロジェクトを説明するコメントを追加します。

```
# projmod -c 'Book Auction Project' booksite
```

6. /etc/project ファイルに加えた変更を確認します。

```
# cat /etc/project
system:0::::
user.root:1::::
```

```
noproject:2::::
default:3::::
group.staff:10::::
booksite:4113:Book Auction Project:mark::
```

▼ /etc/project ファイルからプロジェクトを削除する方法

次の例は、projdel コマンドを使ってプロジェクトを削除する方法を示します。

1. スーパーユーザーになります。
2. projdel コマンドを使ってプロジェクト *booksite* を削除します。

   ```
   # projdel booksite
   ```

3. /etc/project ファイルを表示します。

   ```
   # cat /etc/project
   system:0::::
   user.root:1::::
   noproject:2::::
   default:3::::
   group.staff:10::::
   ```

4. ユーザー名 *mark* でログインし、projects と入力して、割り当てられているプロジェクトを表示します。

   ```
   # su - mark
   # projects
   default
   ```

▼ ユーザーおよびプロジェクトのメンバーシップ情報を取得する方法

-p フラグを付けて id コマンドを使用し、起動中のプロセスの現在のプロジェクトメンバーシップを表示します。

```
$ id -p
uid=100(mark) gid=1(other) projid=3(default)
```

▼ 新しいタスクを作成する方法

1. スーパーユーザーになります。
2. システムのタスク **ID** を取得するための -v **(**冗長**)** オプションを付けた newtask コマンドを使用して、*booksite* プロジェクトに新しいタスクを作成します。

```
# newtask -v -p booksite
16
```

newtask を実行すると、指定したプロジェクト内に新しいタスクが作成され、そのタスクにユーザーのデフォルトのシェルが置かれます。

3. 起動中のプロセスの現在のプロジェクトメンバーシップを表示します。

```
# id -p
uid=100(mark) gid=1(other) projid=4113(booksite)
```

今度は、プロセスが新しいプロジェクトのメンバーになっています。

▼ 実行中のプロセスを新しいタスクに移動する方法

次の例は、実行中のプロセスを別のタスクとプロジェクトに関連付ける方法を示します。このタスクを実行するには、スーパーユーザーでなければなりません。または、プロセスの所有者で、かつ新しいプロジェクトのメンバーでなければなりません。

1. スーパーユーザーになります。
2. *book_catalog* プロセスのプロセス **ID** を取得します。

```
# pgrep book_catalog
    8100
```

3. プロセス *8100* を、新しいタスク **ID** を使って *booksite* プロジェクトに関連付けます。

```
# newtask -v -p booksite -c 8100
    17
```

-c オプションは、newtask が指定された既存のプロセスに対して動作することを指定します。

4. タスクとプロセス **ID** の対応を確認します。

```
# pgrep -T 17
    8100
```

第 7 章

拡張アカウンティング

プロジェクトおよびタスク機能 (第 6 章で説明) を使って作業負荷にラベル付けを行い、分離することにより、作業負荷ごとの資源消費を監視できます。「拡張アカウンティング」サブシステムを使用すると、プロセスとタスクの両方について詳細な資源消費統計情報を取得できます。 拡張アカウンティングサブシステムでは、行われた作業の対象プロジェクトの使用状況レコードにラベル付けします。 また、拡張アカウンティングを IPQoS (Internet Protocol Quality of Service、IP サービス品質) フローアカウンティングモジュールと組み合わせて、システム上のネットワークフロー情報を取得することもできます。 IPQoS フローアカウンティングモジュールについては、『*IPQoS* の管理』の「フローアカウンティングの使用と統計情報の収集 (手順)」を参照してください。

拡張アカウンティングを開始する方法については、85 ページの「プロセス、タスク、およびフローの拡張アカウンティングを起動する方法」を参照してください。

概要

資源管理メカニズムを適用する前に、まず、さまざまな作業負荷がシステムに対して行う資源消費要求の特徴をつかむ必要があります。Solaris オペレーティング環境の拡張アカウンティング機能は、システムやネットワークの資源消費をタスクまたはプロセスベース、または IPQoS が提供するセレクタベース (`ipqos(7IPP)` 参照) で記録する柔軟な方法を提供します。 システムの使用状況をリアルタイムで計測するオンライン監視ツールとは異なり、拡張アカウンティング機能を使用すると、使用状況の履歴を調べることができます。 その上で、将来の作業負荷の容量要件を算定できます。

拡張アカウンティングのデータを使用すれば、資源についての課金、作業負荷の監視、容量計画などの目的でソフトウェアを開発したり購入したりできます。

拡張アカウンティングの動作

Solaris 環境の拡張アカウンティング機能は、アカウンティングデータを含めるために、バージョン番号が付けられた拡張可能なファイル形式を使用します。このデータ形式を使用するファイルは、添付のライブラリ `libexacct` で提供される API を使ってアクセスまたは作成できます。作成されたファイルは、拡張アカウンティング機能を使用できる任意のプラットフォーム上で分析でき、データを容量計画やチャージバックに使用できます。

拡張アカウンティングを起動すると、`libexacct` API で調べることができる統計情報が収集されます。`libexacct` は、`exacct` ファイルを前後どちらの方向からも検査できます。API は、カーネルが作成するファイルだけでなく、`libexacct` によって生成されたサードパーティ製のファイルもサポートします。

拡張アカウンティングを有効にすると、タスクは、自分のメンバープロセスの総資源使用状況を追跡します。タスクのアカウンティングレコードは、そのタスクの完了時に書き込まれます。中間レコードを書き込むこともできます。タスクの詳細については、第 6 章を参照してください。

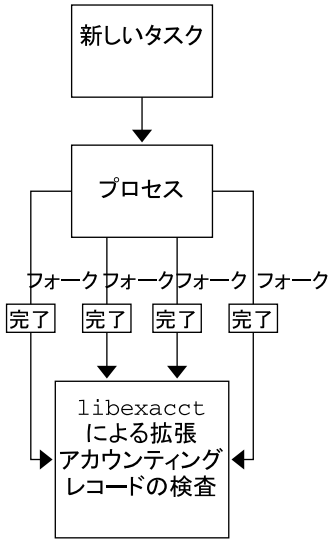


図 **7–1** 拡張アカウンティング起動時のタスクの追跡

拡張可能な形式

拡張アカウンティングの形式は、古い SunOS™ システムのアカウンティングソフトウェアの形式に比べて拡張性があります (『*Solaris* のシステム管理 (上級編)』の「システムアカウンティング」を参照)。拡張アカウンティングでは、システムアカウンティングメトリックスのシステムへの追加や削除をシステムの解放時またはシステムの操作中に行うことができます。

注 – 拡張アカウンティングと古いシステムのアカウンティングソフトウェアの両方をシステム上で同時に起動できます。

---

exacct レコードとその形式

exacct レコードを作成するルーチンは、次の 2 つの目的で使用できます。

- サードパーティ製の exacct ファイルを作成できるようにする
- putacct システムコールを使ってカーネルアカウンティングファイルに組み込まれるタグ付けレコードを作成できるようにする (getacct(2) 参照)。

この形式では、すべての変更を明示的なバージョン変更にしなくても、さまざまな形式のアカウンティングレコードを取得できます。アカウンティングデータを使用するアプリケーションは、認識不可能なレコードを無視するように作成する必要があります。

libexacct ライブラリは、ファイルを exacct 形式に変換し、exacct でファイルを作成します。このライブラリは、exacct 形式のファイルに対するインタフェースとしてサポートされている唯一のインタフェースです。

注 – getacct、 putacct、wracct の各システムコールは、フローには適用されません。 IPQoS フローアカウンティングの構成時には、カーネルによってフローレコードが作成され、ファイルに書き込まれます。

拡張アカウンティング構成

/etc/acctadm.conf ファイルには、現在の拡張アカウンティング構成が含まれます。このファイルは、ユーザーが直接編集するのではなく、acctadm インタフェースを介して編集します。

拡張アカウンティングデータは、デフォルトでは /var/adm/exacct ディレクトリに置かれます。acctadm(1M) コマンドを使用すると、プロセスやタスクのアカウンティングデータファイルの格納場所を変更できます。

拡張アカウンティングで使用されるコマンド

コマンド名	説明
acctadm(1M)	拡張アカウンティング機能のさまざまな属性の変更、拡張アカウンティングの停止と起動を行う。また、プロセス、タスク、およびフローを追跡するためのアカウンティング属性を選択するのに使用する
wracct(1M)	アクティブなプロセスおよびタスクの拡張アカウンティングアクティビティを書き込む

コマンド名	説明
lastcomm(1)	直前に呼び出されたコマンドを表示する。lastcomm では、標準アカウンティングプロセスのデータまたは拡張アカウンティングプロセスのデータのどちらかを使用できる

タスクおよびプロジェクト関連のコマンドについては、75 ページの「プロジェクトとタスクの管理に使用するコマンド」を参照してください。IPQoS フローアカウンティングについては、ipqosconf（1M）のマニュアルページを参照してください。

拡張アカウンティング機能の使用

▼ プロセス、タスク、およびフローの拡張アカウンティングを起動する方法

タスク、プロセス、およびフローの拡張アカウンティング機能を起動するには、acctadm(1M) コマンドを使用します。 acctadm の最後に付けられたオプションのパラメータは、このコマンドが、拡張アカウンティング機能のプロセスアカウンティングコンポーネント、システムタスクアカウンティングコンポーネント、フローアカウンティングコンポーネントのいずれに作用するかを示します。

1. スーパーユーザーになります。
2. プロセスの拡張アカウンティングを起動します。

   ```
   # acctadm -e extended -f /var/adm/exacct/proc process
   ```

3. タスクの拡張アカウンティングを起動します。

   ```
   # acctadm -e extended,mstate -f /var/adm/exacct/task task
   ```

4. フローの拡張アカウンティングを起動します。

   ```
   # acctadm -e extended -f /var/adm/exacct/flow flow
   ```

▼ 起動スクリプトを使って拡張アカウンティングを起動する方法

/etc/init.d/acctadm スクリプトへのリンクを /etc/rc2.d に作成することにより、実行中に拡張アカウンティングを起動できます。

```
# ln -s /etc/init.d/acctadm /etc/rc2.d/Snacctadm
# ln -s /etc/init.d/acctadm /etc/rc2.d/Knacctadm
```

変数 *n* は番号で置き換えられます。

アカウンティング構成の詳細については、84 ページの「拡張アカウンティング構成」を参照してください。

▼ 拡張アカウンティング状態を表示する方法

引数なしで acctadm と入力すると、拡張アカウンティング機能の現在の状態が表示されます。

```
# acctadm
                Task accounting: active
           Task accounting file: /var/adm/exacct/task
         Tracked task resources: extended
       Untracked task resources: none
             Process accounting: active
        Process accounting file: /var/adm/exacct/proc
      Tracked process resources: extended
    Untracked process resources: host,mstate
                Flow accounting: active
           Flow accounting file: /var/adm/exacct/flow
         Tracked flow resources: extended
       Untracked flow resources: none
```

この例では、システムタスクアカウンティングが拡張モードと mstate モードで動作しています。プロセスアカウンティングとフローアカウンティングは、拡張モードで動作しています。

注 – 拡張アカウンティングの文脈では、マイクロステート (mstate) は、プロセス状態の微小な変化を反映した拡張データを意味し、このデータはプロセス使用状況ファイルで利用できます (proc(4) 参照)。このデータは、プロセスの活動に関して、基本レコードや拡張レコードよりも詳細な情報を提供します。

▼ 使用可能なアカウンティング資源を表示する方法

使用可能な資源は、システムやプラットフォームによってさまざまです。-r オプションを使用すると、システム上の使用可能なアカウンティング資源を表示できます。

```
# acctadm -r
process:
extended pid,uid,gid,cpu,time,command,tty,projid,taskid,ancpid,wait-status,flag
basic    pid,uid,gid,cpu,time,command,tty,flag
task:
```

```
extended taskid,projid,cpu,time,host,mstate,anctaskid
basic    taskid,projid,cpu,time
flow:
extended
saddr,daddr,sport,dport,proto,dsfield,nbytes,npkts,action,ctime,lseen,projid,uid
basic    saddr,daddr,sport,dport,proto,nbytes,npkts,action
```

▼ プロセス、タスク、およびフローアカウンティングを停止する方法

プロセス、タスク、およびフローアカウンティングを停止するには、それぞれを個別にオフにします。

1. スーパーユーザーになります。
2. プロセスアカウンティングをオフにします。

   ```
   # acctadm -x process
   ```

3. タスクアカウンティングをオフにします。

   ```
   # acctadm -x task
   ```

4. フローアカウンティングをオフにします。

   ```
   # acctadm -x flow
   ```

5. タスクアカウンティング、プロセスアカウンティング、およびフローアカウンティングがオフになったことを確認します。

   ```
       # acctadm
               Task accounting: inactive
          Task accounting file: none
        Tracked task resources: extended
      Untracked task resources: none
             Process accounting: inactive
       Process accounting file: none
     Tracked process resources: extended
   Untracked process resources: host,mstate
                Flow accounting: inactive
           Flow accounting file: none
         Tracked flow resources: extended
       Untracked flow resources: none
   ```

第 8 章

資源制御

第 7 章で説明したようにシステム上の作業負荷の資源消費を判定したら、資源の使用方法に制限を設けて、作業負荷による資源の過剰消費を防ぐことができます。 資源制御は、UNIX の資源制限の概念を拡張した機能で、資源を制限するために使用される制約メカニズムです。

概要

従来から、UNIX システムには資源制限機能があります (*rlimit*)。rlimit の機能を使用すると、管理者は、プロセスが使用できる資源の量に対して 1 つ以上の数値制限を設定できます。この制限には、プロセスごとの CPU 使用時間、プロセスごとのコアファイルサイズ、プロセスごとの最大ヒープサイズが含まれます。ヒープサイズは、プロセスのデータセグメントに割り当てられるメモリー領域のサイズです。

Solaris オペレーティング環境では、プロセスごとの資源制限という概念が、第 6 章で説明したタスクおよびプロジェクトに拡張されています。この拡張された資源制限機能は、システム全体に適用できます。この拡張機能は、「資源制御」(*rctls*) 機能によって提供されます。

資源制御機能は、資源制限機能に対する互換インタフェースを提供します。資源制限機能を使用する既存のアプリケーションは、変更せずに、引き続き使用できます。また、既存のアプリケーションは、資源制御機能を利用するように変更されたアプリケーションと同様に監視することができます。

資源制御機能は、システム資源に対する制約メカニズムを提供します。これにより、プロセス、タスク、およびプロジェクトが、指定したシステム資源を過剰消費することを防止できます。このメカニズムは、資源の過剰消費を防ぐことにより、より管理しやすいシステムを実現します。

制約メカニズムは、容量計画を実施するときにも使用できます。制約を設けることにより、アプリケーションへの資源の提供を必ずしも拒否することなく、アプリケーションが必要とする資源量に関する情報を取得できます。

また、資源制御は、資源管理機能のための簡単な属性メカニズムとしても利用できます。たとえば、フェアシェアスケジューラ (FSS) のスケジューリングクラスで動作しているプロジェクトで利用できる CPU のシェア数は、資源制御 `project.cpu-shares` によって定義されます。プロジェクトは資源制御によって一定のシェア数を割り当てられるため、制御の超過につながる各種のアクションは許可されません。そのため、資源制御 `project.cpu-shares` の現在値は、指定したプロジェクトの属性とみなすことができます。

資源制御の管理

資源制御機能は、`project` データベース (第 6 章参照) によって構成されます。資源制御の属性は、`project` データベースエントリの最後のフィールドで設定します。各資源制御に対応付けられる値は、括弧で囲まれ、コンマ区切りのプレーンテキストとして示されます。 括弧内の値は「action 文節」を示します。各 action 文節は、特権レベル、しきい値、および特定のしきい値に対応付けられたアクションで構成されます。 各資源制御は複数の action 文節を持つことができ、各 action 文節もコンマで区切られます。 次のエントリは、プロジェクトエンティティにおけるプロセスごとのアドレス空間制限と、タスクごとの軽量プロセス (LWP) 制限を定義します。

```
development:101:Developers:::task.max-lwps=(privileged,10,deny);
  process.max-address-space=(privileged,209715200,deny)
```

`rctladm(1M)` コマンドを使用すると、資源制御機能の実行時に問い合わせや制御機能の変更を広域的に行うことができます。 `prctl(1)` コマンドを使用すると、資源制御機能の実行時に問い合わせや制御機能の変更をローカルに行うことができます。

使用可能な資源制御

次の表に、このリリースで使用できる標準の資源制御を示します。

この表では、各制御によって制約される資源について説明し、`project` データベースにおけるその資源のデフォルトの単位を示します。 デフォルトの単位には次の 2 種類があります。

- 数量は制限される量を意味します。
- インデックスは最大有効識別子を意味します。

したがって、`project.cpu-shares` は、プロジェクトで使用することが許可されているシェア数を示します。 一方、`process.max-file-descriptor` は、`open(2)` システムコールによってプロセスに割り当てられるファイルの最大数を示します。

表 **8–1** 標準の資源制御

制御名	説明	デフォルトの単位
`project.cpu-shares`	プロジェクトに対して、FSS (7) で使用することが許可されている CPU シェア数	数量 (シェア数)
`task.max-cpu-time`	タスクのプロセスで使用できる最大 CPU 時間	時間 (ミリ秒)
`task.max-lwps`	タスクのプロセスで同時に使用できる LWP の最大数	数量 (LWP 数)
`process.max-cpu-time`	プロセスで使用できる最大の CPU 時間	時間 (ミリ秒)
`process.max-file-descriptor`	プロセスで使用できる最大のファイル記述子インデックス	インデックス (最大ファイル記述子)
`process.max-file-size`	プロセスで書き込むことができるファイルオフセットの最大サイズ	サイズ (バイト)
`process.max-core-size`	プロセスによって作成されるコアファイルの最大サイズ	サイズ (バイト)
`process.max-data-size`	プロセスで使用できるヒープメモリーの最大サイズ	サイズ (バイト)
`process.max-stack-size`	プロセスで使用できるスタックメモリーセグメントの最大サイズ	サイズ (バイト)
`process.max-address-space`	プロセスで使用できる、セグメントサイズの総計としての最大アドレス空間	サイズ (バイト)

資源制御値と特権レベル

資源制御のしきい値は、ローカルアクションやロギングなどの広域アクションをトリガーできる実行ポイントを設定します。

各しきい値は、次の 3 種類の特権レベルのいずれかに対応付ける必要があります。

- 基本値 — 呼び出し元プロセスの所有者が変更できる
- 特権値 — 特権を持っている呼び出し元 (スーパーユーザー) だけが変更できる
- システム値 — オペレーティングシステムによる処理が実行されている間は、固定される

資源制御は、システムまたは資源の提供者によって定義されるシステム値を 1 つ持つことが保証されます。システム値は、オペレーティングシステムが提供できる資源の量を意味します。

特権値はいくつでも定義できます。基本値は 1 つだけ許可されます。特権値を指定しないで実行される操作には、デフォルトで、基本レベルの特権が割り当てられます。

資源制御値の特権レベルは、資源制御ブロックの特権フィールドで、RCTL_BASIC、RCTL_PRIVILEGED、または RCTL_SYSTEM のように定義します。詳細は、`getrctl(2)` を参照してください。`prctl` コマンドを使用すると、基本レベルおよび特権レベルに対応付けられている値を変更できます。

資源制御値に対応付けられたアクション

資源制御に設定された各しきい値に対して、1 つ以上のアクションを対応付けることができます。

- しきい値を超える量の資源要求を拒否できる
- しきい値に達した場合は、違反プロセスまたは監視プロセスにシグナルを送信できる

実装上の制限により、しきい値に設定できるアクションは、各制御のグローバルプロパティによって制限されます。次の表に、使用できるシグナルアクションを示します。シグナルの詳細については、`signal(3HEAD)` を参照してください。

表 **8–2** 資源制御値に使用できるシグナル

シグナル	注
SIGABRT	
SIGHUP	
SIGTERM	
SIGKILL	
SIGSTOP	
SIGXRES	
SIGXFSZ	RCTL_GLOBAL_FILE_SIZE プロパティ (`process.max-file-size`) を持つ資源制御だけで使用可能。詳細は `rctlblk_set_value(3C)` を参照
SIGXCPU	RCTL_GLOBAL_CPUTIME プロパティ (`process.max-cpu-time`) を持つ資源制御だけで使用可能。詳細は `rctlblk_set_value(3C)` を参照

資源制御のフラグとプロパティ

システム上の資源制御には、それぞれ特定のプロパティセットが対応付けられています。このプロパティセットは、一連のグローバルフラグとして定義されます。グローバルフラグは、その資源が制御されているすべてのインスタンスに対応付けられます。グローバルフラグは変更できませんが、`rctladm` または `getrctl` システムコールを使って取得できます。

ローカルフラグは、特定のプロセスまたはプロセス集合に対する資源制御の特定のしきい値について、デフォルトの動作と構成を定義します。あるしきい値のローカルフラグが、同じ資源制御で定義されている別のしきい値の動作に影響することはありません。ただし、グローバルフラグは、特定の制御に対応付けられているすべての値の動作に影響します。ローカルフラグは、対応するグローバルフラグによる制約の範囲内で、`prctl` コマンドまたは `setrctl` システムコールを使って変更できます (`setrctl(2)` 参照)。

ローカルフラグ、グローバルフラグ、およびそれらの定義の詳細なリストについては、`rctlblk_set_value(3C)` を参照してください。

特定の資源制御がしきい値に達したときのシステムの動作を確認するには、`rctladm` を使ってその資源制御のグローバルフラグを表示します。たとえば、`process.max-cpu-time` の値を表示するには、次のように入力します。

```
$ rctladm process.max-cpu-time
    process.max-cpu-time   syslog=off   [ lowerable no-deny cpu-time inf ]
```

広域フラグは、次のことを示します。

`lowerable`	この制御の特権値を下げるのに、スーパーユーザー特権を必要としない
`no-deny`	しきい値を超えても、資源へのアクセスは拒否されない
`cpu-time`	資源がしきい値に達したとき、SIGXCPU を送信できる
`inf`	RCTL_LOCAL_MAXIMAL が設定されている値は、実際には無限数を意味し、制約はない

資源制御のローカル値とアクションを表示するには、`prctl` を使用します。

```
$ prctl -n process.max-cpu-time $$
    353939: -ksh
    process.max-cpu-time   [ lowerable no-deny cpu-time inf ]
        18446744073709551615 privileged signal=XCPU   [ max ]
        18446744073709551615 system     deny          [ max ]
```

この例では、2 つのしきい値の両方に `max` (RCTL_LOCAL_MAXIMAL) フラグが設定されており、資源制御には `inf` (RCTL_GLOBAL_INFINITE) フラグが設定されています。したがって、設定されているように、両方のしきい値は無限値を意味し、これらの値を上回ることはありません。

資源制御の実行

1 つの資源には、複数の資源制御を設定できます。 資源制御は、プロセスモデルの包含レベルごとに 1 つずつ設定できます。 同じ資源上の異なるコンテナレベルで資源制御がアクティブな場合、まず、最も小さいコンテナの制御が実行されます。したがって、`process.max-cpu-time` と `task.max-cpu-time` の両方の制御が同時に検出された場合は、まず `process.max-cpu-time` に対するアクションが実行されます。

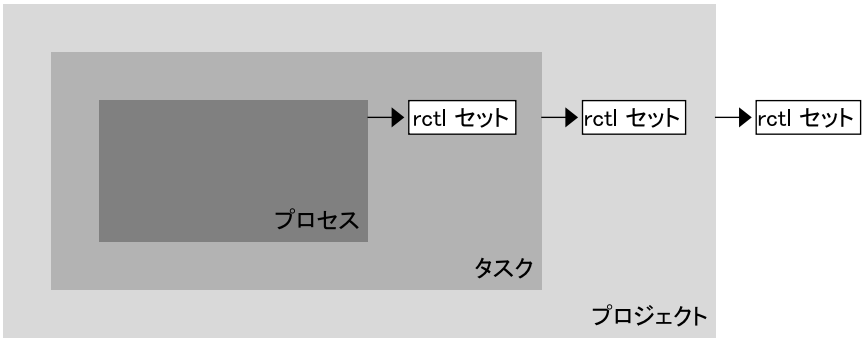


図 **8–1** プロセス集合、コンテナの包含関係、およびその資源制御セット

資源制御イベントの広域監視

プロセスの資源消費が不明な場合がよくあります。 資源消費に関する情報を入手するには、`rctladm(1M)` で利用できる広域資源制御アクションを使用します。 `rctladm` を使用して、資源制御に `syslog` アクションを設定します。 その資源制御が管理するエンティティでしきい値が検出されると、設定したログレベルでシステムメッセージが記録されます。

構成

表 8–1 に示されている各資源制御をプロジェクトに割り当てることができるのは、ログイン時、`newtask(1)` が呼び出されたとき、または `at(1)`、`batch` (`at(1)` を参照)、`cron(1M)` など、プロジェクトを扱うことができる起動ツールが呼び出されたときです。開始される各コマンドは、呼び出し側のユーザーのデフォルトプロジェクトとは異なるタスクで起動されます。

project データベース内のエントリに対する更新は、/etc/project ファイルまたはネットワークネームサービスのデータベース表現のどちらに対するものであっても、現在アクティブなプロジェクトには適用されません。 更新内容は、新しいタスクが login(1) または newtask によってプロジェクトに参加したときに適用されます。

動作中のシステム上の資源制御値を一時的に更新する

project データベースで変更された値は、プロジェクト内で開始される新しいタスクに対してだけ有効になります。ただし、rctladm および prctl コマンドを使用すると、動作中のシステムの資源制御を更新できます。

ログ状態の更新

rctladm コマンドは、システム全体で、各資源制御の広域ログ状態に影響を与えます。このコマンドは、広域的状態を表示し、制御の限度を超えたときに syslog が記録するログのレベルを設定できます。

資源制御の更新

prctl コマンドを使用すると、プロセスごと、タスクごと、またはプロジェクトごとに資源制御値とアクションを表示したり、一時的に変更したりできます。プロジェクト ID、タスク ID、またはプロセス ID を入力として指定すると、このコマンドは、定義されているレベルで資源制御に対して動作します。

変更した値とアクションはすぐに適用されます。ただし、これらの変更が適用されるのは、現在のセッションだけです。変更内容は、project データベースには記録されません。システムを再起動すると、変更内容は失われます。資源制御を永続的に変更するには、project データベースで変更を行う必要があります。

project データベースで変更できる資源制御設定はすべて、prctl コマンドを使って変更することもできます。基本値と特権値はどちらも、追加、削除が可能で、またアクションも変更できます。デフォルトでは、基本レベルの資源制御はすべての操作の影響を受けます。スーパーユーザー特権があるプロセスとユーザーは、特権レベルの資源制御も変更できます。システム資源の制御は変更できません。

資源制御の使用

▼ プロジェクト内の各タスクの最大 LWP 数を設定する方法

/etc/project データベースで次のエントリを入力し、*x-files* プロジェクトの各タスクの最大 LWP 数を *3* に設定します。

```
x-files:100::root::task.max-lwps=(privileged,3,deny)
```

スーパーユーザーが newtask を使って *x-files* プロジェクトに参加することにより、このプロジェクトに新しいタスクを作成した場合、スーパーユーザーは、次の注釈付きのサンプルセッションからもわかるように、実行中はこのタスク内に LWP を 3 つまでしか作成できません。

```
# newtask -p x-files csh

# prctl -n task.max-lwps $$
688: csh
task.max-lwps
                           3 privileged deny
                  2147483647 system     deny
# id -p
uid=0(root) gid=1(other) projid=100(x-files)

# ps -o project,taskid -p $$
 PROJECT TASKID
 x-files   236

# csh        /* 2 つ目の LWP を作成 */

# csh        /* 3 つ目の LWP を作成 */

# csh        /* これ以上 LWP を作成することはできない */
Vfork failed

#
```

▼ プロジェクトに複数の制御を設定する方法

/etc/project ファイルには、各プロジェクトごとに複数の資源制御設定を記述でき、さらに各資源制御ごとに複数のしきい値を記述できます。しきい値は action 文節で定義されます。複数の値はコンマで区切られます。

ファイル内の次の行は、basic (基本) レベルの制御を設定します。この設定では、*x-files* プロジェクトのタスクごとの最大 LWP 数に対して、アクションは発生しません。また、タスクごとの最大 LWP 数に対して特権レベルの deny 制御を設定しています。この制御により、前述の例のように、最大数を超える数の LWP を作成しようとすると失敗します。最後に、プロセスごとの最大ファイル記述子は basic レベルに制限されており、最大値を超えるオープンコールは失敗します。

```
x-files:101::root::task.max-lwps=(basic,10,none),(privileged,500,deny);
    process.max-file-descriptor=(basic,128,deny)
```

▼ prctl を使用する方法

スーパーユーザーは、prctl と入力することにより、実行中の現在のシェルの最大ファイル記述子を表示できます。

```
# prctl -n process.max-file-descriptor $$
8437:   sh
process.max-file-descriptor              [ lowerable deny ]
                         256 basic      deny
                       65536 privileged deny
                  2147483647 system     deny
```

prctl コマンドを使って新しい特権値を一時的に追加し、*x-files* プロジェクトの各タスクで 4 つ以上の LWP の使用を拒否することもできます。結果は 96 ページの「プロジェクト内の各タスクの最大 LWP 数を設定する方法」の結果と同じです。次の注釈付きサンプルセッションでこれを示します。

```
# newtask -p x-files

# id -p
uid=0(root) gid=1(other) projid=101(x-files)

# prctl -n task.max-lwps -t privileged -v 3 -e deny -i project x-files

# prctl -n task.max-lwps -i project x-files
670:    sh
task.max-lwps
                       3 privileged deny
                2147483647 system     deny
```

prctl -r を使って資源制御の最小値を変更することもできます。

```
# prctl -n process.max-file-descriptor -r -v 128 $$
```

▼ rctladm を使用する方法

rctladm を使用すると、資源制御のグローバル syslog 属性を有効にできます。制御が限界を超えたとき、指定された syslog レベルで通知が記録されます。次のコマンドを入力します。

```
# rctladm -e syslog process.max-file-descriptor
```

容量に関する警告

資源制御に対してグローバルアクションを設定すると、資源制御値を超えたエンティティに関する通知を受け取ることができます。

たとえば、一般的な作業負荷のための十分な CPU 資源が Web サーバーに割り当てられているかどうかを確認する場合を考えます。この容量は、sar(1) データで CPU のアイドル時間と平均負荷率を分析すれば判定できます。 また、拡張アカウンティングデータを調べて、Web サーバープロセスで同時に実行しているプロセス数を確認することもできます。

より簡単な方法は、Web サーバーをタスクに配置することです。その上で、syslog を使ってグローバルアクションを設定すると、タスクがマシンのパフォーマンスに適した LWP の計画数を上回ったときに、警告が通知されます。

▼ Web サーバーに十分な CPU 容量が割り当てられているかどうかを判定する方法

1. prctl コマンドを使用して、httpd プロセスを含むタスクにスーパーユーザーが所有する特権レベルの資源制御を設定します。各タスクの **LWP** の総数を **40** に制限し、すべてのローカルアクションを無効にします。

   ```
   # prctl -n task.max-lwps -v 40 -t privileged -d all `pgrep httpd`
   ```

2. 資源制御 task.max-lwps で、システムログのグローバルアクションを有効にします。

   ```
   # rctladm -e syslog task.max-lwps
   ```

3. 作業負荷が資源制御を超えるかどうかを監視します。

   作業負荷が資源制御を超えると、次のような内容が /var/adm/messages に記録されます。

   ```
   Jan  8 10:15:15 testmachine unix: [ID 859581 kern.notice]
   NOTICE: privileged rctl task.max-lwps exceeded by task 19
   ```

第 9 章

フェアシェアスケジューラ

作業負荷データを分析することによって、特定の作業負荷または作業負荷のグループが CPU 資源を占有しているかどうかを判定できます。作業負荷が CPU 使用量の制限を超えていない場合は、システム上での CPU 時間の割り当て方針を変更することができます。この章で説明するフェアシェアスケジューリングクラスを使用すると、タイムシェアリング (TS) スケジューリングクラスの優先順位方式ではなく、シェア数に基づいて CPU 時間を割り当てることができます。

概要

オペレーティングシステムの基本的な仕事は、どのプロセスがシステム資源へのアクセスを取得できるようにするか調整することです。プロセススケジューラ (別名、ディスパッチャ) は、カーネルの一部であり、プロセスへの CPU の割り当てを制御します。スケジューラには、スケジューリングクラスという概念があります。各スケジューリングクラスでは、クラス内のプロセスのスケジューリングに使用するスケジューリング方針を定義します。TS スケジューラは Solaris オペレーティング環境におけるデフォルトのスケジューラであり、使用可能な CPU へのアクセスをすべてのプロセスに等しく与えます。ただし、特定のプロセスにより多くの資源を与えたい場合もあります。

フェアシェアスケジューラ (FSS) では、各作業負荷に対する使用可能な CPU 資源の割り当てを、その作業負荷の重要性に基づいて制御します。この重要性は、各作業負荷に割り当てる CPU 資源のシェア数で表します。

各プロジェクトに CPU シェアを与えて、CPU 資源に対するプロジェクトの使用権を制御します。FSS では、プロジェクトに属するプロセス数ではなく、割り当てられたシェア数に基づいて、プロジェクト間に CPU 資源が公平に配分されることが保証されています。FSS は、他のプロジェクトとの比較に基づいて、CPU 資源を多く使用するプロジェクトの CPU 使用権を減らし、CPU 資源の使用が少ないプロジェクトの CPU 使用権を増やすことで公平さを実現します。

FSS は、カーネルスケジューリングクラスモジュールとクラスに固有な `dispadmin(1M)` および `priocntl(1)` コマンドから構成されます。FSS が使用するプロジェクトシェアは、`project` データベース内の `project.cpu-shares` プロパティで指定します。

CPU シェアの定義

「シェア」という用語は、プロジェクトに割り当てられる CPU 資源の配分を定義するために使用されます。プロジェクトに割り当てる CPU シェア数を他のプロジェクトよりも多くすると、そのプロジェクトがフェアシェアスケジューラから受け取る CPU 資源も多くなります。

CPU シェアは、CPU 資源の比率ではありません。シェアは、他の作業負荷との比較に基づいた作業負荷の相対的な重要性を定義します。プロジェクトに CPU シェアを割り当てる場合に重要なことは、プロジェクトが持つシェア数自体ではありません。他のプロジェクトと比較して、そのプロジェクトがシェアをいくつ持っているかを把握することが重要です。 また、そのプロジェクトが CPU 資源について、他のいくつのプロジェクトと競合しているかということも考慮に入れる必要があります。

注 – シェア数がゼロのプロジェクトに属するプロセスは、常に最下位のシステム優先順位 (0) で実行されます。このようなプロセスが実行されるのは、シェア数がゼロでないプロジェクトが CPU 資源を使用していないときだけです。

CPU シェアとプロセスの状態

Solaris オペレーティング環境では、プロジェクトの作業負荷は一般に複数のプロセスから構成されます。フェアシェアスケジューラの観点からは、各プロジェクトの作業負荷は、アイドル状態かアクティブ状態のどちらかです。プロジェクトのどのプロセスも CPU 資源を使用していないとき、プロジェクトはアイドル状態であるといいます。 このような場合、プロセスは一般にスリープ (入出力の完了を待機している状態) または停止状態にあります。プロジェクトの 1 つ以上のプロセスが CPU 資源を使用しているとき、プロジェクトはアクティブ状態であるといいます。 すべてのアクティブなプロジェクトが持つシェア数の合計が、プロジェクトに割り当てられる CPU 資源の配分の計算に使用されます。

次の式から、FSS スケジューラによるプロジェクトごとの CPU 資源の割り当て方法が算出されます。(allocation = 割り当て、project = プロジェクト、share = シェア)

$$allocation_{project\,i} = \frac{shares_{project\,i}}{\sum_{j=1...n} (shares_{project\,j})}$$

j : すべてのアクティブプロジェクトのインデックス

図 **9–1** FSS スケジューラのシェア計算

アクティブなプロジェクトが増えると、各プロジェクトの CPU 割り当ては減りますが、プロジェクト間の CPU 割り当て比率は変わりません。

CPU シェアと使用率

シェア割り当ては、使用率とは異なります。CPU 資源の 50% が割り当てられているプロジェクトの CPU 使用率は、平均するとわずか 20% ほどです。その上、シェアが CPU 使用量を制限するのは、他のプロジェクトと競合するときだけです。プロジェクトに対する割り当てが低い場合でも、そのプロジェクトがシステムで単独に実行されているときは、常に 100% の処理能力を CPU から受け取ります。使用可能な CPU サイクルが浪費されることはありません。つまり、使用可能な CPU サイクルはプロジェクト間に配分されます。

動作中の作業負荷に小さいシェアを割り当てると、パフォーマンスが低下します。ただし、システムが過負荷にならないかぎり、シェア割り当て数が原因で作業が完了しないことはありません。

CPU シェアの例

2 つの CPU を搭載したシステムがあり、それらの CPU は CPU にバインドされた 2 つの作業負荷 *A* および *B* を並列に実行しているとします。各作業負荷は別個のプロジェクトとして実行されています。各プロジェクトは、プロジェクト *A* に S_A シェアが割り当てられ、プロジェクト *B* に S_B シェアが割り当てられるように構成されています。

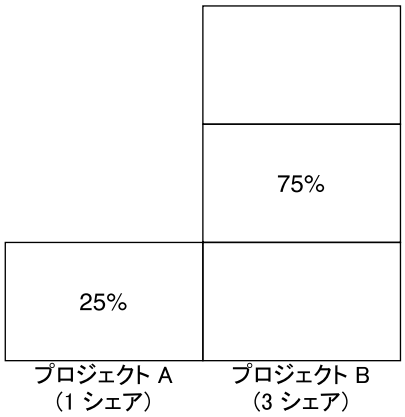
従来の TS スケジューラを使用した場合、システムで実行されている各作業負荷には、平均して同じ量の CPU 資源が与えられます。つまり、各作業負荷にはシステム容量の 50% が割り当てられます。

FSS スケジューラの制御で実行する場合でも、$S_A = S_B$ のシェアを割り当てると、各プロジェクトにほぼ等量の CPU 資源が与えられます。これに対して、プロジェクトに異なるシェア数を与えた場合、CPU 資源の割り当て量は異なります。

次に示す 3 つの例は、さまざまな構成でのシェアの働きを示しています。これらの例に示されているとおり、シェア数は、要求が使用可能な資源量と同じまたはそれを超えている場合にのみ使用量を数学的に正確に表します。

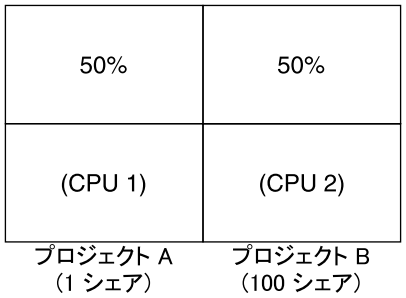
例 1: CPU にバインドされた 2 つのプロセスが各プロジェクトに存在する場合

プロジェクト *A* および *B* がそれぞれ CPU にバインドされたプロセスを 2 つ持ち、かつ $S_A = 1$ および $S_B = 3$ である場合、シェアの合計数は 1 + 3 = 4 になります。 この構成で、十分な数の CPU 要求があると、プロジェクト *A* および *B* には、それぞれ CPU 資源の 25% と 75% が割り当てられます。



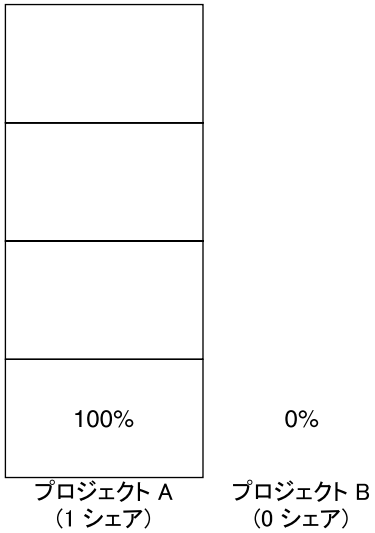
例 2: プロジェクト間に競合がない場合

プロジェクト *A* および *B* がそれぞれ CPU にバインドされたプロセスを *1* つだけ持ち、かつ $S_A = 1$ および $S_B = 100$ である場合、シェアの合計数は 101 になります。 各プロジェクトは、実行中のプロセスを 1 つしか持たないため、CPU を 1 つしか使用できません。この構成では、CPU 資源を得るための競合がプロジェクト間に存在しないので、プロジェクト *A* および *B* には、それぞれ全 CPU 資源の 50% が割り当てられます。この構成の場合、CPU シェア数は CPU 資源の割り当てに影響しません。プロジェクトへの割り当ては同じ (50/50) になります。これは、両方のプロジェクトに割り当てられるシェア数がゼロの場合でも同様です。



例 3: 一方のプロジェクトが実行されない場合

プロジェクト *A* および *B* がそれぞれ CPU にバインドされたプロセスを 2 つ持ち、かつ *A* に 1 シェア、*B* に 0 シェアが与えられている場合、プロジェクト *B* には CPU 資源が割り当てられず、プロジェクト *A* にすべての CPU 資源が割り当てられます。プロジェクト *B* のプロセスは常にシステム優先順位 0 で実行されるため、実行される可能性はまったくありません。これは、プロジェクト *A* のプロセスの方が常に高い優先順位を持っているためです。



FSS の構成

プロジェクトとユーザー

プロジェクトは、FSS スケジューラの作業負荷コンテナです。プロジェクトに割り当てられているユーザーのグループは、個別の管理可能なブロックとして扱われます。個人ユーザー用に独自のシェア数を持つプロジェクトを作成できます。

ユーザーは、異なるシェア数が割り当てられているさまざまなプロジェクトのメンバーになることができます。プロセスをあるプロジェクトから別のプロジェクトに移動すると、プロセスに割り当てられる CPU 資源量は変化します。

project データベースとネームサービスについては、71 ページの「project データベース」を参照してください。

CPU シェアの構成

CPU シェアの構成は project データベースのプロパティとして、ネームサービスによって管理されます。

プロジェクトに関連付けられている最初のタスクまたはプロセスが setproject (3PROJECT) ライブラリ関数を使って生成されると、project データベース内で資源制御 project.cpu-shares として定義されている CPU シェア数がカーネルに渡されます。資源制御 project.cpu-shares が定義されていないプロジェクトには、1 シェアが割り当てられます。

次の例では、/etc/project ファイル内のエントリでプロジェクト *x-files* のシェア数に *5* が設定されています。

```
x-files:100::::project.cpu-shares=(privileged,5,none)
```

プロジェクトに割り当てられている CPU シェア数を、プロセスの実行中にデータベースで変更しても、プロジェクトのシェア数は、その時点では変更されません。 変更内容を有効にするには、プロジェクトを再起動する必要があります。

プロジェクトに割り当てられているシェア数を、project データベース内のプロジェクトの属性を変更しないで一時的に変更するには、prctl(1) を使用します。たとえば、*x-files* プロジェクトに関連付けられているプロセスの実行中に、そのプロジェクトの資源制御 project.cpu-shares の値を *3* に変更するには、次のように入力します。

```
# prctl -r -n project.cpu-shares -v 3 -i project x-files
```

-r 指定された資源制御の現在の値を置き換えます。

`-n` *name*　資源制御の名前を指定します。

`-v` *val*　資源制御の値を指定します。

`-i` *idtype*　ID タイプを指定します。

x-files　変更対象を指定します。この例では、プロジェクト *x-files* が変更対象です。

プロジェクト ID 0 のプロジェクト `system` には、起動時の初期化スクリプトで起動されるすべてのシステムデーモンが含まれます。`system` は、無制限のシェア数を持つプロジェクトとしてみなされます。したがって、プロジェクト `system` は、他のプロジェクトに与えられているシェア数とは関係なく、常に最初にスケジュールされます。プロジェクト `system` に無制限のシェア数を割り当てない場合は、`project` データベースでこのプロジェクトのシェア数を変更します。

前述のように、シェア数がゼロのプロジェクトに属するプロセスには、常にシステム優先順位 0 が与えられます。シェア数が 1 以上のプロジェクトは、優先順位 1 以上で実行されます。したがって、シェア数がゼロのプロジェクトは、ゼロ以外のシェア数を持つプロジェクトが CPU 資源を要求していないときにだけスケジュールされます。

1 つのプロジェクトに割り当てられるシェアの最大数は 65535 です。

FSS とプロセッサセット

FSS は、プロセッサセットと連携して使用すると、連携させない場合よりも、各プロセッサセット上で実行するプロジェクト間の CPU 資源の割り当てをよりきめ細かく制御できます。FSS スケジューラは、プロセッサセットを完全に独立したパーティションとして処理します。つまり、各プロセッサセットは、CPU 割り当てについてそれぞれ個別に制御されます。

1 つのプロセッサセットで実行されるプロジェクトの CPU 割り当てが、別のプロセッサセットで実行されるプロジェクトの CPU シェアや動作によって影響を受けることはありません。なぜなら、異なるプロセッサセットで実行されるプロジェクトが同じ資源について競合することはないからです。競合が発生するのは、プロジェクトが同じプロセッサセット内で実行されている場合だけです。

プロジェクトに割り当てられているシェア数はシステム全体に適用されます。どのプロセッサセットで実行されようと、プロジェクトの各部分には同じシェア数が与えられます。

次に示すように、プロセッサセットが使用されている場合、プロジェクトの CPU 割り当ては、各プロセッサセット内で実行されるアクティブなプロジェクトに対して算出されます。(allocation = 割り当て、project = プロジェクト、share = シェア、processor set = プロセッサセット)

$$\text{allocation}_{\text{project}_x^i} = \frac{\text{shares}_{\text{project}_x^i}}{\text{processor set } X \quad \sum_{j=1...n} (\text{shares}_{\text{project}^j})}$$

j : プロセッサセット X で実行される
すべてのアクティブプロジェクトのインデックス

図 **9–2** プロセッサセットを使用する場合の FSS スケジューラのシェア計算

異なるプロセッサセット内で実行されるプロジェクトのパーティションは、異なる CPU 割り当てを持つことになります。1 つのプロセッサセット内の各プロジェクトパーティションに対する CPU 割り当ては、同じプロセッサセット内で実行される他のプロジェクトの割り当てにだけ依存します。

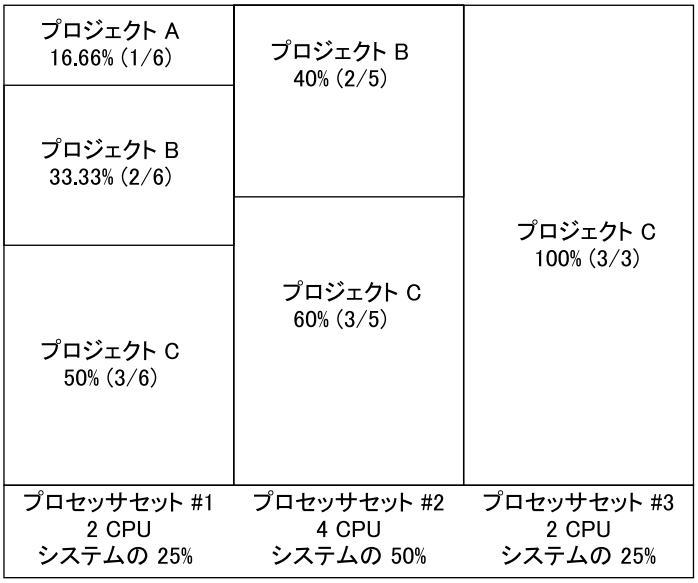
プロセッサセット境界内で実行されるアプリケーションのパフォーマンスと可用性が、新しいプロセッサセットの導入によって影響を受けることはありません。また、他のプロセッサセットで実行されるプロジェクトのシェア割り当ての変更によって、アプリケーションが影響を受けることもありません。

空のプロセッサセット (プロセッサが存在しないセット) や、プロセッサセットにバインドされたプロセスを持たないプロセッサセットは、FSS スケジューラの動作にまったく影響を与えません。

FSS とプロセッサセットの例

8 つの CPU を持つサーバーがプロジェクト *A*、*B*、および *C* 内で CPU にバインドされたアプリケーションをいくつか実行しているものとします。プロジェクト *A* には 1 シェア、プロジェクト *B* には 2 シェア、プロジェクト *C* には 3 シェアがそれぞれ割り当てられています。

プロジェクト *A* は、プロセッサセット 1 だけで実行されています。 プロジェクト *B* は、プロセッサセット 1 および 2 で実行されています。 プロジェクト *C* は、プロセッサセット 1、2、および 3 で実行されています。各プロジェクトには、使用可能なすべての CPU 処理能力を利用するだけの十分な数のプロセスが存在しているものとします。したがって、CPU 資源を得るための競合が各プロセッサセットで常に発生します。



このようなシステムでは、システム全体でのプロジェクトの CPU 割り当ての合計は、次のようになります。(pset = プロセッサセット)

プロジェクト A	$4\% = (1/6 \times 2/8)_{pset1}$
プロジェクト B	$28\% = (2/6 \times 2/8)_{pset1} + (2/5 \times 4/8)_{pset2}$
プロジェクト C	$67\% = (3/6 \times 2/8)_{pset1} + (3/5 \times 4/8)_{pset2} + (3/3 \times 2/8)_{pset3}$

これらの割合は、プロジェクトに与えられている CPU シェア値とは一致しません。ただし、各プロセッサセット内では、プロジェクトごとの CPU 割り当て比率はプロジェクトのそれぞれのシェアに比例します。

このシステム上にプロセッサセットが存在しない場合、CPU 資源の配分は、次に示すように、異なったものになります。

プロジェクト A	16.66% = (1/6)
プロジェクト B	33.33% = (2/6)
プロジェクト C	50% = (3/6)

FSS と他のスケジューリングクラスの併用

デフォルトでは、FSS スケジューリングクラスは、タイムシェアリング (TS)、対話型 (IA)、および固定優先順位 (FX) の各スケジューリングクラスと同じ範囲の優先順位 (0 から 59) を使用します。そのため、これらのスケジューリングクラスのプロセスが同じプロセッサセットを共有しないようにする必要があります。FSS、TS、IA、および FX の各クラスにプロセスが混在すると、予期せぬスケジューリング処理が実行される場合があります。

プロセッサセットを使用する場合は、1 つのシステム内で TS、IA、および FX を FSS と混在させることができます。ただし、各プロセッサセットで実行されるすべてのプロセスは、*1* つのスケジューリングクラスに所属している必要があります。このようにすると、これらのプロセスが同じ CPU について競合することはありません。プロセッサセットを使用しない場合は、特に FX スケジューラを FSS スケジューリングクラスと併用しないようにしてください。これにより、FX クラスのアプリケーションが高い優先順位を使用して、FSS クラスのアプリケーションの実行を妨げることはありません。

TS クラスと IA クラスのプロセスは、同じプロセッサセット内で、またはプロセッサセットが存在しない同じシステム内で混在させることができます。

Solaris オペレーティング環境では、スーパーユーザー権限を持つユーザーは、リアルタイム (RT) スケジューラも利用できます。デフォルトでは、RT スケジューリングクラスは FSS とは異なる範囲のシステム優先順位 (通常は 100 から 159) を使用します。RT と FSS は互いに重複しない範囲の優先順位を使用しているので、FSS は同じプロセッサセット内の RT スケジューリングクラスと共存できます。ただし、FSS スケジューリングクラスは、RT クラスで実行するプロセスを制御することはできません。

たとえば、4 つのプロセッサから構成されるシステムで、CPU に結合されているシングルスレッドの RT プロセスは 1 つのプロセッサを占有できます。システムが FSS も実行している場合、通常のユーザープロセスは、RT プロセスが使用していない残りの 3 つの CPU について競合します。RT プロセスは CPU を使い続けることはありません。 RT プロセスがアイドル状態になったとき、FSS は 4 つのプロセッサをすべて使用します。

次のコマンドを入力して、プロセッサセットが実行しているスケジューリングクラスを特定し、各プロセッサセットが TS、IA、FX、または FSS のプロセスのいずれかを実行するように構成されていることを確認します。

```
$ ps -ef -o pset,class | grep -v CLS | sort | uniq
1 FSS
1 SYS
2 TS
2 RT
3 FX
```

システムにデフォルトのスケジューラを設定するには、110 ページの「FSS の構成例」と `dispadmin(1M)` を参照してください。実行中のプロセスを別のスケジューリングクラスに移動するには、110 ページの「FSS の構成例」と `priocntl(1)` を参照してください。

FSS の監視

`prstat(1M)` を使用して、CPU 使用量をアクティブなプロジェクトごとに監視できます。

タスク用の拡張アカウンティングデータを使用して、長期間使用される CPU 資源の合計量について、プロジェクトごとの統計情報を取得できます。詳細は、第 7 章を参照してください。

▼ システムの CPU 使用量をプロジェクトごとに監視する方法

システム上で実行されるプロジェクトの CPU 使用量を監視するには、次のように入力します。

```
% prstat -J
```

▼ プロセッサセット内の CPU 使用量をプロジェクトごとに監視する方法

プロセッサセットに登録されているプロジェクトの CPU 使用量を監視するには、次のように入力します。

```
% prstat -J -C pset-list
```

pset-list　　コンマ区切りのプロセッサセット ID のリスト

FSS の構成例

Solaris 環境における他のスケジューリングクラスと同様に、FSS では、スケジューリングクラスを設定するコマンドや、スケジューラのチューンアップパラメータを設定するコマンド、個々のプロセスのプロパティを設定するコマンドを使用できます。

▼ スケジューリングクラスの設定方法

`dispadmin` コマンドを使用して、システムのデフォルトのスケジューラとして FSS を設定します。

```
# dispadmin -d FSS
```

この変更指定は次の再起動で有効になります。再起動後は、システムのすべてのプロセスが FSS スケジューリングクラスで実行されます。

▼ プロセスを TS から FSS クラスに手動で移動する方法

デフォルトのスケジューリングクラスを変更した後で再起動しなくても、プロセスを TS スケジューリングクラスから FSS スケジューリングクラスに手動で移動できます。

1. スーパーユーザーになります。
2. `init` プロセス **(pid 1)** を **FSS** スケジューリングクラスに移動します。

   ```
   # priocntl -s -c FSS -i pid 1
   ```

3. すべてのプロセスを **TS** スケジューリングクラスから **FSS** スケジューリングクラスに移動します。

   ```
   # priocntl -s -c FSS -i class TS
   ```

   すべてのプロセスは、再起動後には再び TS スケジューリングクラスで実行されます。

▼ プロセスをすべてのユーザークラスから FSS クラスに手動で移動する方法

TS 以外のデフォルトのクラスを使用している場合、たとえば、デフォルトで IA クラスを使用するウィンドウ環境がシステムで実行されている場合があります。デフォルトのスケジューリングクラスを変更した後で再起動しなくても、すべてのプロセスを FSS スケジューリングクラスに手動で移動できます。

1. スーパーユーザーになります。
2. `init` プロセス **(pid 1)** を **FSS** スケジューリングクラスに移動します。

   ```
   # priocntl -s -c FSS -i pid 1
   ```

3. すべてのプロセスを現在のスケジューリングクラスから **FSS** スケジューリングクラスに移動します。

   ```
   # priocntl -s -c FSS -i all
   ```

   すべてのプロセスは、再起動後には再びデフォルトのスケジューリングクラスで実行されます。

▼ プロジェクトのプロセスを FSS クラスに移動する方法

特定のプロジェクト内のプロセスを現在のスケジューリングクラスから FSS スケジューリングクラスに手動で移動できます。

1. スーパーユーザーになります。
2. プロジェクト **ID** *10* で実行するプロセスを **FSS** スケジューリングクラスに移動します。

   ```
   # priocntl -s -c FSS -i projid 10
   ```

   プロジェクトのプロセスは、再起動後には再びデフォルトのスケジューリングクラスで実行されます。

▼ スケジューラのパラメータを調整する方法

`dispadmin` コマンドを使用して、FSS スケジューラのタイムクォンタム (time quantum) 値を調べ、調整できます。タイムクォンタムとは、スレッドがプロセッサに上で実行を開始してからそのプロセッサを放棄するまでの時間量のことです。FSS スケジューラの現在のタイムクォンタムを表示するには、次のように入力します。

```
$ dispadmin -c FSS -g
#
# Fair Share Scheduler Configuration
#
RES=1000
#
# Time Quantum
#
QUANTUM=110
```

-g オプションを使用するときに、同時に -r オプションも指定すると、タイムクォンタム値の表示に使用する最小単位を指定できます。最小単位を指定しないと、タイムクォンタム値はデフォルトのミリ秒で表示されます。次のコマンドを入力します。

```
$ dispadmin -c FSS -g -r 100
#
# Fair Share Scheduler Configuration
#
RES=100
#
# Time Quantum
#
QUANTUM=11
```

FSS スケジューリングクラスにスケジューリングパラメータを設定するには、dispadmin -s を使用します。*file* 内の値は、-g オプションで得られる出力と同じ形式で指定する必要があります。これらの値は、カーネル内の現在の値を上書きします。次のコマンドを入力します。

```
$ dispadmin -c FSS -s file
```

関連項目

FSS スケジューラの使用方法については、priocntl(1)、ps(1)、dispadmin (1M)、および FSS(7) を参照してください。

第 10 章

資源プール

この章では、マシン資源のパーティション分割に使用される資源プールについて説明します。資源プールを使用すると、作業負荷によって特定の資源が重複して消費されないように、作業負荷を分離することができます。このような方法で資源を確保すると、さまざまな作業負荷が混在するシステム上で予測どおりのパフォーマンスを得ることができます。

概要

資源プールは、プロセッサセットの構成やスケジューリングクラスの割り当て (オプション) に対して一貫した構成メカニズムを提供します。

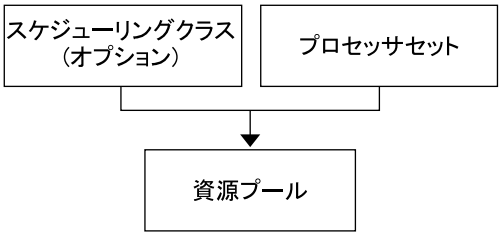


図 **10–1** 資源プールのフレームワーク

資源プールは、複数のパーティションをグループ化することにより、ラベル付けされている作業負荷とハンドルを対応付けることができます。`/etc/project` データベースにある各プロジェクトのエントリは、そのエントリに対応付けられた資源プールを持つことができます。プロジェクトで開始される新しい作業は、適切なプールに結合されます。

資源プールメカニズムは主に、5 つ以上の CPU を搭載する大規模なマシンで使用されます。ただし、小規模なマシンでもこの機能を活用することができます。小規模なマシンでは、重要でない資源パーティションを共有するプールを作成できます。 重要な資源にだけ、専用のプールが使用されます。

資源プールを使用する場合

資源プールは、以下で説明するように、多くの管理作業に適用できる汎用メカニズムを提供します。

バッチ処理サーバー

プールの機能を使用して、1 つのサーバーを 2 つのプールに分割します。

一方のプールは、ログインセッションとタイムシェアリングユーザーによる対話型作業に使用されます。もう一方のプールは、バッチシステムを介して投入されるジョブに使用されます。

アプリケーションサーバーまたはデータベースサーバー

アプリケーションの要件に基づいて、対話型アプリケーション用の資源をパーティション分割します。

アプリケーションの段階的な調整

ユーザーが期待するサービスレベルを設定します。

最初は、目標とする最終的なサービスの一部だけを実行するマシンを導入することがあります。マシンをオンラインにしたときに、予約方式の資源管理メカニズムが確立していなければ、ユーザーがサービスに不満を持つ可能性があります。

たとえば、フェアシェアスケジューラは CPU の使用率を最適化します。ユーザーから見ると、1 つしかアプリケーションを実行していないマシンの応答時間は、複数のアプリケーションがロードされているマシンの応答時間に比べ、極端に速くなります。アプリケーションごとに個別のプールを用意することにより、各アプリケーションで使用可能な CPU 数の上限をあらかじめ設定してから、すべてのアプリケーションを運用することができます。

複雑なタイムシェアリングサーバー

多数のユーザーをサポートするサーバーをパーティション分割します。

サーバーのパーティション分割によって、ユーザーごとの応答が時間をより確実に予測できる分離メカニズムが提供されます。

ユーザーをグループに分割して個別のプールに結合し、フェアシェアスケジューラ (FSS) 機能を使用すれば、CPU 割り当てを調整して、優先順位を持つユーザーグループをサポートできます。このような割り当ては、ユーザーの役割や課金などに基づいて行えます。

定期的に変動する作業負荷

資源プールを使用して、変動する作業負荷に対応します。

サイトでの作業負荷の変動が月次、四半期、年次などの周期で予想できる場合があります。 このような変動がサイトで予想できる場合は、`cron(1M)` ジョブで `pooladm` を実行して、複数のプール構成を使い分けることができます。

リアルタイムアプリケーション

RT スケジューラと専用のプロセッサ資源を使用して、リアルタイムプールを作成します。

資源プールの管理

次の表に示すコマンドは、プール機能に対する主要な管理インタフェースを提供します。

コマンド名	説明
`pooladm(1M)`	特定の構成を起動したり、現在の構成を終了したりする。オプションを指定しないで実行すると、`pooladm` は、現在実行中のプール構成を表示する
`poolbind(1M)`	プロジェクト、タスク、およびプロセスを手動でプールに結合できる

コマンド名	説明
poolcfg(1M)	プール構成ファイルの作成と変更を行う。-c オプションに info を指定して実行すると、poolcfg は現在の構成を表示する

ライブラリの API は、libpool(3LIB) で提供されます。プログラムからプール構成を操作するには、このライブラリを使用します。

プールのフレームワーク

資源プールのフレームワークは、マシンのビューを独自の構成ファイルに格納します。このファイルの格納場所は、プールのフレームワークの実装によって異なります。この構成ファイルは、プールのフレームワークにとってのマシンのビューを表します。この構成ファイルには、構成されたプールとパーティション分割可能な資源の編成に関する情報も含まれます。各プールには次のものが含まれます。

- プロセッサセットまたは CPU 資源のパーティションへの参照
- プールのデフォルトのスケジューリングクラスを示すプロパティ

システム上でのプールの実装

プールをシステム上に実装するには、次のどちらかの方法を使用します。

1. Solaris ソフトウェアが起動すると、init スクリプトは /etc/pooladm.conf ファイルが存在するかどうかをチェックします。このファイルが存在する場合は、pooladm が呼び出され、この構成をアクティブなプール構成にします。システムは、/etc/pooladm.conf で指定されている編成に従って、独自の構成ファイルを作成します。マシンの資源は指定どおりにパーティション分割されます。
2. Solaris 環境が起動しているときは、pooladm コマンドを使用して、プール構成が存在しない場合はプール構成を起動したり、プール構成を変更したりできます。デフォルトでは、pooladm は /etc/pooladm.conf の内容を使用します。ただし、別のディレクトリとファイル名を指定し、そのファイルを使用してプール構成を変更することもできます。

動的再構成の処理と資源プール

動的再構成 (DR) を使用すると、システムの実行中にハードウェアを再構成できます。DR は使用可能な資源量に影響を与えるので、プール機能を DR 操作に含めておく必要があります。DR 処理が開始されると、プールのフレームワークは構成の妥当性を検証します。

現在のプール構成が無効にならないかぎり、DR 処理は、独自の構成ファイルが更新されるまで実行を続けます。無効な構成とは、使用可能な資源でサポートできない構成のことです。

DR 処理によってプール構成が無効になった場合、操作は失敗し、メッセージログにメッセージが書き込まれます。構成処理を強制的に最後まで実行するには、DR の強制オプションを使用します。強制オプションで処理を続行すると、プール構成は、新しい資源構成に合うように変更されます。

プール構成の作成

構成ファイルには、システム上で作成されるプールに関する記述が含まれます。構成ファイルには、操作可能な構成要素と、そのリソースタイプが記述されています。

種類	説明
`pset`	プロセッサセット資源
`pool`	資源の対応付けを示す名前付きの集合
`system`	マシンレベルの実体

操作可能な構成要素については、`poolcfg(1M)` を参照してください。

構成ファイル `/etc/pooladm.conf` は、次の 2 つの方法で作成できます。

- `poolcfg` を使って現在のシステム上の資源を検出し、その結果を構成ファイルに記録します。

  この方法では、ファイルを簡単に作成できます。プール機能で操作できるシステム上のアクティブな資源とコンポーネントがすべて記録されます。資源には、既存のプロセッサセットの構成が含まれます。最後に、プロセッサセットの名前を変更したり、必要に応じてプールを作成したりして、構成を変更できます。
- `poolcfg` を使用して、新しいプール構成を作成します。

この方法は、他のマシンの構成を作成する場合や、作成済みの構成を後で現在のマシンに適用する場合に使用します。

poolcfg または libpool を使用して、/etc/pooladm.conf ファイルを変更します。このファイルを直接編集しないでください。

▼ 検出によって構成を作成する方法

/usr/sbin/poolcfg コマンドの -c オプションに discover を指定して、プール構成ファイルを作成します。作成される /etc/pooladm.conf ファイルには、既存のプロセッサセットが含まれます。

1. スーパーユーザーになります。
2. 次のコマンドを入力します。

```
# poolcfg -c discover
```

デフォルトのファイル名 /etc/pooladm.conf を使用する代わりに別のファイル名を指定することもできます。別のファイル名を指定すると、poolcfg コマンドは指定した別のファイルに対して実行されます。

たとえば、検出された構成を /tmp/foo ファイルに記録するには、次の手順に従います。

1. スーパーユーザーになります。
2. 次のコマンドを入力します。

```
# poolcfg -c discover /tmp/foo
```

▼ 新しい構成を作成する方法

/usr/sbin/poolcfg コマンドの -c オプションの引数に create を指定して、tester というシステムに簡単な構成ファイルを作成します。-c オプションの引数に空白が含まれている場合は、引用符で囲んでください。

1. スーパーユーザーになります。
2. 次のコマンドを入力します。

```
# poolcfg -c 'create system tester'
```

3. 構成ファイルの内容を読みやすい形式で表示します。

```
# poolcfg -c info
system tester
        int system.version 1
        boolean system.bind-default true
```

```
        string system.comment
```

▼ 構成の変更方法

単純な構成を拡張するには、*batch* というプロセッサセットと *batch* というプールを作成して、両者を対応付けて結合します。-c オプションの引数に空白が含まれている場合は、引用符で囲んでください。

1. スーパーユーザーになります。
2. batch というプロセッサセットを作成します。

   ```
   # poolcfg -c 'create pset batch (uint pset.min = 2; uint pset.max = 10)'
   ```

3. batch というプールを作成します。

   ```
   # poolcfg -c 'create pool batch'
   ```

4. プロセッサセットとプールを対応付けて結合します。

   ```
   # poolcfg -c 'associate pool batch (pset batch)'
   ```

5. 対応付けた後の構成を表示します。

   ```
   # poolcfg -c info
   system tester
           int system.version 1
           boolean system.bind-default true
           string system.comment

           pool batch
                   boolean pool.default false
                   boolean pool.active true
                   int pool.importance 1
                   string pool.comment
                   pset batch

           pset batch
                   int pset.sys_id -2
                   string pset.units population
                   boolean pset.default true
                   uint pset.max 10
                   uint pset.min 2
                   string pset.comment
                   boolean pset.escapable false
                   uint pset.load 0
                   uint pset.size 0
   ```

▼ プールをスケジューリングクラスに対応付ける方法

プールをスケジューリングクラスに対応付けることで、プールに結合されているすべてのプロセスがこのスケジューラを使用できるようになります。このためには、`pool.scheduler` プロパティにスケジューリングクラスの名前を設定します。次の例は、`batch` というプールを FSS に対応付ける方法を示します。

1. スーパーユーザーになります。
2. `batch` プールを変更して、**FSS** に対応付けます。

```
# poolcfg -c 'modify pool batch (string pool.scheduler="FSS")'
```

3. 対応付けた後の構成を表示します。

```
# poolcfg -c info
system tester
        int system.version 1
        boolean system.bind-default true
        string system.comment

        pool batch
                boolean pool.default false
                boolean pool.active true
                int pool.importance 1
                string pool.comment
                string pool.scheduler FSS
                pset batch

        pset batch
                int pset.sys_id -2
                string pset.units population
                boolean pset.default true
                uint pset.max 10
                uint pset.min 2
                string pset.comment
                boolean pset.escapable false
                uint pset.load 0
                uint pset.size 0
```

▼ poolcfg でコマンドファイルを使用する方法

`poolcfg -f` を使用すると、`poolcfg` コマンドの `-c` オプションに指定する引数をテキストファイルから入力できます。この手法は、一連の操作を 1 つずつ実行する場合に使用します。複数のコマンドを処理した場合でも、それらのコマンドがすべて正常に終了するまで、構成は更新されません。特に大規模な構成や複雑な構成の場合は、この手法を使用した方が、個々のサブコマンドを起動するよりも便利です。

1. 入力ファイルを作成します。

```
$ cat> poolcmds.txt
create system tester
create pset batch (int pset.man = 2; int pset.max = 10)
create pool batch
associate pool batch (pset batch)
```

2. スーパーユーザーになります。
3. 次のコマンドを入力します。

```
# /usr/sbin/poolcfg -f poolcmds.txt
```

プール構成の起動と終了

pooladm(1M) を使用して、特定のプール構成を起動したり、実行中のプール構成を終了したりします。

▼ プール構成を起動する方法

デフォルトの静的構成ファイル /etc/pooladm.conf 内の構成を起動するには、「commit configuration」を意味する -c オプションを指定して、pooladm を実行します。

1. スーパーユーザーになります。
2. 次のコマンドを入力します。

```
# /usr/sbin/pooladm -c
```

▼ プール構成を終了する方法

実行中の構成とプロセッサセットなどの関連するすべての資源を削除するには、「remove configuration」を意味する -x オプションを使用します。

1. スーパーユーザーになります。
2. 次のコマンドを入力します。

```
# /usr/sbin/pooladm -x
```

pooladm コマンドの -x オプションを使用すると、独自の動的構成ファイルだけでなく、その動的構成ファイルに対応付けられているすべての資源構成も削除されます。つまり、-x オプションは、無効なプール構成から回復するためのメカニズムを提供します。すべてのプロセスは、マシン上のすべての資源を共有します。

注 – 1 つのプロセッサセット内でスケジューリングクラスを混在させると、予期できない結果が生じる可能性があります。pooladm -x を使って無効な構成から回復する場合は、priocntl(1) を使用して、実行中のプロセスを別のスケジューリングクラスに移動する必要があります。

プールへの結合

実行中のプロセスをプールに結合するには、次の 2 つの方法を使用できます。

- poolbind(1M) コマンドを使用して、特定のプロセスを名前付き資源プールに結合する
- project(4) データベース内の project.pool 属性を使用して、新しいログインセッションや newtask(1) で起動されるタスクが結合されているプールを特定する

▼ プロセスをプールに結合する方法

次の手順では、プロセス (現在のシェルなど) を手動で *ohare* というプールに結合します。

1. スーパーユーザーになります。
2. 次のコマンドを入力します。

```
# poolbind -p ohare $$
```

▼ タスクまたはプロジェクトをプールに結合する方法

タスクまたはプロジェクトをプールに結合するには、poolbind コマンドに -i オプションを指定します。次の例では、*airmiles* プロジェクト内のすべてのプロセスを *laguardia* プールに結合します。

1. スーパーユーザーになります。
2. 次のコマンドを入力します。

```
# poolbind -i project -p laguardia airmiles
```

▼ project 属性を使って新しいプロセスをプールに結合する方法

プロジェクト内の新しいプロセスを自動的にプールに結合するには、project.pool 属性を project データベース内の各エントリに追加します。

たとえば、studio と backstage という 2 つのプールを持つ構成が存在するものとします。/etc/project ファイルの内容は、次のとおりです。

```
user.paul:1024::::project.pool=studio
user.george:1024::::project.pool=studio
user.ringo:1024::::project.pool=backstage
passes:1027::paul::project.pool=backstage
```

この構成の場合、ユーザー paul によって起動されるプロセスは、デフォルトで studio プールに結合されます。

▼ project 属性を使ってプロセスを別のプールに結合する方法

上記の構成を使用することにより、ユーザー paul は自分が起動したプロセスに結合されているプールを変更できます。newtask を使用して、passes プロジェクトで起動されるプロセスを backstage プールにも結合できます。

1. passes プロジェクトでプロジェクトを起動します。

   ```
   $ newtask -l -p passes
   ```

2. プロジェクトに結合されているプールを検証します。

   ```
   $ poolbind -q $$
   process id 6384 : pool 'backstage'
   ```

第 11 章

資源管理の構成例

この章では、資源管理のフレームワークについて考察し、仮想的なサーバー統合プロジェクトについて説明します。この例では、5 つのアプリケーションを 1 つのシステムに統合します。対象となるアプリケーションは、それぞれ資源要件、ユーザー数、およびアーキテクチャが異なります。

統合前の構成

現在、各アプリケーションは、それぞれの要件を満たす専用サーバーに置かれています。次の表にアプリケーションとその特性を示します。

アプリケーション	特性
アプリケーションサーバー	2 CPU を超えるとスケーラビリティが低い
アプリケーションサーバー用のデータベースインスタンス	負荷の高いトランザクション処理
テストおよび開発環境用のアプリケーションサーバー	GUI ベースでのコードテスト
トランザクション処理サーバー	応答時間の保証
スタンドアロンのデータベースインスタンス	大量のトランザクション処理と、複数のタイムゾーンに対するサービスの提供

統合後の構成

次の構成を使用して、アプリケーションを 1 つのシステムに統合します。

- アプリケーションサーバーは、2 つの CPU から構成されるプロセッサセットを持つ
- アプリケーションサーバーのデータベースインスタンスとスタンドアロンのデータベースインスタンスは、4 つ以上の CPU から構成される 1 つのプロセッサセットに統合される。スタンドアロンのデータベースインスタンスはその資源の 75% が保証される
- テストおよび開発用のアプリケーションサーバーには IA スケジューリングクラスを適用して、UI の応答性を保証する。メモリーを制約して、不正なコードによる影響を低減する
- トランザクション処理サーバーには 2 つ以上の CPU から構成される専用のプロセッサセットを割り当てて、応答時間を短縮する

構成の作成

`project` データベースファイルを編集します。エントリを追加して必要な資源制御を実装し、ユーザーを資源プールにマップしたら、ファイルを表示します。

```
# cat /etc/project
.
.
.
user.app_server:2001:Production Application Server:::project.pool=appserver_pool
user.app_db:2002:App Server DB:::project.pool=db_pool,project.cpu-shares(privileged,1,deny)
development:2003:Test and development::staff:project.pool=dev_pool,
  process.max-address-space=(privileged,536870912,deny)
user.tp_engine:2004:Transaction Engine:::project.pool=tp_pool
user.geo_db:2005:EDI DB:::project.pool=db_pool,project.cpu-shares=(privileged,3,deny)
.
.
.
```

注 – 開発チームはタスクを開発プロジェクトで実行する必要があります。これは、このプロジェクトへのアクセスをユーザーのグループ ID (GID) で制限しているためです。

`pool.host` という名前で入力ファイルを作成し、必要な資源プールの構成に使用します。次に、ファイルを表示します。

```
# cat pool.host

create system host
create pset default_pset (uint pset.min = 1)
create pset dev_pset (uint pset.max = 2)
create pset tp_pset (uint pset.min = 2)
create pset db_pset (uint pset.min = 4; uint pset.max = 6)
create pset app_pset (uint pset.min = 1; uint pset.max = 2)
create pool default_pool (string pool.scheduler="TS"; boolean pool.default = true)
create pool dev_pool (string pool.scheduler="IA")
create pool appserver_pool (string pool.scheduler="TS")
create pool db_pool (string pool.scheduler="FSS")
create pool tp_pool (string pool.scheduler="TS")
associate pool default_pool (pset default_pset)
associate pool dev_pool (pset dev_pset)
associate pool appserver_pool (pset app_pset)
associate pool db_pool (pset db_pset)
associate pool tp_pool (pset tp_pset)
```

次のコマンドを入力します。

```
# poolcfg -f pool.host
```

構成をアクティブにします。

```
# pooladm -c
```

システム上でフレームワークが有効になっています。

構成の表示

フレームワークの構成を表示するには、次のコマンドを入力します。

```
# pooladm
system host
        int system.version 1
        boolean system.bind-default true
        string system.comment

        pool default_pool
                boolean pool.default true
                boolean pool.active true
                int pool.importance 1
                string pool.comment
                string pool.scheduler TS
                pset default_pset

        pool dev_pool
                boolean pool.default false
                boolean pool.active true
```

```
        int pool.importance 1
        string pool.comment
        string pool.scheduler IA
        pset dev_pset

pool appserver_pool
        boolean pool.default false
        boolean pool.active true
        int pool.importance 1
        string pool.comment
        string pool.scheduler TS
        pset app_pset

pool db_pool
        boolean pool.default false
        boolean pool.active true
        int pool.importance 1
        string pool.comment
        string pool.scheduler FSS
        pset db_pset

pool tp_pool
        boolean pool.default false
        boolean pool.active true
        int pool.importance 1
        string pool.comment
        string pool.scheduler TS
        pset tp_pset

pset default_pset
        int pset.sys_id -1
        string pset.units population
        boolean pset.default true
        uint pset.max 4294967295
        uint pset.min 1
        string pset.comment
        boolean pset.escapable false
        uint pset.load 0
        uint pset.size 0

pset dev_pset
        int pset.sys_id 1
        string pset.units population
        boolean pset.default false
        uint pset.max 2
        uint pset.min 0
        string pset.comment
        boolean pset.escapable false
        uint pset.load 0
        uint pset.size 0

pset tp_pset
        int pset.sys_id 2
        string pset.units population
        boolean pset.default false
```

```
        uint pset.max 4294967295
        uint pset.min 2
        string pset.comment
        boolean pset.escapable false
        uint pset.load 0
        uint pset.size 0

pset db_pset
        int pset.sys_id 3
        string pset.units population
        boolean pset.default false
        uint pset.max 6
        uint pset.min 4
        string pset.comment
        boolean pset.escapable false
        uint pset.load 0
        uint pset.size 0

pset app_pset
        int pset.sys_id 4
        string pset.units population
        boolean pset.default false
        uint pset.max 2
        uint pset.min 1
        string pset.comment
        boolean pset.escapable false
        uint pset.load 0
        uint pset.size 0
```

フレームワークのグラフィック表示が続きます。

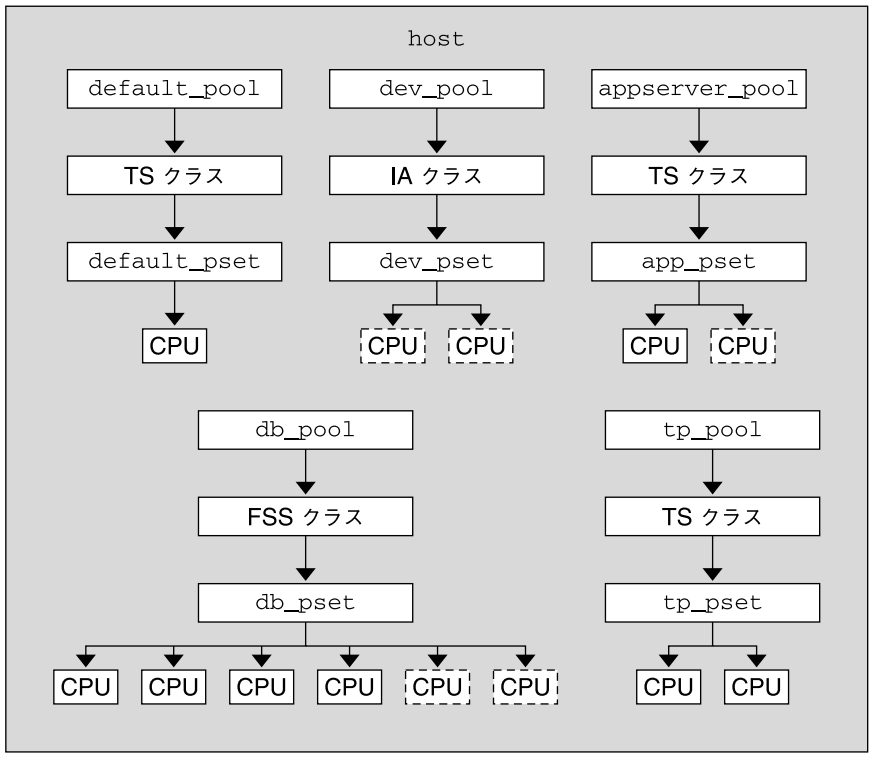


図 **11–1** サーバー統合の構成

注 – 上記の図の `db_pool` では、スタンドアロンのデータベースインスタンスに CPU 資源の 75% が保証されています。

第 12 章

Solaris 管理コンソールの資源制御機能

この章では、Solaris 管理コンソールの資源制御機能とパフォーマンス監視機能について説明します。

このコンソールでは、システムのパフォーマンスを監視したり、プロジェクト、タスク、およびプロセスに資源制御の値を設定したりします。このコンソールは、何台ものシステムに渡って設定されている数百の構成パラメータを管理する際に、コマンド行インタフェース (CLI) の代わりとして使用できる便利で安全なツールです。各システムは個別に管理されます。このコンソールのグラフィカルインタフェースは、ユーザーの経験レベルに合った使い方ができます。

Solaris 管理コンソールの使用 (作業マップ)

作業	説明	参照先
Solaris 管理コンソールを使用する	ローカル環境、ネームサービス環境、またはディレクトリサービス環境で Solaris 管理コンソールを起動する。 ネームサービス環境では、パフォーマンスツールは使用できない	『*Solaris* のシステム管理 (基本編)』の「Solaris Management Console を起動する」 および 『*Solaris* のシステム管理 (基本編)』の「ネームサービス環境で Solaris 管理ツールを使用する (作業マップ)」
システムのパフォーマンスを監視する	「System Status」の下にある「パフォーマンスツール」にアクセスする	133 ページの「パフォーマンスツールにアクセスする方法」

作業	説明	参照先
プロジェクトに資源制御機能を追加する	「System Configuration」の下にある「資源制御 (Resource Controls)」タブにアクセスする	137 ページの「「資源制御 (Resource Controls)」タブへのアクセス方法」

概要

資源管理機能は、Solaris 管理コンソールの構成要素です。このコンソールは、GUI ベースの管理ツールのためのコンテナです。管理ツールはツールボックスと呼ばれるコレクションに格納されています。コンソールとその使用方法については、『*Solaris* のシステム管理 *(*基本編*)*』の「Solaris Management Console の操作 (手順)」を参照してください。

コンソールとそのツール群のマニュアルは、コンソールのオンラインヘルプで参照できます。オンラインヘルプで参照できるマニュアルの内容については、『*Solaris* のシステム管理 *(*基本編*)*』の「Solaris Management Console (概要)」を参照してください。

管理範囲

管理範囲とは、選択した管理ツールで使用するネームサービス環境のことです。資源制御機能とパフォーマンスツールを使用するときの管理範囲は、`/etc/project` ローカルファイルまたは NIS から選択します。

コンソールセッションで選択する管理範囲は、`/etc/nsswitch.conf` ファイルで特定される最も優先度の高いネームサービスと一致する必要があります。

パフォーマンスツール

パフォーマンスツールは、資源の使用状況を監視するために使用します。資源の使用状況をシステム単位で集計したり、プロジェクト単位または個人ユーザー単位で表示したりできます。

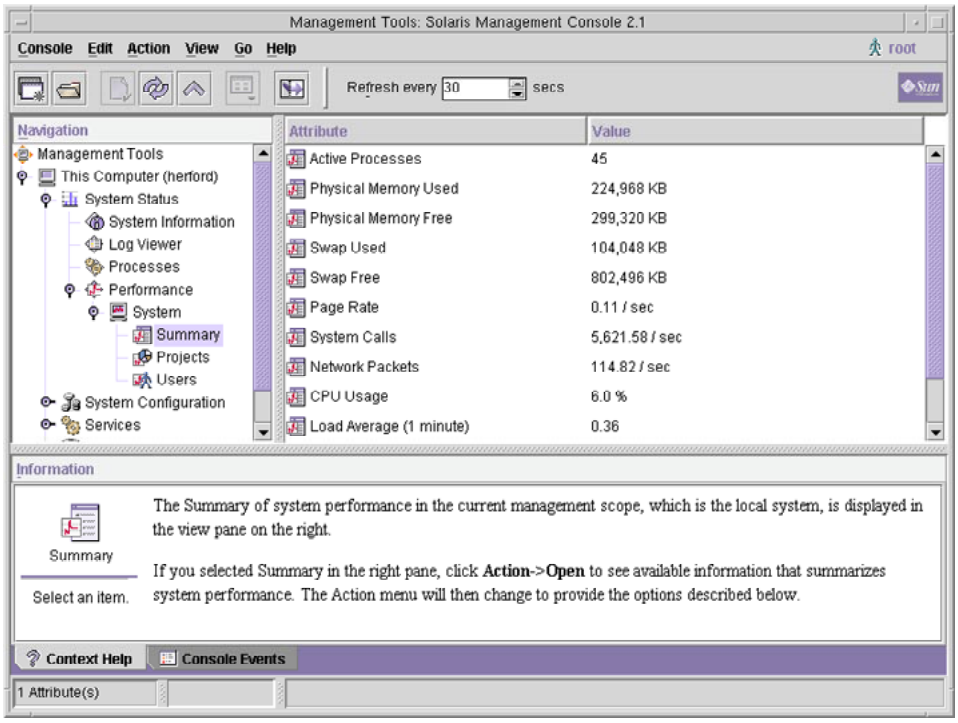


図 **12–1** Solaris 管理コンソールのパフォーマンスツール

▼ パフォーマンスツールにアクセスする方法

パフォーマンスツールは、ナビゲーション区画の「System Status」の下にあります。パフォーマンスツールにアクセスするには、次の手順に従います。

1. ナビゲーション区画の「**System Status**」コントロール要素をクリックします。
   このコントロール要素は、ナビゲーション区画のメニュー項目を拡張するために使用します。
2. 「パフォーマンス **(Performance)**」コントロール要素をクリックします。
3. 「システム **(System)**」コントロール要素をクリックします。
4. 「概要 **(Summary)**」、「プロジェクト **(Projects)**」、または「ユーザー **(Users)**」をダブルクリックします。
   何を選択するかは、監視する対象によって異なります。

システム単位の監視

次の属性の値が表示されます。

属性	説明
アクティブプロセス (Active Processes)	システム上でアクティブなプロセス数
物理メモリー使用量 (Physical Memory Used)	使用中のシステムメモリーのサイズ
物理メモリー空き容量 (Physical Memory Free)	使用可能なシステムメモリーのサイズ
スワップ使用量 (Swap Used)	使用中のシステムスワップ領域のサイズ
スワップ空き容量 (Swap Free)	使用可能なシステムスワップ領域のサイズ
ページング頻度 (Page Rate)	システムページングの頻度
システムコール (System Calls)	秒当たりのシステムコール数
ネットワークパケット (Network Packets)	秒当たりに送信されるネットワークのパケット数
CPU 使用率 (CPU Usage)	現在使用中の CPU の比率
平均負荷率 (Load Average)	過去 1 分、5 分、または 15 分の間にシステム実行キューに存在した平均プロセス数

プロジェクト単位またはユーザー単位の監視

次の属性の値が表示されます。

属性	短縮名	説明
入力ブロック (Input Blocks)	`inblk`	読み取られたブロック数
書き込まれたブロック (Blocks Written)	`oublk`	書き込まれたブロック数
読み取られた/書き込まれた文字数 (Chars Read/Written)	`ioch`	読み取りおよび書き込みが行われた文字数
データページフォルトのスリープ時間 (Data Page Fault Sleep Time)	`dftime`	データページフォルトの処理で経過した時間
強制的なコンテキストスイッチ (Involuntary Context Switches)	`ictx`	コンテキストの強制的な切り替え数

属性	短縮名	説明
システムモード時間 (System Mode Time)	`stime`	カーネルモードで経過した時間
メジャーページフォルト (Major Page Faults)	`majfl`	メジャーページフォルト数
受信したメッセージ (Messages Received)	`mrcv`	受信されたメッセージ数
送信したメッセージ (Messages Sent)	`msend`	送信されたメッセージ数
マイナーページフォルト (Minor Page Faults)	`minf`	マイナーページフォルトの数
プロセス数 (Num Processes)	`nprocs`	ユーザーまたはプロジェクトが所有するプロセス数
LWP 数 (Num LWPs)	`count`	軽量プロセスの数
その他のスリープ時間 (Other Sleep Time)	`slptime`	`tftime`、`dftime`、`kftime`、 および `ltime` を除いたスリープ時間
CPU 時間 (CPU Time)	`pctcpu`	プロセス、ユーザー、またはプロジェクトが使用した最新の CPU 時間の比率
使用メモリー (Memory Used)	`pctmem`	プロセス、ユーザー、またはプロジェクトが使用したシステムメモリーの比率
ヒープサイズ (Heap Size)	`brksize`	プロセスのデータセグメントに割り当てられているメモリーサイズ
常駐サイズ (Resident Set Size)	`rsssize`	プロセスによって要求されている現在のメモリーサイズ
プロセスイメージサイズ (Process Image Size)	`size`	プロセスイメージのサイズ (K バイト)
受信したシグナル (Signals Received)	`sigs`	受信されたシグナルの数
停止時間 (Stopped Time)	`stoptime`	停止状態で経過した時間
スワップ操作 (Swap Operations)	`swaps`	進行中のスワップ操作の数
実行されたシステムコール (System Calls Made)	`sysc`	設定された (直前の) 時間間隔で実行されたシステムコール数

属性	短縮名	説明
システムページフォルトのスリープ時間 (System Page Fault Sleep Time)	kftime	ページフォルトの処理で経過した時間
システムトラップ時間 (System Trap Time)	ttime	システムトラップの処理で経過した時間
テキストページフォルトのスリープ時間 (Text Page Fault Sleep Time)	tftime	テキストページフォルトの処理で経過した時間
ユーザーロック待機のスリープ時間 (User Lock Wait Sleep Time)	ltime	ユーザーロックを待機している間に経過した時間
ユーザーモード時間 (User Mode Time)	utime	ユーザーモードで経過した時間
ユーザーおよびシステムモード時間 (User and System Mode Time)	time	CPU 実行の累積時間
任意コンテキストスイッチ (Voluntary Context Switches)	vctx	コンテキストの自主的な切り替え数
待機 CPU 時間 (Wait CPU Time)	wtime	CPU を待機している間に経過した時間 (応答時間)

「資源制御 (Resource Controls)」タブ

資源制御を使用して、プロジェクトを資源制約の集合と対応付けることができます。これらの制約によって、プロジェクトのコンテキストで実行するタスクまたはプロセスの資源許容量が決定されます。

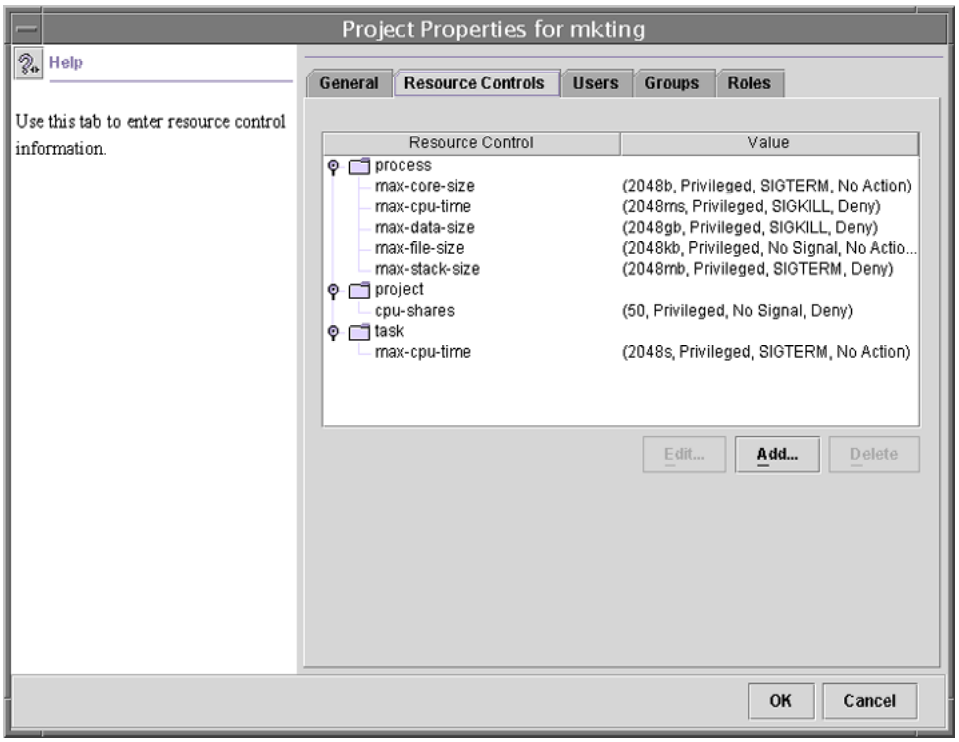


図 **12–2** Solaris 管理コンソールの「資源制御 (Resource Controls)」タブ

▼ 「資源制御 (Resource Controls)」タブへのアクセス方法

「資源制御 (Resource Controls)」タブは、ナビゲーション区画の「System Configuration」の下にあります。 「資源制御 (Resource Controls)」にアクセスするには、次の手順に従います。

1. ナビゲーション区画の「**System Configuration**」コントロール要素をクリックします。
2. 「プロジェクト **(Projects)**」をダブルクリックします。
3. コンソールのメインウィンドウにあるプロジェクトをクリックして選択します。
4. 「アクション **(Action)**」メニューから「プロパティ **(Properties)**」を選択します。
5. 「資源制御 **(Resource Controls)**」タブをクリックします。
   プロセス、プロジェクト、およびタスクの資源制御の値を表示、編集、または削除します。

設定可能な資源制御

使用可能な資源制御のリストを表示するには、コンソールヘルプの「資源制御」または 90 ページの「使用可能な資源制御」を参照してください。

値の設定

プロセス、プロジェクト、およびタスクの資源制御の値を表示、編集、または削除できます。 これらの操作は、コンソールのダイアログボックスで実行します。

資源制御 (Resource Control) と値 (Value) は、コンソールに表形式で表示されます。資源制御 (Resource Control) の欄には、設定可能な資源制御の一覧が表示されます。値 (Value) の欄には、各資源制御に対応付けられているプロパティが表示されます。表内では、これらの値は括弧で囲まれており、コンマで区切られたプレーンテキストとして表示されます。括弧内の値は「action 文節」を示します。各 action 文節は、しきい値、特権レベル、シグナル、および特定のしきい値に対応付けられているローカルアクションで構成されます。 各資源制御は複数の action 文節を持つことができ、各 action 文節もコンマで区切られます。

注 – 実行中のシステムでは、コンソールから `project` データベースに加えた変更は、プロジェクトで起動される新しいタスクに対してだけ有効になります。

関連項目

プロジェクトとタスクについては、第 6 章を参照してください。資源制御については、第 8 章を参照してください。フェアシェアスケジューラ (FSS) については、第 9 章を参照してください。

第 13 章

リモートファイルシステムへのアクセス (トピック)

第 14 章	NFS サービスの概要
第 15 章	NFS サービスの設定と障害回避方法 (トラブルシューティング)
第 16 章	NFS サービスの背景情報について

第 14 章

ネットワークファイルシステムの管理 (概要)

この章では、ネットワーク経由でファイルシステムにアクセスするために使用する NFS サービスの概要を説明します。また、NFS サービスを理解するために必要な概念、および NFS と autofs の最新の機能についても説明します。

NFS の用語

ここでは、NFS サービスを使用するために必要な基本用語について説明します。NFS サービスの詳細は、第 16 章で説明します。

NFS サーバーとクライアント

クライアントとサーバーという用語は、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他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ーと呼びます。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他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になります。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ー、またはその両方の役割として動作することができます。

クライアントは、サーバー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ー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。マウントプロセ

スでは一連のリモート手続き呼び出しによって、クライアントからサーバー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、ユーザー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。ファイルシステムをマウントする方法については、155 ページの「ファイルシステムのマウント」 を参照してください。

サーバー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。`share` コマンドと autofs に関連する作業については、150 ページの「ファイルシステムの自動共有」 と 167 ページの「autofs 管理作業の概要」 を参照してください。

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ファイル階層と重複するファイル階層は共有できません。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ますが、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。したがってこのマニュアルでファイルシステムという語を使用する場合、NFS 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 サービスについて

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。このモデルにより、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 サービスの利点は、次のとおりです。

- 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

- 各ユーザーアプリケーションがローカルのディスクスペース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
- すべてのユーザー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ー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
- さまざま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ーとは無関係に決めることができます。ユーザー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、リモートファイルシステムとローカルファイルシステムの区別がありません。

autofs について

NFS サービスで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` という常駐型の自動マウントデーモンが、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ー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。

ブート時にマウントする必要はなく、ユーザーはディレクトリをマウントするためにスーパーユーザーのパスワードを知る必要はありません。ユーザーが `mount` と `umount` コマンドを使用する必要もありません。autofs は、ユーザーの介入なしに、必要に応じてファイルシステムをマウントまたは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` によって他の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` ファイルを使用して / (ルート)、`/usr`、および `/usr/kvm` をマウントしなければなりません。

autofs サービスについては、167 ページの「autofs 管理作業の概要」と 241 ページの「autofs のしくみ」で詳しく説明します。

NFS サービスの機能

ここでは、NFS サービスの重要な機能について説明します。

NFS バージョン 2 プロトコル

バージョン 2 は、一般に広く使用された初めての 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 のすべてのリリースが NFS プロトコルのバージョン 2 をサポートし、Solaris 2.5 より以前のリリースはバージョン 2 だけをサポートします。

NFS バージョン 3 プロトコル

NFS バージョン 3 のプロトコルは、Solaris 2.5 で新機能として追加されたものです。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に利用するには、NFS サーバー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、サーバーで非同期の書き込みが可能になります。サーバーがクライアントの書き込み要求をメモリーに保存するので、効率が向上しました。クライアントは、サーバー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。サーバーは要求をバッチ処理することもできるので、サーバー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。したがってサーバーに対する 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、ユーザー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」というメッセージが出力されます。

NFS バージョン 3 プロトコルでは、8K 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ーは、バージョン 2 の 8K バイトの制限を受けることなく、サポートされている転送サイズをネゴシエートします。Solaris 2.5 から、転送サイズが 32K 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 で、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 よりも厳密に設定します。この追加機能では効率は改善されませんが、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。ACL の詳細は、『*Solaris* のシステム管理 *(*セキュリティサービス*)*』の「アクセス制御リスト (ACL)」を参照してください。

NFS の TCP への依存

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は、Solaris 2.5 で TCP (Transmission Control Protocol) に変更されました。TCP は、低速ネットワークとワイドエリアネットワークのパフォーマンスの向上に役立ちます。TCP には、トラフィック抑制機能とエラー回復機能もあります。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。Solaris 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と NFS

Solaris 2.5 から、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 のファイル共有を使用できます。NFS ファイルのロックメカニズムの信頼性の向上により、ロックを使用するコマンドのハングが起こりにくくなりました。

NFS 大規模ファイルのサポート

Solaris 2.6 の NFS バージョン 3 プロトコルから、2G バイトを超えるサイズのファイル (大規模ファイル) も正しく処理できるようになりました。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 2G 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 クライアントのフェイルオーバー機能

Solaris 2.6 では、読み取り専用ファイルシステムの動的フェイルオーバー機能が追加されました。フェイルオーバーによって、マニュアルページ、その他のドキュメント、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。フェイルオーバー機能は、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、フェイルオーバー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。詳細は、158 ページの「クライアント側フェイルオーバーを使用する方法」 と 227 ページの「クライアント側フェイルオーバー機能」 を参照してください。

NFS サービスのための Kerberos 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` と `share` コマンドが Kerberos V5 認証を使用する NFS バージョン 3 のマウントをサポートするように変更されました。`share` コマンドもクライアントごとに異なる複数の認証機能を使用できるように変更されました。各種のセキュリティ機能の詳細は、146 ページの「RPCSEC_GSS セキュリティ方式」 を参照してください。Kerberos V5 認証の詳細は、『*Solaris* のシステム管理 (セキュリティサービス)』の「SEAM NFS サーバーの構成」を参照してください。

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追加されました。インターネットアクセスに WebNFS™ プロトコルを使用する利点の 1 つは、信頼性が高いことです。このサービスは、NFS バージョン 3 とバージョン 2 プロトコルの拡張として構築されています。また NFS サーバーでは、負荷が大きい状態のときに HTTP (HyperText Transfer Protocol) から Web サーバーへのアクセスよりも高いスループットを確保できます。このため、ファイルを取得するために必要な時間を短縮できます。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。WebNFS サービスのその他の変更については、147 ページの「WebNFS サービスのセキュリティネゴシエーション」 を参照してください。作業については、164 ページの「WebNFS の管理作業」 を参照してください。

RPCSEC_GSS セキュリティ方式

Solaris 7 から、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メカニズムをサポートしています。Kerberos V5 認証のサポートについての詳細は、146 ページの「NFS サービスのための Kerberos のサポート」 を参照してください。GSS-API についての詳細は、『*GSS-API* のプログラミング』を参照してください。

Solaris 7 の NFS に対する拡張機能

Solaris 7 で、`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。MOUNT プロトコルは、WebNFS サービスが使用するアクセス方法と同じです。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、サーバー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。WebNFS サービスの変更の詳細は、146 ページの「WebNFS のサポート」 を参照してください。

WebNFS サービスのセキュリティネゴシエーション

Solaris 8 で、WebNFS クライアントが NFS サーバーとセキュリティメカニズムをネゴシエートするための新しいプロトコルが追加されました。このプロトコルの追加により、WebNFS サービスの使用時に、セキュリティ保護されたトランザクションを使用できます。詳細については、230 ページの「WebNFS セキュリティネゴシエーション機能のしくみ」 を参照してください。

NFS サーバーロギング

Solaris 8 で、NFS サーバーはサーバーログ機能によって、ファイルシステムに実行されたファイル操作の記録を提供できるようになりました。このログには、アクセスされた対象、時間、アクセスした人を追跡するための情報が含まれています。一連の構成オプションを使用して、これらの情報を含むログの場所を指定することができます。また、これらのオプションを使用して、ログに記録する処理を選択することもできます。この機能は、NFS クライアントや WebNFS クライアントで匿名 ftp を利用するサイトで特に便利です。詳細は、153 ページの「NFS サーバーログを有効にする方法」 を参照してください。

autofs の特徴

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+、およびローカルファイルに保存されます。

Solaris 2.6 から、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、サーバー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。Solaris 2.6 より前のリリースでは、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。現在は、いちばん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他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。これによりユーザーは、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 を使用することによって、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。詳細は、180 ページの「autofs のブラウズ機能を無効にする」 を参照してください。

第 15 章

リモートファイルシステムの管理 (手順)

この章では、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など、NFS の管理作業の実行方法について説明します。また、Secure NFS システムおよび WebNFS の機能の使用方法についても説明します。章の最後では障害追跡の手順を説明し、NFS のいくつかのエラーメッセージとその意味を示します。

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。

- サーバー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
- サーバー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ーの保守には、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したりすること
- 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、サーバー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、サーバー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 サーバー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。たとえばサーバーがホームディレクトリをサポートしている場合、ホーム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、テストまたは障害追跡の場合だけです。

`dfstab` ファイルには、サーバーがクライアントと共有しているすべてのファイルシステムがリストされています。このファイルを使用して、ファイルシステムをマウントできるクライアントを制御します。`dfstab` ファイルを変更して、ファイルシステムを追加または削除したり、共有方法を変更したりできます。その場合は、`vi` などのサポートされているテキストエディタを使って `dfstab` ファイルを編集します。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。

表 **15–1** ファイルシステムの共有 (作業マップ)

作業	説明	参照先
自動ファイルシステムの共有を確立する	サーバーのリブート時、ファイルシステムが自動的に共有されるようにサーバーを設定する手順	151 ページの「ファイルシステム自動共有を設定する方法」
WebNFS を有効にする	ユーザーが WebNFS でファイルにアクセスできるようにサーバーを設定する手順	152 ページの「WebNFS アクセスを有効にする方法」
NFS サーバーログを有効にする	NFS ログが選択したファイルシステム上で動作するようにサーバーを設定する手順	153 ページの「NFS サーバーログを有効にする方法」

▼ ファイルシステム自動共有を設定する方法

1. スーパーユーザー、またはそれと同等の役割になります。

   役割については、『*Solaris* のシステム管理 *(*セキュリティサービス*)*』の「特権付きアプリケーションの使用」を参照してください。

2. 共有する対象の各ファイルシステムに関してエントリを追加します。

   /etc/dfs/dfstab を編集し、自動的に共有する各ファイルシステムのファイルにエントリを 1 つ追加します。各エントリは、ファイル中に 1 行で記述する必要があり、次のような構文を使用します。

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

   /etc/dfs/dfstab ファイルについては、dfstab(4) のマニュアルページを、全オプションのリストについては、share_nfs(1M) のマニュアルページを参照してください。

3. **NFS** サービスがサーバー上で動作していることを確認します。

   share コマンドまたは share コマンドセットをはじめて実行する場合、NFS サービスが動作していないことがあります。次のコマンドを使用して、NFS デーモンのどれかが動作していることをチェックします。

   ```
   # pgrep nfsd
   318
   ```

   この例では、318 は nfsd のプロセス ID です。 ID が表示されない場合は、サービスが動作していないことを意味します。 次に、mountd が動作していることをチェックします。

4. **(**省略可能**) NFS** サービスを起動します。

   前の手順を実行しても nfsd のプロセス ID が表示されない場合は、次のコマンドを使用して、NFS サービスを起動します。

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

   このコマンドを実行すると、NFS サービスがサーバーで実行されます。リブート時にサーバーが実行レベル 3 であるときには、NFS サービスが自動的に再起動されます。

5. **(**省略可能**)** ファイルシステムを共有します。

   エントリを /etc/dfs/dfstab に追加したあと、システムをリブートするか、shareall コマンドを使用して、ファイルシステムを共有可能にできます。NFS サービスを起動したすぐ後は、このコマンドはスクリプトにより実行されているため、実行する必要はありません。

   ```
   # shareall
   ```

6. 情報が正しいことを確認します。

   share コマンドを実行し、適切なオプションが表示されていることを確認します。

   ```
   # share
   -        /export/share/man   ro   ""
   -        /usr/src     rw=eng   ""
   ```

```
-        /export/ftp     ro,public  ""
```

次に進む手順

次の手順は、サーバー上で共有化した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 autofs マップを設定する手順です。167 ページの「autofs 管理作業の概要」を参照してください。

▼ WebNFS アクセスを有効にする方法

Solaris 2.6 から、デフォルトでは、NFS のマウントが利用可能なファイルシステムはすべて、WebNFS アクセスも自動的に利用できます。この手順を使用する必要があるのは、次のいずれかの場合だけです。

- サーバー上での NFS マウントを許可しても、WebNFS アクセスが許可されない場合
- `public` オプションを使用して、公共ファイルハンドルをリセットし NFS URL を短くする場合
- `index` オプションを使用して、特定の html ファイルを強制的に読み込む場合

WebNFS サービスを起動する際の注意事項については、165 ページの「WebNFS アクセスの計画」 を参照してください。

1. スーパーユーザー、またはそれと同等の役割になります。

   役割については、『*Solaris* のシステム管理 *(*セキュリティサービス*)*』の「特権付きアプリケーションの使用」を参照してください。

2. **WebNFS** サービスを使用して、共有する各ファイルシステムのエントリを追加します。

   `/etc/dfs/dfstab` を編集して、各ファイルシステムのファイルにエントリを 1 つ追加します。次の例の `public` タグおよび `index` タグは省略できます。

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

   `/etc/dfs/dfstab` ファイルについては、`dfstab(4)` のマニュアルページを、全オプションのリストについては、`share_nfs(1M)` のマニュアルページを参照してください。

3. **NFS** サービスがサーバー上で動作していることを確認します。

   `share` コマンドまたは `share` コマンドセットをはじめて実行する場合、NFS デーモンが動作していないことがあります。次のコマンドを使用して、いずれかの NFS デーモンが動作していることをチェックします。

   ```
   # pgrep nfsd
   318
   ```

   この例では、`318` は `nfsd` のプロセス ID です。 ID が表示されない場合は、サービスが動作していないことを意味します。 次に、`mountd` が動作していることをチェックします。

4. **(省略可能) NFS** サービスを起動します。

   前の手順を実行しても `nfsd` のプロセス ID が表示されない場合は、次のコマンドを使用して、NFS サービスを起動します。

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

   このコマンドを実行すると、NFS サービスがサーバーで実行されます。ブート時にサーバーが実行レベル 3 になったときには、NFS サービスが自動的に再起動されます。

5. **(省略可能)** ファイルシステムを共有します。

   エントリを `/etc/dfs/dfstab` に追加したあと、システムをリブートするか、`shareall` コマンドを使用して、ファイルシステムを共有可能にできます。NFS サービスを起動したすぐ後は、このコマンドはスクリプトにより実行されているため、実行する必要はありません。

   ```
   # shareall
   ```

6. 情報が正しいことを確認します。

   `share` コマンドを実行し、適切なオプションが表示されていることを確認します。

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

▼ NFS サーバーログを有効にする方法

1. スーパーユーザー、またはそれと同等の役割になります。

   役割については、『*Solaris* のシステム管理 *(*セキュリティサービス*)*』の「特権付きアプリケーションの使用」を参照してください。

2. **(省略可能)** ファイルシステム構成の設定を変更します。

   `/etc/nfs/nfslog.conf` で、`global` タグに関連するデータを変更してすべてのファイルシステムについてデフォルトの設定を編集するか、このファイルシステムに新しいタグを追加することができます。これらの変更が必要でない場合には、このファイルを変更する必要はありません。`/etc/nfs/nfslog.conf` の書式については、`nfslog.conf(4)` を参照してください。

3. **NFS** サーバーログを使用して、共有する各ファイルシステムについてエントリを追加します。

   `/etc/dfs/dfstab` を編集し、NFS サーバー記録を有効にするファイルシステムについてエントリを 1 つ追加します。`log=`*tag* オプションとともに使用するタグは、`/etc/nfs/nfslog.conf` にも記述する必要があります。次の例では、`global` タグ内のデフォルト設定を使用しています。

   ```
   share -F nfs -o ro,log=global /export/ftp
   ```

   `/etc/dfs/dfstab` ファイルについては、`dfstab(4)` のマニュアルページを、全オプションのリストについては、`share_nfs(1M)` のマニュアルページを参照してくだ

さい。

4. **NFS** サービスがサーバー上で動作していることを確認します。

   share コマンドまたは share コマンドセットをはじめて実行する場合、NFS デーモンが動作していないことがあります。次のコマンドを使用して、NFS デーモンのどれかが動作していることをチェックします。

   ```
   # pgrep nfsd
   318
   ```

   この例では、318 は nfsd のプロセス ID です。 ID が表示されない場合は、サービスが動作していないことを意味します。 次に、mountd が動作していることをチェックします。

5. **(**省略可能**) NFS** サービスを起動します。

   前の手順を実行しても nfsd のプロセス ID が表示されない場合は、次のコマンドを使用して、NFS サービスを起動します。

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

   このコマンドを実行すると、NFS サービスがサーバーで実行されます。ブート時にサーバーが実行レベル 3 になったときには、NFS サービスが自動的に再起動されます。

6. **(**省略可能**)** ファイルシステムを共有します。

   エントリを /etc/dfs/dfstab に追加したあと、システムをリブートするか、shareall コマンドを使用して、ファイルシステムを共有可能にできます。NFS サービスを起動したすぐ後は、このコマンドはスクリプトにより実行されているため、実行する必要はありません。

   ```
   # shareall
   ```

7. 情報が正しいことを確認します。

   share コマンドを実行し、適切なオプションが表示されていることを確認します。

   ```
   # share
   -       /export/share/man   ro   ""
   -       /usr/src     rw=eng   ""
   -       /export/ftp    ro,log=global  ""
   ```

8. **(**省略可能**) NFS** ログデーモン nfslogd がすでに実行されていなければ、起動します。

   nfs.server スクリプトを使用して NFS デーモンの再起動をすると、nfslog.conf ファイルが存在している場合、デーモンが起動されます。それ以外の場合は、サーバーのリブート時にコマンドが自動的に再起動されるように、一度手動でコマンドを実行してファイルを作成する必要があります。

   ```
   # /usr/lib/nfs/nfslogd
   ```

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。システムを起動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、状況によってこの 3 つの方法を組み合わせる必要があります。このような 3 つのファイルシステムのマウント方法に加え、ファイルシステムのマウント時に使用するオプションに応じて、プロセスを有効または無効にする方法がいくつかあります。ファイルシステムのマウントに関するすべての作業のリストについては、次の表を参照してください。

表 **15–2** ファイルシステムのマウント (作業マップ)

作業	説明	参照先
ブート時にファイルシステムをマウントする	システムがリブートされるときに必ずファイルシステムがマウントされるようにする手順	156 ページの「ブート時のファイルシステムのマウント方法」
コマンドを使用してファイルシステムをマウントする	システムの動作時にファイルシステムをマウントする手順。この手順はテストに有効	156 ページの「コマンド行からファイルシステムをマウントする方法」
オートマウンタによりマウントする	コマンド行を使用せずに、要求に応じてファイルシステムにアクセスする手順	157 ページの「オートマウンタによるマウント」
大規模ファイルを避ける	ファイルシステム上に大規模ファイルが作成されないようにする手順	157 ページの「NFS サーバー上で大規模ファイルを無効にする方法」
クライアント側フェイルオーバーを開始する	サーバーの不良時、動作中のファイルシステムへの自動切り換えを有効にする手順	158 ページの「クライアント側フェイルオーバーを使用する方法」
クライアントに対するマウントアクセスを無効にする	任意のクライアントがリモートシステムにアクセスする機能を無効にする手順	159 ページの「1 つのクライアントに対するマウントのアクセスを無効にする方法」
ファイアウォールを越えてファイルシステムへのアクセスを提供する	WebNFS プロトコルを使用してファイアウォールを越えてファイルシステムへのアクセスを許可する手順	159 ページの「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」
NFS URL を使用してファイルシステムをマウントする	NFS URL を使用してファイルシステムへのアクセスを許可する手順。このプロセスによって、MOUNT プロトコルを使用しないでファイルシステムへのアクセスが可能になる	160 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」

▼ ブート時のファイルシステムのマウント方法

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に実行されるため、この手順をリモートファイルシステムで使用しないでください。

1. スーパーユーザー、またはそれと同等の役割になります。

   役割については、『*Solaris* のシステム管理 *(*セキュリティサービス*)*』の「特権付きアプリケーションの使用」を参照してください。

2. ファイルシステムに関するエントリを `/etc/vfstab` に追加します。

   `/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
   ```

   詳細は、`vfstab(4)` のマニュアルページを参照してください。

注意 – NFS サーバーに 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、作成しないでください。`/etc/vfstab` のエントリが確認されたのち、NFS サービスが起動します。その結果、互いにファイルシステムをマウントしている 2 つのサーバーが同時に停止した場合、それらのシステムはリブート時にハングアップすることがあります。

例 - `vfstab` エントリ

`wasp` サーバーの `/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。それには、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
```

▼ コマンド行からファイルシステムをマウントする方法

新規マウントポイントをテストするために、コマンド行からファイルシステムをマウントすることがあります。このようにしてマウントすると、オートマウンタでアクセスできないファイルシステムに、一時的にアクセスすることができます。

1. スーパーユーザー、またはそれと同等の役割になります。

   役割については、『*Solaris* のシステム管理 *(*セキュリティサービス*)*』の「特権付きアプリケーションの使用」を参照してください。

2. ファイルシステムをマウントします。
   次のコマンドを入力します。

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

   上の例では、bee サーバーの /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。

オートマウンタによるマウント

167 ページの「autofs 管理作業の概要」では、オートマウンタによるマウントの確立とサポートについて詳細に説明します。通常のシステムに変更を加えることなく、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、次の内容を入力するだけです。

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

オートマウンタでは、すべてのユーザー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 サーバー上で大規模ファイルを無効にする方法

2G バイトを超えるファイルを処理できないクライアントをサポートしているサーバーについては、大規模ファイルを作成する機能を無効にしておく必要があります。

注 – Solaris 2.6 より前の動作環境では、大規模ファイルは使用できません。クライアントが大規模ファイルにアクセスする必要がある場合には、NFS サーバーのクライアントが Solaris 2.6 以降のリリースで動作していることを確認してください。

1. スーパーユーザー、またはそれと同等の役割になります。

役割については、『*Solaris* のシステム管理 *(*セキュリティサービス*)*』の「特権付きアプリケーションの使用」を参照してください。

2. ファイルシステム上に大規模ファイルが存在していないことを確認してください。

   次の例は、大規模ファイルを検索するためのコマンドです。

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

   システム上に大規模ファイルが存在する場合には、削除するか、他のファイルシステムに移動する必要があります。

3. ファイルシステムをアンマウントします。

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

4. ファイルシステムがマウントされている場合は、`largefiles` を使ってファイルシステムをリセットします。

   `fsck` は、ファイルシステム上に大規模ファイルが存在しない場合に、ファイルシステムの状態をリセットします。

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

5. `nolargefiles` を使用して、ファイルシステムをマウントします。

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

   コマンド行からマウントすることができますが、オプションをさらに固定的にするには、`/etc/vfstab` に次のようなエントリを追加してください。

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

▼ クライアント側フェイルオーバーを使用する方法

1. スーパーユーザー、またはそれと同等の役割になります。

   役割については、『*Solaris* のシステム管理 *(*セキュリティサービス*)*』の「特権付きアプリケーションの使用」を参照してください。

2. **NFS** クライアント上で、`ro` オプションを使用してファイルシステムをマウントします。

   コマンド行からも、オートマウンタを使用しても、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マウントできます。

   ```
   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フェイルオーバー機能は使用できなかったため、サーバー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ーを、コマンド行や vfstab のエントリに混在させないでください。NFS バージョン 2 プロトコルとバージョン 3 プロトコルをサポートしているサーバーを混在して使用できるのは、autofs を使用する場合だけです。autofs では、バージョン 2 またはバージョン 3 のサーバーの最適なサブセットが使用されます。

▼ 1 つのクライアントに対するマウントのアクセスを無効にする方法

1. スーパーユーザー、またはそれと同等の役割になります。

   役割については、『*Solaris* のシステム管理 *(*セキュリティサービス*)*』の「特権付きアプリケーションの使用」を参照してください。

2. /etc/dfs/dfstab にエントリを追加します。

   最初の例では、rose という名称のホストを除き、eng ネットグループ内のすべてのクライアントへのマウントアクセスを許可しています。2 つ目の例では、rose を除き、eng.sun.com DNS ドメイン内にあるすべてのクライアントへのマウントアクセスを許可しています。

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

   アクセスリストに関する補足的な情報については、214 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。/etc/dfs/dfstab については、dfstab(4) のマニュアルページを参照してください。

3. ファイルシステムを共有します。

   /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ーがリブートされるまでは NFS サーバーに反映されません。

   ```
   # 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. スーパーユーザー、またはそれと同等の役割になります。

   役割については、『*Solaris* のシステム管理 *(*セキュリティサービス*)*』の「特権付きアプリケーションの使用」を参照してください。

2. 次のコマンドを使用して、ファイルシステムを手動でマウントします。

   ```
   # 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 サーバー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 – この手順では、NFS サーバーのファイルシステムを public オプションで共有する必要があります。また、クライアントとサーバー間のファイアウォールでは、ポート 2049 で TCP 接続できるようにする必要があります。Solaris 2.6 から、共有しているすべてのファイルシステムに、公共ファイルハンドルでアクセスできます。そのため、デフォルトでは、public オプションが適用されています。

▼ 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. スーパーユーザー、またはそれと同等の役割になります。

   役割については、『*Solaris* のシステム管理 *(*セキュリティサービス*)*』の「特権付きアプリケーションの使用」を参照してください。

2. 次のコマンドを使用して、ファイルシステムを手動でマウントします。

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

   この例では、bee というサーバーの /export/share/local というファイルシステムが、NFS ポート番号 3000 を使っ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、public オプションを含めるかどうかを選択できます。public オプションを指定しない場合、サーバー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。

NFS サービスの設定

この節では、NFS サービスの初期化や使用に必要な作業をいくつか説明します。

表 **15–3** NFS サービス (作業マップ)

作業	説明	参照先
NFS サーバーを起動する	NFS サービスが自動的に起動されていない場合に、NFS サービスを起動する手順	161 ページの「NFS サービスの起動方法」

表 **15–3** NFS サービス (作業マップ)　(続き)

作業	説明	参照先
NFS サーバーを停止する	NFS サービスを停止する手順。通常は、サービスを停止する必要はない	161 ページの「NFS サービスの停止方法」
オートマウンタを起動する	オートマウンタを起動する手順。オートマウンタマップが変更された場合、この手順が必要	161 ページの「オートマウンタの起動方法」
オートマウンタを停止する	オートマウンタを停止する手順。オートマウンタマップが変更された場合、この手順が必要	162 ページの「オートマウンタの停止方法」

▼ NFS サービスの起動方法

1. スーパーユーザー、またはそれと同等の役割になります。
   役割については、『*Solaris* のシステム管理 *(*セキュリティサービス*)*』の「特権付きアプリケーションの使用」を参照してください。
2. **NFS** サービスデーモンを有効にします。
   次のコマンドを入力します。

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

   `/etc/dfs/dfstab` にエントリがあると、このコマンドはデーモンを起動します。

▼ NFS サービスの停止方法

1. スーパーユーザー、またはそれと同等の役割になります。
   役割については、『*Solaris* のシステム管理 *(*セキュリティサービス*)*』の「特権付きアプリケーションの使用」を参照してください。
2. **NFS** サービスデーモンを無効にします。
   次のコマンドを入力します。

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

▼ オートマウンタの起動方法

1. スーパーユーザー、またはそれと同等の役割になります。
   役割については、『*Solaris* のシステム管理 *(*セキュリティサービス*)*』の「特権付きアプリケーションの使用」を参照してください。
2. **autofs** デーモンを有効にします。

次のコマンドを入力します。

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

▼ オートマウンタの停止方法

1. スーパーユーザー、またはそれと同等の役割になります。
   役割については、『*Solaris* のシステム管理 *(*セキュリティサービス*)*』の「特権付きアプリケーションの使用」を参照してください。
2. **autofs** デーモンを無効にします。
   次のコマンドを入力します。

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

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」とは管理上のエンティティであり、通常、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。ネームサービスを実行している場合、そのドメインに対してネームサービスを設定しなければなりません。『*Solaris* のシステム管理 *(*ネーミングとディレクトリサービス *: FNS*、*NIS+* 編*)*』を参照してください。

Diffie-Hellman 認証を使用するように、Secure NFS 環境を設定できます。この認証サービスについては、『*Solaris* のシステム管理 *(*セキュリティサービス*)*』の「認証サービスの使用 (手順)」で説明しています。

NFS サービスでは、Kerberos バージョン 5 認証もサポートされています。Kerberos サービスについては、『*Solaris* のシステム管理 *(*セキュリティサービス*)*』の「SEAM について」で説明しています。

▼ DH 認証を使用して 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。
   NIS+ をネームサービスとして使用している場合は、『*Solaris* のシステム管理 *(*ネーミングとディレクトリサービス *: FNS*、*NIS+* 編*)*』を参照してください。
2. `newkey` コマンドまたは `nisaddcred` コマンドを使用して、クライアントのユーザーの公開鍵と秘密鍵を設定します。`chkey` コマンドを使用して、各ユーザーに独自の

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)、および 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 **(**キーサーバー**)** を確認します。

次のコマンドを入力します。

```
# 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、次の内容を入力してキーサーバーを起動します。

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

5. 秘密鍵の復号化と保存を実行します。

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、ユーザー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、実行する必要があります。

---

6. ファイルシステムに対するマウントオプションを更新します。

/etc/dfs/dfstab ファイルを編集し、任意のエントリ (Diffie-Hellman 認証) に sec=dh オプションを追加します。

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

/etc/dfs/dfstab については、dfstab(4) のマニュアルページを参照してください。

7. ファイルシステムに対するオートマウンタマップを更新します。

   auto_master データを編集し、任意のエントリ (Diffie-Hellman 認証) 内にマウントオプションとして sec=dh を含めます。

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

注 – Solaris 2.5 以前のリリースでは、その機能が制限されています。クライアントが、共有されているファイルシステムと同程度のセキュリティモードでマウントしない場合、ユーザーは、そのユーザー自身ではなく、nobody ユーザーとしてのアクセス権を持つことになります。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かぎり、NFS サーバー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 サーバーから引き継がれるので、クライアントが sec=dh を指定する必要はありません。ユーザーは、そのユーザー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、アップグレードしたりするときに、新しい鍵を設定せず、root 用の鍵も変更しない場合は、必ず /etc/.rootkey を保存してください。 /etc/.rootkey を削除するには、通常、次のコマンドを入力します。

```
# keylogin -r
```

WebNFS の管理作業

この節では、WebNFS システムを管理する方法について説明します。次に示すのは、関連する作業の一覧です。

表 **15–4** WebNFS 管理 (作業マップ)

作業	説明	参照先
WebNFS に関する計画を作成する	WebNFS サービスを有効にする前に考慮する項目	165 ページの「WebNFS アクセスの計画」
WebNFS を有効にする	WebNFS プロトコルを使用して NFS ファイルシステムのマウントを有効にする手順	152 ページの「WebNFS アクセスを有効にする方法」

表 **15–4** WebNFS 管理 (作業マップ)　(続き)

作業	説明	参照先
ファイアウォール経由で WebNFS を有効にする	WebNFS プロトコルを使用して、ファイアウォール経由でファイルへのアクセスを許可する手順	167 ページの「ファイアウォール経由で WebNFS アクセスを有効にする方法」
NFS URL を使ってブラウズする	Web ブラウザ内での NFS URL の使用についての説明	166 ページの「NFS URL を使ってブラウズする方法」
autofs で公共ファイルハンドルを使用する	オートマウンタでファイルシステムをマウントする場合に、公共ファイルハンドルの使用を強制するための手順	180 ページの「autofs で公共ファイルハンドルを使用する方法」
autofs で NFS URL を使用する	オートマウンタマップに NFS URL を追加するための手順	180 ページの「autofs で NFS URL を使用する方法」
ファイアウォールを越えてファイルシステムにアクセスを提供する	WebNFS プロトコルでファイアウォールを越えてファイルシステムへのアクセスを許可する手順	159 ページの「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」
NFS URL を使ってファイルシステムをマウントする	NFS URL を使ってファイルシステムへのアクセスを許可する手順。このプロセスによって、MOUNT プロトコルを使用しないファイルシステムへのアクセスが可能になる	160 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」

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 ブラウザの場合は、Web サーバーの文書のルートがよく使用されます。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は、次の事項を検討する必要があります。

1. 各サーバー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ー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、サーバーによってアクセスが実現されます。`share` コマンドの `public` オプションを使用すると、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用すると、URL はサーバーのルートファイルシステムではなく公共ファイルシステムからの相対パスになります。ルートファイルシステムを共有しないと、ルートファイルシステムへの 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ー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。ユーザーは NFS の設定によってファイルへの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恐れはありません。ファイルシステムをマウントできないユーザーは、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 アクセスを使用できるようになります。

3. 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のに適しています。たとえば、ftp アーカイブの最上位のディレクトリや Web サイトのメイン URL ディレクトリなどです。
4. share コマンドで index オプションを使用すると、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 ファイルが読み込まれます。

   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。

   また、NFS と HTTP URL の両方を使用して 1 つの Web サイトにアクセスする場合は、その他の事項も検討する必要があります。これについては、231 ページの「Web ブラウザの使用と比較した場合の WebNFS の制約」で説明します。

▼ NFS URL を使ってブラウズする方法

ブラウザが WebNFS サービスをサポートしている場合は、次のような NFS URL にアクセスできます。

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

server	ファイルサーバー名
port	使用するポート番号 (デフォルト値は 2049)
path	公共ファイルハンドラまたはルートファイルシステムに関連するファイルへのパス

注 – ほとんどのブラウザでは、前のトランザクションで使用した URL サービスのタイプ (NFS や HTTP など) を次のトランザクションでも使用します。例外は、異なるタイプのサービスを含む URL を読み込んだ場合は、前のトランザクションで使用した URL サービスのタイプが使われることがあります。NFS URL を使用した後に、HTTP URL に対する参照が読み込まれることがあります。その場合、次のページは、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 アクセスを有効にする方法

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。

autofs 管理作業の概要

この節では、ユーザー自身の環境で遭遇する可能性のあるもっとも一般的な作業について説明します。各シナリオについて、ユーザーのクライアントで必要とする条件に最も適合するように autofs を設定するために推奨される手順も示します。

注 – この節で説明する作業を実行するには、『*Solaris* のシステム管理 *(*ネーミングとディレクトリサービス *: FNS*、*NIS+* 編*)*』を参照してください。

---

autofs 管理 (作業マップ)

次の表に、autofs に関連する作業についての説明と参照箇所を示します。

表 **15–5** autofs 管理 (作業マップ)

作業	説明	参照先
autofs を起動する	システムをリブートすることなくオートマウントサービスを起動する	161 ページの「オートマウンタの起動方法」
autofs を停止する	他のネットワークサービスを使用不可にすることなくオートマウントサービスを停止する	162 ページの「オートマウンタの停止方法」
autofs でファイルシステムにアクセスする	オートマウントサービスを使ってファイルシステムにアクセスする	157 ページの「オートマウンタによるマウント」
autofs マップを修正する	他のマップをリストするために使用されるマスターマップの修正を行う手順	170 ページの「マスターマップの修正方法」
	ほとんどのマップに対して使用される間接マップの修正を行う手順	170 ページの「間接マップの修正方法」

表 **15–5** autofs 管理 (作業マップ)　(続き)

作業	説明	参照先
	クライアント上のマウントポイントとサーバー間の直接の関係が必要な場合に使用される直接マップの修正を行う手順	171 ページの「直接マップの修正方法」
非 NFS ファイルシステムにアクセスするために autofs マップを修正する	CD-ROM アプリケーション用のエントリで autofs マップを設定する手順	172 ページの「autofs で CD-ROM アプリケーションにアクセスする」
	PC-DOS フロッピーディスク用のエントリで autofs マップの設定を行う手順	173 ページの「autofs で PC-DOS データフロッピーディスクにアクセスする方法」
	autofs を使用して CasheFS ファイルシステムにアクセスする手順	173 ページの「CasheFS を使用して NFS ファイルシステムにアクセスする方法」
/home を使用する	共通の /home マップの設定方法の例	174 ページの「/home の共通表示の設定」
	複数のファイルシステムを参照する /home マップを設定する手順	175 ページの「複数のホームディレクトリファイルシステムで /home を設定する方法」
新しい autofs マウントポイントを使用する	プロジェクト関連の autofs マップを設定する手順	176 ページの「/ws 下のプロジェクト関連ファイルを統合する方法」
	異なるクライアントアーキテクチャをサポートする autofs マップを設定する手順	177 ページの「共有名前空間にアクセスするために異なるアーキテクチャを設定する方法」
	異なるオペレーティングシステムをサポートする autofs マップを設定する手順	178 ページの「非互換のクライアントオペレーティングシステムのバージョンをサポートする方法」
autofs でファイルシステムを複製する	フェイルオーバーしたファイルシステムへのアクセスを提供する	179 ページの「複数のサーバーを通じて共用ファイルを複製する方法」
autofs でセキュリティ制限を使用する	ファイルへのリモート root アクセスを制限する一方でファイルシステムへのアクセスを提供する	179 ページの「autofs セキュリティ制限を適用する方法」
autofs で公共ファイルハンドルを使用する	ファイルシステムのマウント時に公共ファイルハンドルの使用を強制する	180 ページの「autofs で公共ファイルハンドルを使用する方法」
autofs で NFS URL を使用する	オートマウンタが使用できるように、NFS URL を追加する	180 ページの「autofs で NFS URL を使用する方法」

表 **15–5** autofs 管理 (作業マップ)　(続き)

作業	説明	参照先
autofs のブラウズ機能を無効にする	autofs マウントポイントが 1 つのクライアント上で自動的に生成されないように、ブラウズ機能を無効にする手順	181 ページの「1 つの NFS クライアントの autofs ブラウズ機能を完全に無効にする方法」
	autofs マウントポイントがすべてのクライアント上で自動的に生成されないように、ブラウズ機能を無効にする手順	181 ページの「すべてのクライアントの autofs ブラウズ機能を無効にする方法」
	特定の autofs マウントポイントがある 1 つのクライアント上で自動的に生成されないように、ブラウズ機能を無効にする手順	181 ページの「選択したファイルシステムの autofs ブラウズ機能を無効にする方法」

マップの管理作業

表 15–6 は、autofs マップの管理時に認識しておく必要のある事項について示したものです。選択したマップのタイプおよびネームサービスにより、autofs マップへの変更を行うために使用する必要があるメカニズムが異なります。

表 15–6 に、マップのタイプとその使用方法を示します。

表 **15–6** autofs マップのタイプとその使用方法

マップのタイプ	使用方法
マスター	ディレクトリをマップに関連付ける
直接	autofs を特定のファイルシステム向けにする
間接	autofs をリファレンス指向のファイルシステム向けにする

表 15–7 は、ネームサービスに基づいて autofs 環境に変更を加える方法を示したものです。

表 **15–7** マップの保守

ネームサービス	メソッド
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 15–8 に、マップのタイプについて行なった修正に応じた `automount` コマンドを実行する場合を示します。たとえば、直接マップに対する追加または削除を行なった場合、ローカルシステム上で `automount` コマンドを実行し、変更が反映されるようにする必要があります。しかし、既存のエントリを修正した場合は、変更を反映するために、`automount` コマンドを実行する必要はありません。

表 **15–8** automount コマンドを実行する場合

マップのタイプ	automount を再実行するか否か	
	追加または削除	修正
auto_master	Y	Y
direct	Y	N
indirect	N	N

マップの修正

次の手順では、NIS+ をネームサービスとして使用している必要があります。

▼ マスターマップの修正方法

1. マップを変更する権限を持つユーザーとしてログインします。
2. nistbladm コマンドを使用して、マスターマップへの変更を行います。
   『*Solaris* のシステム管理 *(*ネーミングとディレクトリサービス *: FNS*、*NIS+* 編*)*』を参照してください。
3. 各クライアントで、スーパーユーザーになるか、それと同等の役割になります。
   役割については、『*Solaris* のシステム管理 *(*セキュリティサービス*)*』の「特権付きアプリケーションの使用」を参照してください。
4. 各クライアントで、automount コマンドを実行し、変更が反映されるようにします。
5. マップを変更したことを他のユーザーに通知します。
   他のユーザーがコンピュータ上でスーパーユーザーとして automount コマンドを実行できるように、通知が必要になります。

   automount コマンドは、実行時にマスターマップから情報を収集します。

▼ 間接マップの修正方法

1. マップを変更する権限を持つユーザーとしてログインします。
2. 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* のシステム管理 *(*ネーミングとディレクトリサービス *: FNS*、*NIS+* 編*)*』を参照してください。

変更は、マップを次に使用する時、つまり次回のマウント実行時に反映されます。

▼ 直接マップの修正方法

1. マップを変更する権限を持つユーザーとしてログインします。
2. `nistbladm` コマンドを使用して、直接マップに対する変更点の追加または削除を行います。

   『*Solaris* のシステム管理 *(*ネーミングとディレクトリサービス *: FNS*、*NIS+* 編*)*』を参照してください。
3. 手順 **1** でマウントポイントエントリの追加または削除を行なった場合は、`automount` コマンドを実行します。
4. マップを変更したことを他のユーザーに通知します。

   他のユーザーがコンピュータ上でスーパーユーザーとして `automount` コマンドを実行できるように、通知が必要になります。

注 – 既存の直接マップエントリの内容の変更だけを行なった場合は、`automount` コマンドを実行する必要はありません。

たとえば、異なるサーバーから `/usr/src` ディレクトリがマウントされるように `auto_direct` マップを修正するとします。`/usr/src` がその時点でマウントされていない場合、`/usr/src` にアクセスするとすぐにその新しいエントリが反映されます。`/usr/src` がその時点でマウントされている場合、オートアンマウントが実行されるまで待ちます。その後、アクセスが可能になります。

注 – 直接マップを修正するには上記のような手順が必要であり、間接マップは直接マップよりもマウントテーブル内のスペースを必要としないので、可能であれば間接マップを使用してください。間接マップは構築が容易であり、コンピュータのファイルシステムへの要求が少なくて済みます。

マウントポイントの重複回避

`/src` 上にマウントされたローカルなディスクパーティションがあり、他のソースディレクトリのマウントにもその autofs サービスを使用する場合、問題が発生する可能性があります。マウントポイント `/src` を指定する場合、ユーザーがアクセスするたびに、NFS サービスが対象のローカルパーティションを隠すことになります。

たとえば `/export/src` などの他の場所に、パーティションをマウントする必要があります。その後、次のようなエントリを `/etc/vfstab` に含める必要があります。

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

このエントリは、auto_src にも必要です。

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

terra はコンピュータ名です。

非 NFS 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することができます。autofs は、フロッピーディスクや CD-ROM など、削除可能な媒体上のファイルをマウントします。通常は、ボリュームマネージャを使って削除可能な媒体上のファイルをマウントすることになります。次の例では、autofs を利用してこのマウントがどのように行われるかを示します。ボリュームマネージャと autofs は同時に動作することができないため、まずボリュームマネージャを終了してから次に示すエントリを使用する必要があります。

サーバーからファイルシステムのマウントを行う代わりに、ドライブに媒体を配置してマップから参照します。autofs を使用し非 NFS ファイルシステムにアクセスを行う場合は、次の手順を参照してください。

autofs で 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用していない場合に、この手順を行なってください。

---

1. スーパーユーザー、またはそれと同等の役割になります。
   役割については、『*Solaris* のシステム管理 *(*セキュリティサービス*)*』の「特権付きアプリケーションの使用」を参照してください。
2. **autofs** マップを更新します。
   次のような CD-ROM のファイルシステム用のエントリを追加します。

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

   マウントする CD-ROM 装置の名前が、コロンの後に続けて表示されます。

▼ autofs で PC-DOS データフロッピーディスクにアクセスする方法

注 – ボリュームマネージャを使用していない場合に、この手順を行なってください。

1. スーパーユーザー、またはそれと同等の役割になります。
   役割については、『*Solaris* のシステム管理 (セキュリティサービス)』の「特権付きアプリケーションの使用」を参照してください。
2. **autofs** マップを更新します。
   次のようなフロッピーディスクのファイルシステム用のエントリを追加します。

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

CasheFS を使用して NFS ファイルシステムにアクセスする

キャッシュファイルシステム (CacheFS) は、汎用不揮発性キャッシュメカニズムで、小型で高速ローカルディスクを利用して、特定のファイルシステムのパフォーマンスを向上させます。

CacheFS を使ってローカルディスク上に NFS ファイルシステムからデータをキャッシュすることにより、NFS 環境のパフォーマンスを改善できます。

▼ CasheFS を使用して NFS ファイルシステムにアクセスする方法

1. スーパーユーザー、またはそれと同等の役割になります。
   役割については、『*Solaris* のシステム管理 (セキュリティサービス)』の「特権付きアプリケーションの使用」を参照してください。
2. `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
```

3. 任意の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、次に示すエントリをマスターマップに追加して、すべての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
```

以下のエントリを auto_home マップに追加すると、rich という名称のユーザーのホームディレクトリのキャッシュだけが行われます。

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

注 – 後から検索されるマップ内のオプションは、先に検索されたマップ内のオプションを無効にします。そのため、最後に検出されたオプションが使用されます。前述の例では、マスターマップにリストされたオプションの中に変更の必要がある場合は、auto_home マップに追加された特定のエントリがそのマスターマップのオプションを含む必要だけがあります。

オートマウンタのカスタマイズ

オートマウンタマップの設定方法はいくつかあります。次に、オートマウンタマップをカスタマイズして簡単に使用できるディレクトリ構造を実現する方法について詳細に説明します。

▼ /home の共通表示の設定

ネットワークユーザーすべてにとって理想的なのは、自分自身のホームディレクトリ、または他の人のホームディレクトリを /home の下に配置できるようにすることです。この表示方法は通常、クライアントでもサーバーでも、すべてのコンピュータを通じて共通です。

Solaris をインストールすると、常にマスターマップ /etc/auto_master もインストールされます。

```
# 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 の下にインストールされます。

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

外部 auto_home マップに対する参照を除き、このマップは空になります。/home 下のディレクトリをすべてのコンピュータに対して共通にする場合、この /etc/auto_home マップは修正しないでください。すべてのホームディレクトリのエントリは、NIS または NIS+ のネームサービスファイルで表示されなくてはなりません。

注 – ユーザーは、各ホームディレクトリから setuid 実行可能ファイルを実行することが許可されていません。この制限がないと、すべてのユーザーがすべてのコンピュータ上でスーパーユーザーの権限を持つことになります。

▼ 複数のホームディレクトリファイルシステムで /home を設定する方法

1. スーパーユーザー、またはそれと同等の役割になります。

   役割については、『*Solaris* のシステム管理 (セキュリティサービス)』の「特権付きアプリケーションの使用」を参照してください。

2. /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。

   システムに複数のパーティションがある場合は、/export/home1、/export/home2 のように、別のディレクトリにそれぞれインストールを行います。

3. auto_home マップを作成して維持します。

   新しいユーザーアカウントを作成する場合は、そのユーザーのホームディレクトリの場所を auto_home マップに入力します。マップのエントリは、次のように単純な形式にすることができます。

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

   マップキーを置換する & (アンパサンド) の使い方に注意してください。このアンパサンドは、次の例の 2 つ目の rusty の使用を省略した形式です。

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

   auto_home マップを配置すると、ユーザーは、/home/*user* というパスを使用して、ユーザー自身のホームディレクトリを含むあらゆるホームディレクトリを参照できます。*user* はログイン名で、マップ内でのキーになります。すべてのホームディレクトリを共通に表示するしくみは、他のユーザーのコンピュータにログインする場合に便利です。autofs は、ユーザー自身のホームディレクトリをマウントします。同様に、他のコンピュータ上でリモートのウィンドウシステムクライアントを実行するとウィンドウシステムクライアントと同じ /home ディレクトリが表示されます。

   この共通表示は、サーバーにも拡張されています。前の例を使用すれば、rusty がサーバー dragon にログインする場合、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り、ローカルディスクへの直接アクセスを提供します。

   ユーザーは、各ホームディレクトリの実際の位置を意識する必要はありません。rusty がさらにディスクスペースを必要とし、rusty 自身のホームディレクトリを他のサーバーに再配置する必要がある場合には、auto_home マップ内の rusty のエントリを新しい場所を反映するように変更することだけが必要になります。他のユー

ザーは、/home/rusty パスを継続して使用することができます。

▼ /ws 下のプロジェクト関連ファイルを統合する方法

大きなソフトウェアの開発プロジェクトの管理者を想定してください。そこで、プロジェクト関連のファイルをすべて /ws というディレクトリの下で利用できるようにすると仮定します。このようなディレクトリは、そのサイトのすべてのワークステーションで共通である必要があります。

1. /ws ディレクトリに対するエントリを、サイトの **NIS** または **NIS+** の auto_master マップに追加します。

   ```
   /ws     auto_ws     -nosuid
   ```

   auto_ws マップが、/ws ディレクトリの内容を決定します。

2. -nosuid オプションを用心のために追加しておきます。

   このオプションは、すべての作業空間に存在する可能性のある setuid プログラムをユーザーが実行できないようにします。

3. auto_ws マップにエントリを追加します。

   auto_ws マップは、各エントリがサブプロジェクトを記述するように構成されています。最初の操作により、マップが次のようになります。

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

   各エントリの最後のアンパサンド (&) は、エントリキーを省略したものです。たとえば、最初のエントリは次のエントリと同じ意味です。

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

   この最初の操作により、マップはシンプルなものになりますが、このマップでは不十分です。プロジェクトの調整者が、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のサブディレクトリとして提供しようとしているとします。さらに、各サブプロジェクトは、ソフトウェアの複数のバージョンを記述するために、複数のサブディレクトリを必要とします。この場合、サーバー上のディスクパーティション全体に対して、これらのサブディレクトリをそれぞれ割り当てる必要があります。

   次のように、マップ内のエントリを修正してください。

   ```
   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/&
```

現在のマップはかなり長くなっていますが、まだ 5 つのエントリを含んでいるだけです。各エントリは、複数のマウントがあるために長くなっています。たとえば、/ws/compiler に対する参照は、vers1.0、vers2.0、および man ディレクトリ用に 3 つのマウントを必要とします。各行の最後のバックスラッシュは、エントリが次の行まで続いていることを autofs に伝えるものです。実際、エントリは 1 つの長い行となっていますが、行ブレークやインデントのいくつかはエントリを読みやすくする目的で使用されています。tools ディレクトリには、すべてのサブプロジェクトに対するソフトウェア開発ツールが含まれているため、同じサブディレクトリ構造の対象とはなっていません。tools ディレクトリは単一のマウントのままです。

この配置は、システムの管理者に大きな柔軟性を提供します。ソフトウェアプロジェクトでは、非常に大きなディスクスペースを消費します。プロジェクトのすべての過程を通じて、さまざまなディスクパーティションを再配置し、拡張することになる可能性もあります。このような変更が auto_ws マップに反映される場合は、/ws 下のディレクトリ階層構造が変更されることもなく、ユーザーに対する通知の必要はありません。

サーバー alpha と bravo が同一の autofs マップを参照するため、それらのコンピュータにログインするすべてのユーザーは期待通りに /ws 名前空間を発見できます。このようなユーザーには、NFS マウントではなく、ループバックマウントを通じてのローカルファイルへの直接アクセスが提供されます。

▼ 共有名前空間にアクセスするために異なるアーキテクチャを設定する方法

表計算アプリケーションやワードプロセッサパッケージのようなローカルの実行可能ファイルやアプリケーションについて、共有名前空間を作成する必要があります。この名前空間のクライアントは、異なる実行可能フォーマットを必要とする複数の異なるワークステーションアーキテクチャを使用します。また、ワークステーションには、異なるリリースのオペレーシングシステムを使用するものもあります。

1. nistabladm コマンドで auto_local マップを作成します。

   『*Solaris* のシステム管理 *(*ネーミングとディレクトリサービス *: FNS*、*NIS+* 編*)*』を参照してください。

2. その名前空間に属するファイルとディレクトリを簡単に識別できるように、共有名前空間について、サイト固有の名称を **1** つ選択します。

たとえば、その名称として /usr/local を選択した場合、/usr/local/bin パスは明らかにこの名前空間の一部です。

3. ユーザーのコミュニティ識別を簡単にするため、**autofs** 間接マップを作成し、/usr/local にマウントします。**NIS (**または **NIS+)** の auto_master マップ内で、次のエントリを設定します。

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

なお、-ro マウントオプションは、クライアントがファイルやディレクトリのすべてに対して書き込みができないことを示します。

4. サーバー上の任意のディレクトリをエクスポートします。

5. **auto_local** マップ内に bin エントリを **1** つ含めます。

ディレクトリ構造は、次のようになります。

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

6. **(**省略可能**)** 異なるアーキテクチャのクライアントを処理するため、**autofs** CPU 変数を加えて、エントリの変更を行います。

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

- SPARC クライアント – 実行可能ファイルを /export/local/bin/sparc に配置する
- IA クライアント – 実行可能ファイルを /export/local/bin/i386 に配置する

▼ 非互換のクライアントオペレーティングシステムのバージョンをサポートする方法

1. クライアントのオペレーティングシステムのタイプを決定する変数と、アーキテクチャタイプを結合します。

autofs OSREL 変数と CPU 変数を結合して、CPU タイプと OS リリースの両方を示す名前を作成することができます。

2. 次のような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動作させているクライアントについて、次のファイルシステムをエクスポートします。

- SPARC クライアント – /export/local/bin/sparc5.6 をエクスポートする
- IA クライアント – 実行可能ファイルを /export/local/bin/i3865.6 に配置する

▼ 複数のサーバーを通じて共用ファイルを複製する方法

読み取り専用の複製されたファイルシステムを共有する最良の方法は、フェイルオーバーの利用です。フェイルオーバーについての説明は、227 ページの「クライアント側フェイルオーバー機能」を参照してください。

1. スーパーユーザー、またはそれと同等の役割になります。
   役割については、『*Solaris* のシステム管理 (セキュリティサービス)』の「特権付きアプリケーションの使用」を参照してください。
2. **autofs** マップ内のエントリを修正します。
   すべての複製サーバーのリストを、コンマ区切りのリストとして、次のように作成します。

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

   autofs は、最も近いサーバーを選択します。サーバーが複数のネットワークインタフェースを持っている場合は、各インタフェースのリストを作成してください。autofs はクライアントに最も近接したインタフェースを選択し、NFS トラフィックの不必要なルーティングを避けるようにしています。

▼ autofs セキュリティ制限を適用する方法

1. スーパーユーザー、またはそれと同等の役割になります。
   役割については、『*Solaris* のシステム管理 (セキュリティサービス)』の「特権付きアプリケーションの使用」を参照してください。
2. **NIS** または **NIS+** のネームサービス `auto_master` ファイル内に次のようなエントリを作成します。

   ```
   /home      auto_home      -nosuid
   ```

   `nosuid` オプションは、`setuid` または `setgid` ビットを設定したファイルをユーザーが作成できないようにします。
   このエントリは、汎用ローカルファイル `/etc/auto_master` 内の `/home` のエントリを無効にします (前の例を参照)。これは、`+auto_master` が、ファイル内の `/home` エントリより先に、外部のネームサービスマップを参照するためです。`auto_home` マップ内のエントリにマウントオプションがある場合、`nosuid` オプションは無効になります。そのため、`auto_home` マップ内でオプションを使用しないようにするか、`nosuid` オプションを各エントリに含める必要があります。

注 – サーバー上の `/home` 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用する方法

1. スーパーユーザー、またはそれと同等の役割になります。
   役割については、『*Solaris* のシステム管理 *(*セキュリティサービス*)*』の「特権付きアプリケーションの使用」を参照してください。
2. **autofs** マップに、次のようなエントリを作成します。

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

   `public` オプションは、公共ハンドルの使用を強制します。NFS サーバーが公共ファイルハンドルをサポートしない場合、マウントは失敗します。

▼ autofs で NFS URL を使用する方法

1. スーパーユーザー、またはそれと同等の役割になります。
   役割については、『*Solaris* のシステム管理 *(*セキュリティサービス*)*』の「特権付きアプリケーションの使用」を参照してください。
2. 次のような **autofs** エントリを作成します。

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

   サービスは、NFS サーバー上で公共ファイルハンドルの使用を試みますが、そのサーバーが公共ファイルハンドルをサポートしない場合、MOUNT プロトコルが使用されます。

autofs のブラウズ機能を無効にする

Solaris 2.6 から、インストールされる `/etc/auto_master` のデフォルトバージョンには、`/home` と `/net` 用のエントリに追加された `-nobrowse` オプションが含まれます。さらに、アップグレード手順により、`/home` と `/net` のエントリが修正されていない場合は、`-nobrowse` オプションがそれらのエントリに追加されます。ただし、このような変更を手動で加えるか、あるいはインストール後にサイト固有の autofs マウントポイントに対するブラウズ機能をオフにすることが必要な場合もあります。

ブラウズ機能をオフにする方法はいくつかあります。`automountd` デーモンに対してコマンド行オプションを使用してブラウズ機能を無効にすると、そのクライアントに対する autofs ブラウズ機能は完全に無効になります。あるいは、NIS 名前空間または NIS+ 名前空間の autofs マップを使用して、すべてのクライアントにおける各マップエントリのブラウズ機能を無効にします。また、ネットワーク規模の名前空間を使用していない場合は、ローカルな autofs を使用して、各クライアントにおける各マップエントリのブラウズ機能を無効にすることができます。

▼ 1 つの NFS クライアントの autofs ブラウズ機能を完全に無効にする方法

1. **NFS** クライアント上で、スーパーユーザー、またはそれと同等の役割になります。
   役割については、『*Solaris* のシステム管理 (セキュリティサービス)』の「特権付きアプリケーションの使用」を参照してください。
2. -n オプションを起動スクリプトに追加します。
   root として、/etc/init.d/autofs スクリプトを編集して、autmountd デーモンを開始する行に -n オプションを次のように追加します。

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

3. **autofs** サービスを再起動します。

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

▼ すべてのクライアントの autofs ブラウズ機能を無効にする方法

すべてのクライアントに対するブラウズ機能を無効にするには、NIS または NIS+ のようなネームサービスを使用する必要があります。それ以外の場合には、各クライアント上で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 ディレクトリのブラウズ機能が無効にされています。無効にする必要がある各間接 autofs ノードに対して、この手順を実行してください。

1. ネームサービス auto_master ファイル内の /home エントリに -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上で、automount コマンドを実行します。
   新規の動作は、クライアントシステム上で automount コマンドを実行した後、またはリブートした後に反映されます。

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

▼ 選択したファイルシステムの autofs ブラウズ機能を無効にする方法

この例では、/net ディレクトリのブラウズ機能を無効にします。/home または他の autofs マウントポイントにも、同じ手順を使用できます。

1. automount エントリが /etc/nsswitch.conf にあることを確認します。

優先するローカルファイルエントリについては、ネームサービススイッチファイル内のエントリがネームサービスの前に files をリストする必要があります。たとえば:

```
automount:  files nisplus
```

これは、標準的な Solaris にインストールされるデフォルトの構成です。

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

   名前空間内のエントリに優先するローカルファイルへの追加については、+auto_master エントリが /net の下に移動されている必要があります。

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

   標準的な構成では、+auto_master エントリがファイルの先頭に配置されます。このように配置することにより、ローカルな変更が使用されなくなります。

3. /etc/auto_master ファイル内の /net エントリに -nobrowse オプションを追加します。

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

4. すべてのクライアント上で、automount コマンドを実行します。

   新規の動作は、クライアントシステム上で automount コマンドを実行した後、またはリブートした後に反映されます。

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

NFS の障害追跡の方法

NFS の問題を追跡するには、問題が発生する可能性があるのは主に、サーバー、クライアント、およびネットワークで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け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、サーバー上で 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。そのため、はじめて共有を設定する場合は、mountd および nfsd を手動で起動する必要があります。

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」(サーバーが応答しません) 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ーに問題がある場合、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、サーバーがふたた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にエラーを返します。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発生する恐れがあるため、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、サーバーが応答に失敗した場合は、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
```

サーバー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
```

サーバー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 の障害追跡の手順

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。次の項目をチェックしてください。

- クライアントがサーバーに到達できるかどうか
- クライアントがサーバー上の NFS サービスを受けられるかどうか
- NFS サービスがサーバー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。複数のネームサービスでのデバッグ手順については、『*Solaris* のシステム管理 (ネーミン

グとディレクトリサービス *: DNS*、*NIS*、*LDAP* 編*)*』 で説明しています。また、上記の項目をチェックする過程で、クライアント側には問題がないことが判明することもあります。たとえば、作業領域のすべてのサブネットから、少なくとも 1 つの障害が発生したことが通知された場合などです。このような場合は、問題がサーバーかサーバー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、サーバーでデバッグを開始することをお勧めします。

▼ NFS クライアントの接続性を確認する方法

1. クライアントから **NFS** サーバーに到達できることを確認します。クライアントで次のコマンドを入力します。

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

   コマンドを入力した結果、サーバーが動作していることがわかったら、NFS サーバーをリモートで確認します。185 ページの「NFS サーバーをリモートで確認する方法」を参照してください。

2. クライアントからサーバーに到達できない場合は、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+ クライアントで次のコマンドを入力します。

   ```
   % /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ー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、186 ページの「サーバーで NFS サービスを確認する方法」 を参照してください。

5. 別のクライアントからサーバーに到達できる場合は、`ping` コマンドを使っ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確認します。

   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確認します (`/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確認します。

   クライアントをネットワークの別の場所へ移動して確認します。

▼ NFS サーバーをリモートで確認する方法

1. **NFS** サーバーで **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して確認します。

```
% rpcinfo -s bee|egrep 'nfs|mountd'
 100003  3,2    tcp,udp,tcp6,upd6                nfs     superuser
 100005  3,2,1  ticots,ticotsord,tcp,tcp6,ticlts,udp,upd6  mountd  superuser
```

   デーモンが起動していない場合、187 ページの「NFS サービスを再起動する方法」 を参照してください。

2. サーバーで nfsd プロセスが応答することを確認します。

   クライアント上で、次のコマンドを入力し、サーバーからの UDP NFS 接続をテストします。

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

   サーバー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。上記コマンドでエラーになる場合は、186 ページの「サーバーで NFS サービスを確認する方法」 に進んでください。

3. サーバーで mountd が応答することを確認します。次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
```

   サーバーが動作している場合は、UDP プロトコルに関連しているプログラムとそのバージョン番号が出力されます。 -t オプションを使用すると、TCP 接続を検査できます。エラーになる場合は、186 ページの「サーバーで 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、そのサービスを確認します。

```
% 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確認します。エラーになる場合は、次のコマンドを root としてクライアントから入力し、autofs サービスを再起動します。

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

5. サーバー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)
```

   サーバー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。名前空間も確認します。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。

▼ サーバーで NFS サービスを確認する方法

1. スーパーユーザー、またはそれと同等の役割になります。

   役割については、『*Solaris* のシステム管理 *(*セキュリティサービス*)*』の「特権付きアプリケーションの使用」を参照してください。

2. サーバーがクライアントに到達できることを確認します。

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

3. サーバーからクライアントに到達できない場合は、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+** クライアントで次のコマンドを入力します。

```
% /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
```

4. ネームサービスが動作している場合は、サーバーにあるネットワークソフトウェアの構成を確認します **(**/etc/netmasks、 /etc/nsswitch.conf など**)**。

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

```
# 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
```

   サーバーが動作している場合は、UDP プロトコルに関連しているプログラムとそのバージョン番号が出力されます。 rpcinfo に -t オプションを指定し、TCP 接続も確認します。これらのコマンドを使用するとエラーになる場合は、NFS サービスを再起動します。 187 ページの「NFS サービスを再起動する方法」 を参照してください。

6. 次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
```

サーバーが動作している場合は、UDP プロトコルに関連しているプログラムとそのバージョン番号が出力されます。 rpcinfo に -t オプションを指定し、TCP 接続も確認します。これらのコマンドを使用するとエラーになる場合は、NFS サービスを再起動します。 187 ページの「NFS サービスを再起動する方法」 を参照してください。

7. 次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
```

サーバーが動作している場合は、UDP プロトコルに関連しているプログラムとそのバージョン番号が出力されます。 rpcbind がハングしている場合は、サーバーをリブートするか、187 ページの「rpcbind をウォームスタートする方法」 に示す作業を行なってください。

▼ NFS サービスを再起動する方法

1. スーパーユーザー、またはそれと同等の役割になります。

   役割については、『*Solaris* のシステム管理 (セキュリティサービス)』の「特権付きアプリケーションの使用」を参照してください。

2. リブートせずにデーモンを有効にするために、次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 をウォームスタートする方法

実行中の処理があるために NFS サーバーをリブートできなかった場合に、RPC を使用するすべてのサービスを再起動することなく rpcbind を再実行できます。この手順に従ってウォームスタートを完了します。

1. スーパーユーザー、またはそれと同等の役割になります。

   役割については、『*Solaris* のシステム管理 (セキュリティサービス)』の「特権付きアプリケーションの使用」を参照してください。

2. rpcbind の **PID** を確認します。

   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
```

3. **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を参照)。このコマンドより、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 オプションを使用して rpcbind プロセスを終了させないと、rpcbind のウォームスタートを完了できません。その場合は、サーバーをリブートすることによってサービスを再開する必要があります。

---

4. rpcbind を再起動します。

   rpcbind コマンドを再度ウォームスタートして、kill コマンドにより作成されたファイルが参照され、すべての RPC サービスを再起動することなくプロセスが再開されます。rpcbind(1M) のマニュアルページを参照してください。

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

▼ NFS ファイルサービスを提供しているホストを確認する方法

-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。現在のサーバー名は、「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、次の手順に従います。

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. 次のコマンドを実行し、オプションを確認します。

```
% 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
```

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もっとも正確な方法です。

2. /etc/mnttab でエントリを確認します。

   mount コマンドは、無効なオプションをマウントテーブルに追加することができません。そのため、mnttab ファイルに記述されているオプションとコマンド行のオプションが一致していることを特定してください。このようにすると、nfsstat コマンドにより報告されなかったオプションを特定することができます。

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

autofs の障害追跡

autofs の使用時、問題に遭遇することがあります。この節では、問題解決プロセスについてわかりやすく説明します。この節は、2 つのパートに分かれています。

この節では、autofs が生成するエラーメッセージのリストを示します。このリストは、2 つのパートに分かれています。

- automount の詳細形式 (-v) オプションにより生成されるエラーメッセージ
- 通常表示されるエラーメッセージ

各エラーメッセージの後には、そのメッセージの説明と考えられる原因が続きます。

障害追跡時には、詳細形式 (-v) オプションで autofs プログラムを開始します。そうしないと、理由がわからないまま問題に遭遇することになります。

次の節は、autofs のエラー時に表示されがちなエラーメッセージと、生じうる問題についての説明です。

automount -v により生成されるエラーメッセージ

bad key *key* in direct map *mapname*
: 直接マップのスキャン中、autofs が接頭辞 / のないエントリーキーを発見しました。直接マップ内のキーは、完全パス名でなくてはなりません。

`bad key` *key* `in indirect map` *mapname*
間接マップのスキャン中、autofs が / を含むエントリキーを発見しました。間接マップのキーは、パス名ではなく、単なる名称でなくてはなりません。

`can't mount` *server*`:`*pathname: reason*
サーバー上のマウントデーモンが、*server*:*pathname* により (1 つまたは複数) 指定されます。サーバー上のエクスポートテーブルを確認してください。

`couldn't create mount point` *mountpoint*`:` *reason*
autofs は、マウントに必要なマウントポイントを作成することができませんでした。この問題は、すべてのサーバーのエクスポートされたファイルシステムを階層的にマウントしようとする場合に頻繁に生じます。必要なマウントポイントは、マウントできないファイルシステム内にだけ存在するため、エクスポートできません。エクスポートされる親ファイルシステムは、読み取り専用でエクスポートされるため、マウントポイントを作成できません。

`leading space in map entry` *entry* `text in` *mapname*
autofs はオートマウントマップ内に先頭にスペースを含むエントリを発見しました。この問題は、通常、マップエントリが不当である場合に発生します。次のような例があります。

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

この例では、autofs が 2 つ目の行に遭遇した場合に警告が生成されます。これは、最初の行がバックスラッシュ (\) で終端されていないためです。

mapname`: Not found`
必要とされるマップが配置されていません。このメッセージは、`-v` オプションが使用されている場合にだけ生成されます。マップ名のスペルとパス名を確認してください。

`remount` *server*`:`*pathname* `on` *mountpoint*`: server not responding`
autofs が、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。

`WARNING:` *mountpoint* `already mounted on`
autofs が、既存のマウントポイント上にマウントしようとしました。このメッセージは、autofs 内で内部エラー (異常) が生じたことを意味しています。

その他のエラーメッセージ

`dir` *mountpoint* `must start with '/'`
オートマウンタのマウントポイントは、完全パス名で指定しなくてはなりません。マウントポイントのスペルとパス名を確認してください。

`hierarchical mountpoints:` *pathname1* `and` *pathname2*
autofs は、マウントポイントが階層的な関係を持つことを許可しません。autofs マウントポイントは、他のオートマウントされたファイルシステムに含まれていてはなりません。

`host` *server* `not responding`
autofs が、*server* で示されるサーバーにコンタクトしようとしましたが、応答がありません。

hostname`: exports:` *rpc_err*
このエラーは、*hostname* からエクスポートリストを取得する場合に発生します。このメッセージは、サーバーまたはネットワークに問題があることを示します。

`map` *mapname*`, key` *key*`: bad`
マップエントリが不適切な形式であり、autofs が処理できません。そのエントリを再確認してください。そのエントリにエスケープする必要のある文字が含まれている可能性があります。

mapname`:` *nis_err*
このエラーは、NIS マップのエントリを参照する場合に発生します。このメッセージは、NIS に問題がある可能性があることを示しています。

`mount of` *server*`:`*pathname* `on` *mountpoint:reason*
autofs がマウントに失敗しました。サーバーまたはネットワークに問題のある可能性があります。

mountpoint`: Not a directory`
autofs は、ディレクトリではない *mountpoint* に示される場所に自分自身をマウントすることができません。マウントポイントのスペルとパス名を確認してください。

`nfscast: cannot send packet:` *reason*
autofs が、複製されたファイルシステムの場所を示すリスト内にあるサーバーへの照会パケットを送信できません。

`nfscast: cannot receive reply:` *reason*
autofs が、複製されたファイルシステムの場所を示すリスト内にあるいずれのサーバーからも応答を受けられません。

`nfscast: select:` *reason*
このようなエラーメッセージはすべて、複製されたファイルシステムのサーバーに対して `ping` を実行した際に問題が発生したことを示します。このメッセージは、ネットワークに問題がある可能性があることを示しています。

`pathconf:pathconf: no info for` *server*`:`*pathname*
autofs が、パス名に関する `pathconf` 情報の取得に失敗しました。 (`fpathconf(2)` のマニュアルページを参照。)

`pathconf: server: server not responding`
autofs が、`pathconf(2)` に情報を提供する *server* に示されるサーバー上のマウントデーモンにコンタクトできませんでした。

autofs のその他のエラー

`/etc/auto*` ファイルが実行ビットセットを持っている場合、オートマウンタは次のようなメッセージを生成するマップの実行を試みます。

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

この場合、`auto_home` ファイルは不適切な権限をもつことになります。このファイル内の各エントリは、よく似たエラーメッセージを生成します。ファイルへのこのような権限は、次のコマンドを入力することにより取り消す必要があります。

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

NFS のエラーメッセージ

この節では、エラーメッセージとそのエラーを発生させる原因となった状態について説明し、1 つ以上の解決策を提供しています。

`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用できません。

`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。またこのメッセージは、UDP の状態を示すことがあります。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 udp 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/udp    nfs    # NFS server daemon
```

`Cannot use index option without public option`
`share` コマンドの `index` オプションに `public` オプションを指定してください。`index` オプションを使用するには、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 start` *daemon* `:` *error*
このメッセージは、デーモンが異常終了するか、システムコールにエラーが発生した場合に表示されます。*error* の文字列によって、問題が特定されます。

`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 サーバー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、マウントが失敗します。この問題を解決するには、公共ファイルハンドルを使用しないでマウント要求を行うか、NFS サーバーが公共ファイルハンドルをサポートするように設定し直します。

daemon `running already with pid` *pid*
デーモンがすでに実行されています。新たにデーモンを実行する場合は、現在のデーモンを終了し、新しいデーモンを開始します。

`error locking` *lock file*
このメッセージは、デーモンに関連付けられているロックファイルを正しくロックできなかった場合に表示されます。

`error checking` *lock file* `:` *error*
このメッセージは、デーモンに関連付けられているロックファイルを正しく開くことができなかった場合に表示されます。

`NOTICE: NFS3: failing over from` *host1* `to` *host2*
このメッセージは、フェイルオーバー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、2G 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ーがダウンしているかまたは到達できないことを示すメッセージです。

`mount: ... server not responding: RPC_PROG_NOT_REGISTERED`
マウント要求が 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または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。ディレクトリ名のスペルをチェックするか、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`
コンピュータ名が、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 `mount_nfs(1M)` のマニュアルページ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Solaris 2.6 より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` オプションを使用して NFS サーバー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、大規模ファイルを処理できません。大規模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 サーバーは停止します。このメッセージは、NFS サーバー (*hostname*) がダウンしているか、サーバーかネットワークに問題があることを示すものです。フェイルオーバー機能を使用している場合、*hostname* はサーバー名のリストになります。184 ページの「NFS クライアントの接続性を確認する方法」を参照してください。

`NFS fsstat failed for server `*hostname*`: RPC: Authentication error`
さまざまな状況で発生するエラーです。もっともデバッグが困難なのは、ユーザーの属しているグループが多すぎる場合です。現在、ユーザーは最大 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。ただし、ユーザーが 17 個以上のグループに所属する方法もあります。NFS サーバーおよび NFS クライアントで Solaris 2.5 リリース以降が動作している場合は、アクセス制御リストを使用して、必要なアクセス特権を与えることができます。

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。

replicas must have the same version
: NFS フェイルオーバー機能が正しく機能するためには、複製の NFS サーバー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 のサーバーが混在することは許されません。

replicated mounts must be read-only
: NFS フェイルオーバー機能は、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 のフェイルオーバー機能は、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、フェイルオーバー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、サーバーあたり 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 サーバー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 16 章

リモートファイルシステムへのアクセス (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。

NFS ファイル

いくつかのファイルでは、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。それらの多くは ASCII ファイルで、いくつかはデータファイルです。表 16–1 に、このようなファイルとその機能をまとめます。

表 16–1 NFS ファイル

ファイル名	機能
`/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示す
`/etc/default/nfs`	`lockd` および `nfsd` の構成情報を示す
`/etc/default/nfslogd`	NFS サーバーログデーモン (`nfslogd`) の構成情報を示す
`/etc/dfs/dfstab`	共有するローカルリソースを示す

表 **16–1** NFS ファイル　(続き)

ファイル名	機能
/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示す
/etc/dfs/sharetab	共有されるローカルとリモートのリソースを示す (sharetab(4) のマニュアルページを参照)。このファイルは編集しないでください
/etc/mnttab	自動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示す (mnttab(4) のマニュアルページを参照)。このファイルは編集しないでください
/etc/netconfig	トランスポートプロトコルのリスト。このファイルは編集しないでください
/etc/nfs/nfslog.conf	NFS サーバーログのための一般的な構成情報を示す
/etc/nfs/nfslogtab	nfslogd によるログ後処理のための情報を示す。このファイルは編集しないでください
/etc/nfssec.conf	NFS のセキュリティサービスのリスト。このファイルは編集しないでください
/etc/rmtab	NFS クライアントがリモートでマウントしたファイルシステムを示す (rmtab(4) のマニュアルページを参照)。このファイルは編集しないでください
/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を参照)

/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利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ーでサポートするファイルシステムのタイプは、/kernel/fs のファイルを確認して決定することができます。

/etc/default/nfslogd

このファイルは、NFS サーバーログ機能を使用するときに使用されるいくつかのパラメータを定義します。次のパラメータを定義することができます。

CYCLE_FREQUENCY
: ログファイルを元の状態に戻す前に経過させる必要がある時間数を決めるパラメータです。デフォルト値は 24 時間です。このパラメータはログファイルが大きくなり過ぎないように使用します。

IDLE_TIME
nfslogd が、バッファーファイル内にさらに情報があるかどうかを確認するまでに休眠しなければならない秒数を設定するパラメータです。このパラメータは、構成ファイルの確認頻度も決定します。このパラメータと MIN_PROCESSING_SIZE によりバッファーファイルの処理頻度が決まります。デフォルト値は 300 秒です。この数値を増加させると、確認の回数が減ってパフォーマンスが向上します。

MAPPING_UPDATE_INTERVAL
ファイルハンドルパスマッピングテーブル内でレコードを更新する間隔を秒数で指定します。デフォルト値は 86400 秒つまり 1 日です。このパラメータを使用すると、ファイルハンドルパスマッピングテーブルを常時更新しないで最新の状態に保つことができます。

MAX_LOG_PRESERVE
保存するログファイル数を決めます。デフォルト値は 10 です。

MAN_PROCESSING_SIZE
バッファーファイルが処理してログファイルに書き込むための最小限のバイト数を設定します。このパラメータと IDLE_TIME によりバッファーファイルの処理頻度が決まります。このパラメータのデフォルト値は 524288 バイトです。この数値を大きくするとバッファーファイルの処理回数が減ってパフォーマンスを向上できます。

PRUNE_TIMEOUT
ファイルハンドルパスマッピングレコードを中断して除去できるようになるまでに経過しなければならない時間数を選択するパラメータです。デフォルト値は 168 時間、つまり 7 日間です。

UMASK
nfslogd によって作成されるログファイルのファイルモード生成マスクを指定します。デフォルト値は 0137 です。

/etc/nfs/nfslog.conf

このファイルは nfslogd で使用するログのパス、ファイル名、およびタイプを定義します。各定義はタグと関連付けられています。NFS サーバーのログを開始するためには、各ファイルシステムについてタグを付ける必要があります。広域タグはデフォルト値を定義します。必要に応じて、各タグに、次のパラメータを使用することができます。

defaultdir=*path*
ログファイルのデフォルトのディレクトリパスを指定するパラメータです。

log=*path/filename*
ログファイルのパスとファイル名を指定するパラメータです。

fhtable=*path/filename*
ファイルハンドルパスデータベースのパスとファイル名を選択するパラメータです。

buffer=*path/filename*
バッファーファイルのパスとファイル名を決定するパラメータです。

logformat=*basic* | *extended*
ユーザーから読み取り可能なログファイルを作成するときに使用するフォーマットを選択します。基本フォーマットは、ftpd デーモンと同様なログファイルが作成されます。拡張フォーマットは、より詳細に表示されます。

パスとファイル名の両方を指定することができるパラメータについては、パスが指定されていない場合は、defaultdir が定義するパスが使用されます。絶対パスを使用すると defaultdir を無効にすることができます。

ファイルを識別しやすくするために、ファイルを別々のディレクトリに入れておきます。次に、必要な変更の例を示します。

```
% cat /etc/nfs/nfslog.conf
#ident  "@(#)nfslog.conf        1.5     99/02/21 SMI"
#
  .
  .
# NFS server log configuration file.
#

global  defaultdir=/var/nfs \
        log=nfslog fhtable=fhtable buffer=nfslog_workbuffer

publicftp log=logs/nfslog fhtable=fh/fhtables buffer=buffers/workbuffer
```

この例では、log=publicftp と共有するファイルはすべて、次の値を使用します。

- デフォルトのディレクトリは /var/nfs である
- ログファイルは、/var/nfs/logs/nfslog* に保存される
- ファイルハンドルパスデータベーステーブルは、/var/nfs/fh/fhtables に保存される
- バッファーファイルは、/var/nfs/buffers/workbuffer に保存される

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 かマルチユーザー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 デーモンおよび nfsd デーモンは、NFS サーバーであるシステム上で実行されます。サーバー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 デーモンおよび statd デーモンは、NFS クライアントおよび NFS サーバー上で実行され、NFS のファイルロックをサポートします。

automountd

このデーモンは autofs サービスからのマウントおよびデマウント要求を処理します。このコマンドの構文は次のとおりです。

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

このコマンドは、次のように動作します。

- `-T` は、トレースを有効にする
- `-n` は、すべての autofs ノード上で、ブラウズを無効にする
- `-v` は、コンソールへのすべての状態メッセージを記録する
- `-D` は、*name= value - name* によって示されたオートマウントマップ変数の値を *value* に置き換える

自動マウントマップのデフォルト値は `/etc/auto_master` です。障害追跡には `-T` オプションを使用してください。

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。`lockd` デーモンは、ネットワークロックマネージャ (NLM) プロトコルについて、クライアントとサーバー間の RPC 接続を管理します。通常は、パラメータを指定しないで起動します。使用できるオプションは 3 つあります (`lockd(1M)` のマニュアルページを参照)。これらのオプションは、コマンド行からも、`/etc/default/nfs` への任意の文字列を編集することによっても使用することができます。 `/etc/default/nfs` を変更すると、システムを再起動してもその変更は維持されます。 この機能は、Solaris 9 リリースでのみサポートされています。その他の Solaris リリースでこれらの変更を維持するには、`/etc/init.d/nfs.client` を変更します。

`/etc/default/nfs` に `LOCKD_GRACE_PERIOD=`*graceperiod* パラメータを追加すると、クライアントがサーバーのリブート後にロックを再要求する秒数を選択できます。NFS サーバーはこの秒数の間、それまでのロックの再要求処理しか実行しません。サービスに対する他の要求は、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 サーバーの応答性に関係するため、NFS サーバー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ことにより、NFS クライアントは、サーバーのリブート後に、より迅速に処理を再開できます。ただし、この値を小さくすると、クライアントがすべてのロックを復元できない可能性が増します。デーモンに `-g` *graceperiod* オプションを指定して開始すると、コマンド行から同じ動作を実行することができます。

`/etc/default/nfs` に `LOCKD_RETRANSMIT_TIMEOUT=`*timeout* パラメータを追加すると、ロックの要求をリモートサーバーに再転送するまでの秒数を選択できます。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、トラフィック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ーの負荷が増す可能性があります。デーモンに `-t` *timeout* オプションを指定して開始すると、コマンド行から同じパラメータを使用できます。

`/etc/default/nfs` に `LOCKD_SERVERS=`*nthreads* パラメータを追加すると、サーバーが同時に処理できる各接続ごとのスレッドの最大数を指定できます。 この値は、NFS サーバー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 を使用する各 NFS クライアントは、NFS サーバーとの間で 1 つの接続を使用するため、各クライアントは、サーバー上で、最大 20 のスレッドを同時に使用することができます。UDP を使用するすべての NFS クライアントは、NFS サーバーと 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。各 UDP クライアントには、少なくとも 2 つのスレッドを許可します。ただし、この数は、クライアントの負荷により違います。そのため、クライアントごとに 2 つのスレッドを許可しても、十分ではない場合があります。多くのスレッドを使用する場合の不利な点は、これらのスレッドを使用すると、NFS サーバー上で使用するメモリーの容量が増えるという点です。ただし、スレッドを使用しない場合は、*nthreads* の値を増やしても影響がありません。デーモンに `nthreads` オプションを指定して開始すると、コマンド行から同じパラメータを使用できます。

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 (リモート手続き呼び出し) サーバーです。`mountd` デーモンは、`/etc/dfs/sharetab` を調べて、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できる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 (`mountd(1M)` のマニュアルページを参照してください)。

`-v` オプションは、コマンドを詳細形式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 サーバーが決定するたびに、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。このオプションを指定しても、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を指定できます。オプションをすべて確認するには `nfsd(1M)` のマニュアルページを参照してください。これらのオプションは、コマンド行からも、`/etc/default/nfs` への文字列を編集することによっても使用することができます。 `/etc/default/nfs` を変更すると、システムをリブート

してもその変更は維持されます。 この機能は、Solaris 9 リリースでのみサポートされています。他のリリースで、これらの変更を維持するには、/etc/init.d/nfs.server を変更します。

/etc/default/nfs に NFSD_LISTEN_BACKLOG=*length* パラメータを追加すると、接続型トランスポートを使用した NFS/TCP の接続キューの長さを設定できます。デフォルト値は 32 エントリです。nfsd に -l オプションを指定して開始すると、コマンド行から同じ項目を選択できます。

/etc/default/nfs に NFSD_MAX_CONNECTIONS=#*_conn* パラメータを追加すると、接続型トランスポートごとの最大接続数を選択できます。#*_conn* のデフォルト値はありません。コマンド行から -c #*_conn* オプションを指定してデーモンを開始すると、同じパラメータを使用できます。

/etc/default/nfs に NFSD_SERVER=*nservers* パラメータを追加すると、サーバーが一度に処理する要求の最大数を選択できます。 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。コマンド行から nservers オプションを指定してnfsd を開始すると、同じように最大数を選択できます。

以前のバージョンの nfsd デーモンとは異なり、現在のバージョンの 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。

nfslogd

このデーモンは実行された処理のログ機能を提供します。NFS オペレーションは、/etc/default/nfslogd で定義した構成オプションに基づいてサーバーのログに記録されます。NFS サーバーのログ機能がオンになると、選択されたファイルシステム上でのすべての RPC 操作の記録がカーネルによりバッファーファイルに書き込まれます。次に nfslogd がこれらの要求を後処理します。ログインおよび IP アドレスへの UID をホスト名に割り当てやすくするために、ネームサービススイッチが使用されます。識別されたネームサービスで一致するものが見つからない場合は、その番号が記録されます。

パス名へのファイルハンドルの割り当ても nfslogd により行われます。このデーモンは、ファイルハンドルパスマッピングテーブル内でこれらの割り当てを追跡します。/etc/nfs/nfslogd で識別される各タグについて 1 つのマッピングテーブルが存在します。後処理の後に、レコードが ASCII ログファイルに書き込まれます。

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。statd デーモンは、NFS サーバーにロックを保持するクライアントを追跡します。サーバーがクラッシュした場合は、サーバーのリブート中に、サーバー側 statd がクライアント側 statd にコンタクトします。次にクライアント側 statd は、サー

バー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 はサーバー側 statd にそのことを伝えるので、サーバー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。詳細は、statd(1M) のマニュアルページを参照してください。

Solaris 7 で、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 サーバーと異なるドメインに存在する場合に、このファイルのネーミングが原因となり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、次のようになります。

```
# ls -l /var/statmon/sm
lrwxrwxrwx   1 daemon          11 Apr 29 16:32 ipv4.192.9.200.1 -> myhost
lrwxrwxrwx   1 daemon          11 Apr 29 16:32 ipv6.fec0::56:a00:20ff:feb9:2734 -> v6host
--w-------   1 daemon          11 Apr 29 16:32 myhost
--w-------   1 daemon          11 Apr 29 16:32 v6host
```

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

次のコマンドは、root として実行しなければ、十分な効果がでません。しかし情報の要求は、すべてのユーザーが行うことができます。

- 205 ページの「automount」
- 205 ページの「clear_locks」
- 206 ページの「mount」
- 210 ページの「mountall」
- 218 ページの「setmnt」
- 211 ページの「share」
- 217 ページの「shareall」
- 217 ページの「showmount」
- 209 ページの「umount」
- 210 ページの「umountall」
- 216 ページの「unshare」
- 217 ページの「unshareall」

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、オートマスターファイル内の情報を各マウントポイントに関連付けます。このコマンドの構文は次のとおりです。

```
automount [ -t duration ] [ - v ]
```

`-t` *duration* はファイルシステムがマウントされた状態でいる時間 (秒) で、`-v` は詳細形式モードを選択します。詳細形式モードでこのコマンドを実行すると障害追跡が容易になります。

継続時間の値は、特に設定しないと 5 分に設定されます。ほとんどの場合この時間は適切な値ですが、自動マウントされたファイルシステムを多く持つシステムではこの時間を長くする必要がある場合があります。 特に、サーバーを多くのユーザーが使用中の場合は、自動マウントされたファイルシステムを 5 分ごとにチェックするのは能率的でない場合があります。autofs ファイルシステムは 1800 秒 (30 分) ごとにチェックする方が適しています。5 分おきにファイルシステムのマウントを解除しないと、`/etc/mnttab` が大きくなることがあります。`df` が `/etc/mnttab` にある各エントリをチェックした時の出力を減らすには、`-F` オプション (`df(1M)` マニュアルページを参照) または `egrep` を使用して、`df` の出力にフィルタをかけます。

検討すべき他の要因に、この継続時間を調節するとオートマウンタマップへの変更が反映される速さを変更できるということがあります。変更はファイルシステムがアンマウントされるまでは見ることができません。オートマウンタマップの変更方法については、170 ページの「マップの修正」を参照してください。

clear_locks

このコマンドを使用すると、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、スーパーユーザーでなければなりません。NFS サーバーから、特定のクライアントに対するロックを解除できます。また、NFS クライアントから、特定のサーバーにおけるそのクライアントに対するロックを解除できます。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。このオプションは、ロックを作成した NFS クライアントで実行する必要があります。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 サーバーから解除されます。

```
# 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またはリモートで、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。引数を指定しないで `mount` を使用すると、現在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。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。NFS ファイルシステムの場合は、`mount_nfs(1M)` です。UFS ファイルシステムの場合は、`mount_ufs(1M)` を参照してください。

Solaris 7 で、`server:/pathname` という標準の構文の代わりに NFS URL を使用して NFS サーバー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。詳細は、160 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 ファイルシステムでの mount 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。`-fg` オプションを指定すると、マウントが完了するまで、次の処理は行われません。`-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。

forcedirectio
: このオプションは大規模ファイル上で連続した読み取りをする際にパフォーマンスを向上させます。データは直接ユーザーファイルにコピーされ、クライアント上のカーネル内ではキャッシュへの書き込みは行われません。この機能はデフォルトではオフです。

largefiles
: このオプションを使用すると、Solaris 2.6 が実行されているサーバーに置かれた 2G 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。大規模ファイルにアクセスできるかどうかは、サーバー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、Solaris 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 プロトコルを使用したマウントで `largefiles` 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
: この UFS マウント用のオプションを指定すると、ファイルシステム上に大規模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 (`mount_ufs (1M)` のマニュアルページを参照)。大規模ファイルの存在は NFS サーバー上でのみ制御できるため、NFS マウントを使用している場合は、`nolargefiles` オプションを指定できません。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 サーバー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 サーバー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、ユーザー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、ユーザー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、マウント時に使用される認証メカニズムを指定します。*mode* の値は、表 16–2 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 **16–2** NFS セキュリティモード

モード	選択した認証サービス
`krb5`	Kerberos バージョン 5
`krb5i`	Kerberos バージョン 5 で完成性を指定

表 **16–2** NFS セキュリティモード　(続き)

モード	選択した認証サービス
krb5i	Kerberos バージョン 5 で機密性を指定
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、サーバー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、サーバー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恐れがあるためです。アプリケーションが戻り値を確認する場合は、soft が使用されていると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く判断できず、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用できなくなると、そのファイルシステムを使用する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

次のコマンドのどちらも、bee サーバーから NFS ファイルシステムを読み取り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、次のように /usr/man がすでにマウントされていても bee サーバー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、クライアント側フェイルオーバー機能が使用されています。

```
# 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、リスト内のサーバー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 のサーバーを混在させないでください。autofs を実行するときは、これらのサーバーを混在させることができます。autofs により、バージョン 2 またはバージョン 3 のサーバーの最適な組み合わせが自動的に選択されます。

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 オプションを使用することによって 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。

-f オプションを指定すると、使用中のファイルシステムのマウントが解除されます。このオプションを使用して、マウントできないファイルシステムのマウントを試みたためにハングしたクライアントを復帰させることができます。

注意 – ファイルシステムのアンマウントを強制的に解除すると、ファイルへの書き込み中だった場合には、データを損失することがあります。

umount コマンドの使用

次の例では、/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
```

次の例では、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。このコマンドを実行すると、次の操作を実行することができます。

- - F *FSType* オプションを使用して、ファイルシステムのタイプを選択する
- -r オプションを使用して、ファイルシステムテーブル中にリストされたリモートファイルシステムをすべて選択する
- -l オプションを使用して、ローカルファイルシステムをすべて選択する

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。詳細は、mountall(1M) のマニュアルページを参照してください。

mountall コマンドの使用

次の 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* コマンドを使用する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 と同時に指定できません。

umountall コマンドの使用

次のコマンドでは、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
```

次のコマンドでは、bee サーバー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 サーバー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 サーバーが動作していないと、share コマンドは使用できません。NFS サーバーソフトウェアは、/etc/dfs/dfstab に項目がある場合、起動の途中で自動的に起動されます。NFS サーバーソフトウェアが動作していないと、このコマンドはエラーを報告しません。そのため、ソフトウェアが動作していることを確認する必要があります。

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 および /usr/local が同じディスクスライスにある場合は、/usr または /usr/local を共有できます。ただし、異なる共有オプションを指定してこれらのディレクトリを共有するには、/usr/local を別のディスクスライスに移動する必要があります。

注 – 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、読み取り専用で共有しているファイルシステムに、読み取り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。読み取りと書き込みが設定されているファイルシステムを、読み取り専用で共有しているファイルシステムとは別のパーティションまたはディスクスライスに配置すると、より安全にこれらのファイルシステムを使用できます。

非ファイルシステム用 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次に示します。

rw | ro
: *pathname* に指定したファイルシステムを、読み取りと書き込みの両方が可能な状態で共有するか、読み取り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、リストに記述されているクライアントに対してだけ、読み取り書き込みの両方が可能な状態で共有されます。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、214 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 用 share オプション

NFS ファイルシステムで指定できるオプションは、次のとおりです。

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 サーバーが 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小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 人のユーザー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。ユーザーに与えるアクセス権は、既存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) の詳細は、『*Solaris* のシステム管理 (セキュリティサービス)』の「ファイルのセキュリティの適用手順」を参照してください。

注 – アクセス制御リスト (ACL) を使用するには、クライアントとサーバーが、NFS バージョン 3 プロトコルおよび 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ている必要があります。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ーのユーザー 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ー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、これのオプションを指定すると、認証されていないユーザー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` オプションを使用してください。

index=*filename*

`-index=`*filename* オプションを使用すると、ユーザーが 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` ファイルに、次のようなエントリがあるとします。

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

このとき、次の 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
```

log=*tag*

このオプションは、ファイルシステム用の NFS サーバーレコード構成情報の入った `/etc/nfs/nfslog.conf` 内のタグを指定します。NFS サーバーログ機能を使用可能にするにはこのオプションを選択する必要があります。

nosuid

このオプションを使用すると、`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 台のサーバーにつき 1 つのファイルシステムだけです。

root=*accesslist*
サーバーが、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、サーバー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、214 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=` オプション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、すべてのユーザーがユーザー `nobody` にマップされます。

window=*value*
value は、NFS サーバー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

Solaris 2.6 より前の Solaris では、`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用することなく、サーバー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。

次のコマンドでは、`rose` と `lilac` では読み取り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、読み取りだけが許可されます。

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

次の例では、`eng` ネットグループのすべてのホストで読み取りだけができるようになります。`rose` クライアントでは、読み取り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。

複数のクライアントが 1 つのファイルシステムを共有するには、同じ行にすべてのオプションを入力する必要があります。同じオブジェクトに対して share コマンドを何度も実行しても、最後に実行されたコマンドだけが有効になります。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取り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
```

複数の認証メカニズム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) 認証を使用すれ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。これらのオプションを指定すると、tulip および lilac は、DH 認証を使用していない場合でも、ファイルシステムを読み取り専用でマウントすることができます。ただし、ホスト名が eng ネットグループに含まれている必要があります。

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

デフォルトのセキュリティモードは UNIX 認証ですが、-sec オプションを使用している場合、この UNIX 認証は含まれなくなります。そのため、UNIX 認証を他の認証メカニズムとともに使用する場合は、-sec=sys オプションを指定する必要があります。

実際のドメイン名の前にドットを付けると、アクセスリスト中で DNS ドメイン名を使用できます。ドットは、その後の文字列が完全指定の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.example.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.example.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指定の名前になるので、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を使用できます。この文字は、ネットワーク名をネットグループ名や完全指定の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。次のエントリは、サブネット 129.144 が eng ネットワークと識別されている場合、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 番目と 3 番目のエントリは、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:

```
# 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用されることを表します。CIDR の詳細は、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のエントリ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.example.com /export/share/man
```

この例では、eng.example.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 コマンドを使って dfstab ファイルを使って共有していたファイルシステムの共有を解除する場合は、注意が必要です。一度実行レベル 3 を終了し再度実行レベル 3 を実行すると、ファイルシステムは再度共有されます。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

次のコマンドでは、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

次のコマンドでは、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

このコマンドは、次のいずれかを表示します。

- NFS サーバーから共有している、リモートマウントされたファイルシステムを持つすべてのクライアント
- クライアントによってマウントされたファイルシステムのみ
- 共有されたファイルシステムおよびクライアントのアクセス情報

コマンドは、次のような構文になります。

showmount [-ade] [*hostname*]

`-a`	すべてのリモートマウントのリストを出力する。各エントリには、クライアント名とディレクトリが含まれる
`-d`	クライアントがリモートマウントしたディレクトリのリストを表示する
`-e`	共有されているファイル、またはエクスポートされたファイルのリストを表示する
hostname	表示する情報の取得元 NFS サーバーを指定する

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

次のコマンドでは、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
```

次のコマンドでは、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
```

次のコマンドでは、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、このコマンドは、システムのブート時に自動的に実行されるため、手動で実行する必要はありません。

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。

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。このコマンドの構文は次のとおりです。

```
nfsstat [ -cmnrsz ]
```

`-c`	クライアント側の情報を表示する
`-m`	NFS マウントされた各ファイルシステムの統計を表示する
`-n`	クライアント側とサーバー側の両方で、NFS の情報が表示されるように指定する
`-r`	RPC 統計を表示する
`-s`	サーバー側の情報を表示する
`-z`	統計をゼロに設定するように指定する

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、サーバー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、以前のパフォーマンスの有効な記録となります。

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%
```

このリストは、NFS サーバーの統計の例です。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、`badcalls` または `calls` の平均値、および各週の calls の数がわかるので、問題を特定するのに役立ちます。`badcalls` 値は、クライアントからの不良メッセージ数を示しています。この値は、ネットワークのハードウェアに問題が発生したことを示す場合があります。

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 の統計で注意が必要なのは、`setattr`、`write`、`create`、`remove`、`rename`、`link`、`symlink`、`mkdir`、および `rmdir` です。NFS バージョン 3 では、`commit` の値に特に注意します。ある NFS サーバーの `commit` レベルが、それと同等のサーバーと比較して高い場合は、NFS クライアントに十分なメモリーがあるかどうかを確認してください。サーバーの 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、各プロセスのスタックトレースが表示されます。`pstack` コマンドは、必ずプロセスの所有者、または `root` が実行してください。`pstack` を使用して、プロセスがハングした場所を判断します。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を参照)。

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, effff14, 0, 0, ffffffff, 400) + 3c8
 00010fb0 _start   (0, 0, 0, 0, 0, 0) + 5c
```

たとえば、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わかった場合、そのプロセスはハングしている可能性があります。187 ページの「NFS サービスを再起動する方法」 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、183 ページの「NFS の障害追跡の手順」 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を参照)。以下は、このコマンドで使用できるオプションの構文です。

`rpcinfo [ -m | -s ] [` *hostname* `]`

`rpcinfo -T` *transport hostname* `[` *progname* `]`

`rpcinfo [ -t | -u ] [` *hostname* `] [` *progname* `]`

`-m`	`rpcbind` 処理の統計テーブルを表示する
`-s`	登録されているすべての RPC プログラムを簡易リストで表示する
`-T`	特定のトランスポートまたはプロトコルを使用するサービスの情報を表示する
`-t`	TCP を使用する RPC プログラムを検索する
`-u`	UDP を使用する RPC プログラムを検索する
transport	サービスに使用するトランスポートまたはプロトコルを選択する
hostname	必要な情報の取得元のサーバーのホスト名を選択する
progname	情報の取得対象の RPC プログラムを選択する

hostname を指定しないと、ローカルホスト名が使用されます。*progname* の代わりに RPC プログラム番号が使用できますが、ユーザー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` オプションを使用できます。

このコマンドを実行すると、次の項目を含むデータを生成することができます。

- 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 サービス名
- RPC サービスの所有者

rpcinfo コマンドの使用

次の例では、サーバーで実行されている 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     udp6,tcp6,udp,tcp,ticlts,ticotsord,ticots rpcbind     superuser
    100001  4,3,2     ticlts,udp,udp6                    rstatd      superuser
    100002  3,2       ticots,ticotsord,tcp,tcp6,ticlts,udp,udp6 rusersd     superuser
    100003  3,2       tcp,udp,tcp6,udp6                  nfs         superuser
    100005  3,2,1     ticots,ticotsord,tcp,tcp6,ticlts,udp,udp6 mountd      superuser
    100007  1,2,3     ticots,ticotsord,ticlts,tcp,udp,tcp6,udp6 ypbind      superuser
    100008  1         ticlts,udp,udp6                    walld       superuser
    100011  1         ticlts,udp,udp6                    rquotad     superuser
    100012  1         ticlts,udp,udp6                    sprayd      superuser
    100021  4,3,2,1   tcp,udp,tcp6,udp6                  nlockmgr    superuser
    100024  1         ticots,ticotsord,ticlts,tcp,udp,tcp6,udp6 status      superuser
    100029  3,2,1     ticots,ticotsord,ticlts            keyserv     superuser
    100068  5         tcp,udp                            cmsd        superuser
    100083  1         tcp,tcp6                           ttdbserverd superuser
    100099  3         ticotsord                          autofs      superuser
    100133  1         ticots,ticotsord,ticlts,tcp,udp,tcp6,udp6 -           superuser
    100134  1         ticotsord                          tokenring   superuser
    100155  1         ticots,ticotsord,tcp,tcp6          smserverd   superuser
    100221  1         tcp,tcp6                           -           superuser
    100227  3,2       tcp,udp,tcp6,udp6                  nfs_acl     superuser
    100229  1         tcp,tcp6                           metad       superuser
    100230  1         tcp,tcp6                           metamhd     superuser
    100231  1         ticots,ticotsord,ticlts            -           superuser
    100232  10        udp,udp6                           sadmind     superuser
    100234  1         ticotsord                          gssd        superuser
    100235  1         tcp,tcp6                           -           superuser
    100242  1         tcp,tcp6                           metamedd    superuser
    100249  1         ticots,ticotsord,ticlts,tcp,udp,tcp6,udp6 -           superuser
    300326  4         tcp,tcp6                           -           superuser
    300598  1         ticots,ticotsord,ticlts,tcp,udp,tcp6,udp6 -           superuser
    390113  1         tcp                                -           unknown
```

```
 805306368  1         ticots,ticotsord,ticlts,tcp,udp,tcp6,udp6 -           superuser
1289637086  1,5       tcp                                -           26069
```

次の例では、サーバーの特定トランスポートを選択して、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
% 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 で実行されている mountd サービスをチェックしています。2 番目の例では、UDP で実行されている 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。snoop コマンドは、root で実行する必要があります。このコマンドは、クライアントとサーバーの両方で、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を参照)。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*	ローカルネットワークのインタフェースを指定する
-o *filename*	受信したすべてのパケットを指定したファイルに保存する
hostname	特定のホストが送受信したパケットを表示する

-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ーで特に有効です。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、データを簡単に参照することができます。

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。truss コマンドは、必ずプロセスの所有者、または root が実行してください。 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を参照)。以下で、このコマンドの構文を説明します。

`truss [ -t` *syscall* `] -p` *pid*

`-t` *syscall*	追跡するシステムコールを選択する
`-p` *pid*	追跡するプロセスの PID を指定する

syscall には、追跡するシステムコールをコ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。187 ページの「NFS サービスを再起動する方法」 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、183 ページの「NFS の障害追跡の手順」 を参照してください。

NFS サービスのしくみ

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 サーバーがサポートしているクライアントが NFS バージョン 3 を使用し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ー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 が使用されます。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 が使用されます。

ネゴシエーションによって決まった値は、`mount` コマンドに `-vers` オプションを使用することで変更できます (`mount_nfs(1M)` のマニュアルページ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、ユーザー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トされます。デフォルトでは、クライアントとサーバー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用されます。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 から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 を使用する NFS バージョン 3 プロトコルの方が、TCP を使用する 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 (`mount_nfs(1M)` のマニュアルページを参照)。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ーの間でデータを転送するときに使用されるバッファーのサイズです。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ーサイズは 32K 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 を使用しているシステムとはネゴシエートされません。このとき、ファイル転送サイズの上限は 8K 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` オプションを使用すると、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 サーバーが大きなファイル転送サイズに設定されている場合は、転送サイズを大きくすることができます。

ファイルシステムがどのようにマウントされるか

クライアントがサーバー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ー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ー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ーから `/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 サーバーの 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、サーバー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、サーバー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ー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っ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 オプションを使っ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 サーバー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 プロトコルを使ってマウントします。

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。

クライアント側フェイルオーバー機能

クライアント側のフェイルオーバー (障害時回避) 機能を使用すると、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ー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ー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ーのクラッシュ、サーバーの過負荷、ネットワーク障害が考えられます。通常、このような場合のフェイルオーバー機能はユーザーにはわかりません。設定が行われていれば、フェイルオーバー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。

フェイルオーバー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フェイルオーバー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、228 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。フェイルオーバー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、フェイルオーバー機能には使用できません。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。この情報はフェイルオーバー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フェイルオーバー機能と CasheFS のどちらか片方の機能しか使用できません。

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。原則として、各サブネットに 1 台のサーバーを設置するのではなく、複数のサブネットをサポートしているサーバーを 2 台以上設置するのが理想的です。フェイルオーバー処理の際にはリストにある各サーバーが確認されます。そのため、サーバーの台数を増やすと、それぞれのマウント処理が遅くなります。

フェイルオーバー機能に関する用語

フェイルオーバー機能のプロセスを完全に理解するには、次の 2 つの用語を理解する必要があります。

- フェイルオーバー – 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ーのリストから、1 つのサーバーを選択するプロセス。通常、ソートされたリストの順番を元に、次のサーバーが応答するならばそのサーバーが使用される
- 再マッピング – 新しいサーバーを使用すること。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のそれぞれのパス名を格納する。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ー上のファイルを検出する

複製されたファイルシステムとは

フェイルオーバー機能に関して、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。ファイルサイズまたは vnode タイプが異なると再マッピングは失敗し、元のサーバー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メカニズムを使います。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

フェイルオーバー機能と 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。サーバー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、サーバー側でファイルをロックする必要はありません。

大規模ファイル

Solaris 2.6 およびその互換バージョンでは、2G 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、2G 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 157 ページの「NFS サーバー上で大規模ファイルを無効にする方法」 を参照してください。

`-largefiles` オプションを使ってサーバー上のファイルシステムをマウントする場合、大規模ファイルにアクセスするために Solaris 2.6 NFS クライアントを変更する必要はありません。ただし、Solaris 2.6 のコマンドすべてで大規模ファイルを扱えるわけではありません。大規模ファイルを扱えるコマンドについては、`largefile(5)` を参照してください。大規模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、大規模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 プロトコルを使用することはできますが、大規模ファイルを扱う機能は含まれていません。

NFS サーバーログ機能のしくみ

NFS サーバーログ機能は NFS の読み取りと書き込み、およびこのファイルシステムを変更する操作の記録を提供します。このデータは情報へのアクセスを追跡するのに利用できます。さらに、この記録は、情報へのアクセスを測定する定量的な方法を提供します。

ログ機能が有効になっているファイルシステムにアクセスすると、カーネルが raw データをバッファーファイルに書き込みます。このデータには、次の内容が含まれています。

- タイムスタンプ
- クライアントの IP アドレス
- 要求者の UID
- アクセスされているファイルまたはディレクトリオブジェクトのファイルハンドル
- 発生した処理のタイプ

`nfslogd` デーモンはこの raw データを、ログファイルに保存される ASCII レコードに変換します。使用可能なネームサービス機能が一致しているものを見つけると、その変換中に IP アドレスはホスト名に変更され、UID はログインに変更されます。ファイルハンドルはパス名にも変換されます。デーモンはファイルハンドルを追跡し、情報を別のファイルハンドルパステーブルに保存して、変換を完了します。このようにすると、ファイルハンドルにアクセスされるたびに、パスを識別し直す必要がなくなります。`nfslogd` をオフにするとファイルハンドルパステーブルのマッピングが変更されなくなるため、デーモンは常に実行させておく必要があります。

WebNFS サービスのしくみ

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、サーバー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プロセスが高速化します。

デフォルトでは、NFS サーバー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、サーバー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、

WebNFS サーバー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用することにより、WebNFS サーバー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 接続を介して、複数のファイルを同時にダウンロードすることができます。このようにして接続すると、サーバーに複数の接続を設定することによる負荷をかけることなく、すばやくアクセスすることができます。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 つしか使用しないため、サーバー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、サーバー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
```

WebNFS セキュリティネゴシエーション機能のしくみ

Solaris 8 リリースから、WebNFS クライアントが WebNFS サーバーと、選択されたセキュリティメカニズムについてネゴシエーションできるようにする新しいプロトコルがあります。この新しいプロトコルは、セキュリティネゴシエーションマルチコンポーネントルックアップを使用しています。これは、WebNFS プロトコルの以前のバージョンで使用されていたマルチコンポーネントルックアップの拡張版です。

WebNFS クライアントは、公共ファイルハンドルを使って通常のマルチコンポーネントルックアップ要求を行うことにより、このプロセスを開始します。このクライアントには、サーバーがどのようにしてこのパスを保護しているかについての知識がないため、デフォルトのセキュリティメカニズムが使用されます。デフォルトのセキュリティメカニズムでは不十分な場合は、サーバーは `AUTH_TOOWEAK` エラーを返します。このメッセージは、そのデフォルトメカニズムが有効ではなく、クライアントはより強力なメカニズムを使用する必要があることを意味しています。

クライアントは、`AUTH_TOOWEAK` エラーを受信すると、サーバーに対してどのセキュリティメカニズムが必要か決定するように要求します。この要求が成功すると、サーバーは、指定されたパスに必要なセキュリティメカニズムの配列を返します。このセキュリティメカニズムの配列のサイズによっては、クライアントは完全な配列を得るためにさらに要求を出さなければならない場合があります。サーバーが WebNFS セキュリティネゴシエーションをサポートしていない場合は、この要求は失敗します。

要求が成功すると、WebNFS クライアントは、配列からサポートしている最初のセキュリティメカニズムを選択します。その後、クライアントは、選択したセキュリティメカニズムを使用して、通常のマルチコンポーネントルックアップ要求を発行し、ファイルハンドルを獲得します。この後に続くすべての NFS 要求は、選択されたセキュリティメカニズムとファイルハンドルを使って出されます。

Web ブラウザの使用と比較した場合の WebNFS の制約

HTTP を使用する 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 サーバー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ア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アが起動されます。ブラウザは、NFS URL が使用されている場合、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。これは、WebNFS ソフトウェアが、ファイルの内容からファイルタイプを判別しないため、WebNFS ソフトウェアは、ファイル名の拡張子だけでそのファイルタイプを判別します。
- NFS によるブラウズでは、サーバー側のイメージマップ (クリック可能なイメージ) は使用できません。しかしクライアント側のイメージマップ (クリック可能なイメージ) は、場所とともに URL が定義されているため使用できます。文書サーバー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) 認証を使用してきましたが、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 認証を使用している場合、NFS サーバーは、要求をしたユーザーではなくコンピュータを認証して、ファイル要求を認証します。そのため、クライアントユーザーは、`su` を実行してファイルの所有者を装ったりすることができます。DH 認証では、NFS サーバーはユーザーを認証するため、このような操作が困難になります。

スーパーユーザー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、誰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もっとも危険な攻撃は、有効なパ

ケットを生成したり、または「対話」を記録し後で再生することによってユーザー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。許可を持つユーザーを装うことなく、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。ネットワーク上でやりとりされるデータを暗号化すると、機密情報のプライバシを保護できます。

ネットワークのセキュリティ問題に対す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リモート手続き呼び出し (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 システムといいます。

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 (すべてのユーザーが 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ーを認証します。データ暗号化規格 (DES) 認証でも、同じ認証処理が実行されます。ユーザーは、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。ユーザー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誰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す。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ーに送信します。クライアントはサーバーの資格をすでに知っているため、サーバー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、および KERB 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、`root` ユーザーは `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他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ってネットワーク上のユーザーとコンピュータの両方を認証します。DES は、標準の暗号化メカニズムです。Diffie-Hellman 公開鍵暗号手法は、2 つの鍵、つまり公開鍵と秘密鍵を持つ暗号方式です。 公開鍵と秘密鍵は名前空間に格納されます。NIS では、これらのキーは publickey マップに保存されています。これらのマップにはすべての認証の候補ユーザーの公開鍵と秘密鍵が入っています。このマップの設定方法については、『*Solaris* のシステム管理 *(*ネーミングとディレクトリサービス *: DNS*、*NIS*、*LDAP* 編*)*』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化して、自分の時刻と照合します。タイムスタンプは DES を使用して暗号化されます。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ー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、サーバーの時間を使っ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ーに時間を要求し、自分のクロックとサーバー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、この差を使ってクライアントのクロックを補正します。サーバー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ーのクロック同期がずれた場合、DH 認証システムはサーバーとの間で再び同期をとります。

クライアントとサーバーは、ランダムな対話鍵 (セッションキーとも呼ばれる) 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) を使っ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ー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 サポートは、現在では Secure RPC の一部としては供給されていませんが、Solaris 9 リリースには、サーバー側とクライアント側の実装が含まれています。Solaris 9 に実装されている Kerberos 認証については、『*Solaris* のシステム管理 *(*セキュリティサービス*)*』の「SEAM について」を参照してください。

NFS での Secure RPC の使用

Secure RPC を使用する場合は、次の点に注意してください。

- サーバー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。リブート中の重要な処理は、通常 `root` として実行されます。そのため、`root` の秘密鍵を別に保存していればこれらのプロセスを実行できますが、その秘密鍵を復号化するパスワードを入力することはできません。`keylogin -r` を使用すると `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、シングルユーザーモードで起動し、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。ブートサーバー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ーが起動した後だけです。そうでないと、ブートサーバーからの応答を認証することができません。このような制限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用し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。つまり、ネットワークを通じてブートサーバーにポーリングすれば、不正なブートサーバーの場所がわかります。
- 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、ユーザーが所有者である 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、ブート後 `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` を使用して) 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ーに渡され、キーサーバーがその秘密鍵を格納したためです。このプロセスが問題になるのは、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 環境を使用して、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ー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` ファイルまたは `~/.rhosts` ファイルに、パスワードを要求するように設定されていない場合は、ログインが成功します。ただし、ローカルで認証されていないため、ユーザーは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。

autofs マップ

autofs は 3 種類のマップを使用します。

- マスターマップ
- 直接マップ
- 間接マップ

autofs マスターマップ

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。このマップは、すべてのマップを指定するマスターリストであり、autofs が参照します。auto_master ファイルの内容の例を次に示します。

例 **16–1** /etc/auto_master ファイルの例

```
# 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。マスターマップ /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、ディレクトリの完全 (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 はこのディレクトリを作成します。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 (`/etc/nsswitch.conf`) で指定される検索によりマウント情報を検索します。また、`/net` および `/xfn` には、特別なマップを使用します。詳細は、236 ページの「マウントポイント `/net`」 および、237 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* は省略できます。*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コンマで区切って並べます。特定のファイルシステムのマウントオプションについては、各ファイルシステムについてのマニュアルページ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)` のマニュアルページを参照してください。 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。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。

注 – 2 つのエントリで同じマウントポイントが使用されるときは、1 番目のエントリは `automount` コマンドが使用します。2 番目のエントリは無視されます。

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` テーブルを使用して、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、特別なマップ `-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、`-hosts` データベースにあるコンピュータ `gumbo` が、ファイルシステムのどれかをエクスポートする場合は、次のコマンドを入力すると、現在の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ーが使用できるサーバー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 を使用したアクセスでは、サーバー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ーに移動すると、そのパスは使用できなくなります。このような場合は /net を使用しないで、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ーのエクスポートリストを調べます。サーバー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 はそのサーバー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、それがサーバー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により、NFS 名前空間を通して共有しているリソースで、autofs のディレクトリ構造を使用できます。FNS の詳細は、『*Solaris* のシステム管理 *(*ネーミングとディレクトリサービス *: FNS*、*NIS+* 編*)*』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ー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パス名があり、明示的に関係を示します。次に一般的な /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、次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。特定のファイルシステムのマウントオプションについては、各ファイルシステムについてのマニュアルページを参照してください。 たとえば、CacheFS 固有のマウントオプションについては、`mount_cachefs(1M)` のマニュアルページを参照してください。
location	*location* はファイルシステムの位置を示し、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* にオートマウントされたマウントポイントを含めることはできません。 *pathname* は、実際のファイルシステムの絶対パスでなければなりません。 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* として指定する必要があります。

マスターマップと同様、# 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。

すべてのマップにおいて、直接マップ内のエントリは、`/etc/vfstab` 内の対応するエントリにもっともよく似ています。`/etc/vfstab` のエントリは、次のようになっているとします。

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

直接マップ内では、同じエントリが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。オートマウンタマップに追加されたどのオプションも、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、他にも重要な機能があります。245 ページの「autofs がクライアント用のもっと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 /–

例 16–1 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、名前付きマップ内で指定したマウントポイントを使用します。直接マップ内では、キー、つまりマウントポイントは完全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、キーの置換値を使ってクライアント上のマウントポイントとサーバー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次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。特定のファイルシステムのマウントオプションについては、各ファイルシステムについてのマニュアルページ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) のマニュアルページを参照してください。
location	*location* はファイルシステムの位置を示し、*server*:*pathname* の形式により (1 つまたは複数) 指定します。 注 – *pathname* にオートマウントされたマウントポイントを含めることはできません。 *pathname* は、実際のファイルシステムの絶対パスでなければなりません。 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 として指定する必要があります。

マスターマップと同様、# 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。例 16–1 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。通常、 auto_home マップには、次のパスが含まれています。

```
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。パスワードデータベースに、ユーザー linda のホームディレクトリが /home/linda であることを示すエントリがあるとします。このユーザーがコンピュータ oak にログインするたびに、autofs は、コンピュータ peach にあるディレクトリ /export/home/linda 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。ユーザー 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ー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、ユーザー 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では、オプションの連結はされません。オートマウンタマップに追加されたどのオプションも、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、他のいずれか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、Linda が自分のファイルにアクセスするには、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 マスターサーバーで変更を行い、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ー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。以降にそのマウントを参照すると、その要求は autofs によってリダイレクトされます。ファイルシステムに最後にアクセスしてから一定時間が経過し、autofs がそのファイルシステムを自動的にアンマウントするまで、automountd の介入は不要となります。

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、マスターマップファイル 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。

autofs マウントをマウントしたら、必要に応じて automount コマンドを実行し、autofs マウントを更新します。このコマンドは、auto_master マップにあるマウントのリストと、マウントテーブルファイル /etc/mnttab (前のバージョンでは /etc/mtab) にあるマウントされたファイルシステムのリストを比較します。その後、automount によって、適切な変更が加えられます。このプロセスにより、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したりすることなく、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 は不要になります。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たときです。

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 /etc/vfstab ファイル (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。

自動マウントデーモンである automountd は、ブート時に /etc/init.d/autofs スクリプトによって起動されます (図 16–1 を参照)。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ーマップを読み取り、autofs のマウントポイントをインストールします。詳細は、243 ページの「autofs のナビゲーションプロセス開始法 (マスターマップ)」 を参照してください。

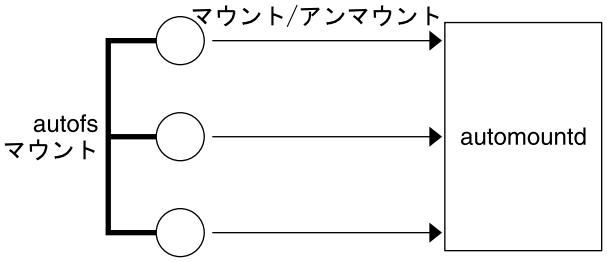


図 **16–1** /etc/init.d/autofs スクリプトによる automount の起動

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 マウントポイントで、ファイルシステムへのアクセスが要求された場合は、次の動作が行われます。

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 はそのファイルシステム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、手動でマウントまたは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。マップは、ネットワーク上の全ユーザーのパスワードエントリや、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を含むファイルです。マップには UNIX の管

理ファイルに相当するネットワーク規模の管理ファイルも含まれています。マップはローカルに使用するか、あるいは 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。ユーザーは自分の環境ニーズに適合するマップを作成します。251 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 (マスターマップ)

automount コマンドはシステムの起動時にマスターマップを読み取ります。図 16–2 に示すように、マスター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、マスター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。

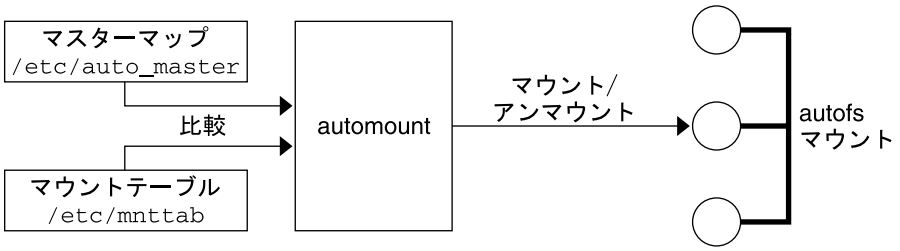


図 **16–2** マスターマップによるナビゲーション

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になります。

```
-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. サーバー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。前の例の /etc/auto_shared ファイルを拡張して、次の行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。この制限についての説明は、245 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。この問題を解決するには、オートマウンタを使用して、階層にあるすべてのコンポーネントのマウントを解除します。オートマウンタでマウントを解除するには、ファイルシステムのマウントが自動的に解除されるのを待つか、システムをリブートします。

autofs がクライアント用のもっと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)

以下は、直接マップの例です。

```
/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
```

マウントポイント `/usr/man` および `/usr/spool/news` には、複数の場所、つまり前者のマウントポイントは 3 つ、後者のマウントポイントは 2 つの場所が記述されています。このような場合、複製された場所のどこからマウントしてもユーザーは同じサービスを受けられます。ユーザー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、あるサーバー上のファイルを変更し、またすぐに別のサーバー上で「同じ」ファイルを変更しなければならないとしたら、たいへん面倒な作業になります。この利点は、もっとも利用しやすいサーバーが、そのユーザー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 (228 ページの「複製されたファイルシステムとは」 を参照)、クライアントはフェイルオーバー機能を使用できます。最適なサーバーが自動的に決定されるだけでなく、そのサーバーが使用できなくなるとクライアントは自動的に 2 番目に適したサーバーを使います。フェイルオーバー機能は、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、複数のサーバーがマニュアルページをエクスポートできます。どのサーバーからマニュアルページをマウントしても、そのサーバーが動作しており、しかもそのファイルシステムをエクスポートしているかぎり、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
```

これで、サーバー `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。どのサーバーがマウントに最適であるかは、特定レベルの FNS プロトコルをサポートしているサーバーの数、サーバーとの距離、およびその重み付けなどにより違います。

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ーの数が数えられます。サポートしているサーバーの数が多いプロトコルがデフォルトになります。これによって、クライアントにとっては利用できるサーバー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ーの組の中で数がもっとも多いものがわかると、サーバー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ーには、リモートサブネット上のサーバーよりも高い優先順位が付けられます。もっとも近いサーバー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。図 16–3 に、サーバーとの距離を示します。

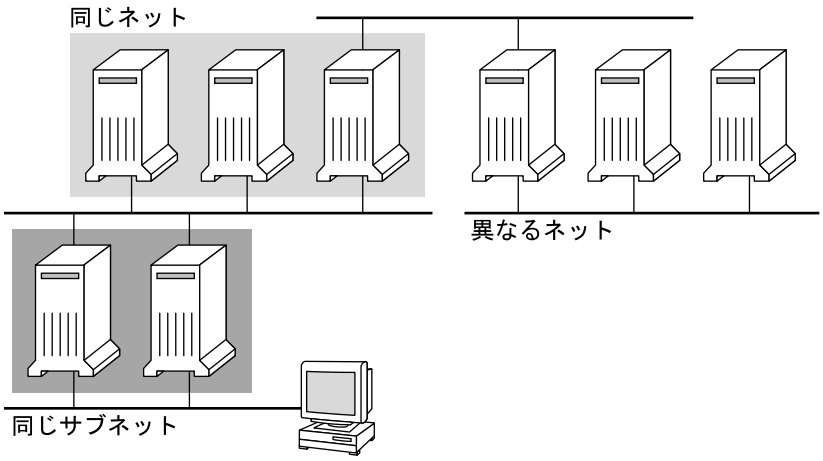


図 **16–3** サーバーとの距離

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ーが複数あるときは、それぞれのサーバーに接続する時間が計測され、速いものが使用されます。優先順位には、重み付けも関係します (248 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 のサーバーの方が多いと、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ーはリモートサブネット上のサーバーよりも優先されます。バージョン 2 のサーバーがあり、それがもっとも近いバージョン 3 サーバー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 サーバーがあり、もっとも近いバージョン 3 サーバーがリモートサブネット上にあると、バージョン 2 サーバーが優先されます。このことは、バージョン 3 サーバーの方がバージョン 2 サーバー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 サーバーの方が多いと、バージョン 2 サーバーしか選択されません。

フェイルオーバー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ー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ーが使用できなくなるたびにチェックされます。複数の場所を指定しておくと、個々のサーバー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 は最も近いサーバーを選択するため、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ーの場合は、それぞれが別々のサーバーであるとみなして、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いちばん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ーから 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:

```
/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ーの値はゼロです。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、そのサーバーが選択される可能性は低くなります。

注 – 重み付けは、サーバーの選択に関係する要素の中でもっと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ー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。この変数は、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、その後に続く文字や数字と変数とを区切ることができます。表 16–3 に定義済みのマップ変数を示します。

表 16–3 定義済みのマップ変数

変数	意味	提供元	例
ARCH	アーキテクチャタイプ	uname -m	sun4u
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
OSREL	オペレーティングシステムのリリース	uname -r	5.8
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	GENERIC

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、/usr/local/bin/sparc および /usr/local/bin/x86 から、SPARC アーキテクチャと IA アーキテクチャのバイナリをそれぞれエクスポートするファイルサーバーがあるとします。クライアントは、次のようなマップエントリを使ってマウントすることができます。

```
/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 ではなく、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、ローカルマップが NIS マップよりも先に検索されます。そのため、ローカルマップ /etc/auto_home に、もっとも頻繁にアクセスするホームディレクトリ用のエントリを含めることができます。他のエントリについては、スイッチを使用して NIS マップにフォールバックすることができます。

```
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後、一致するものがなければ、automount は現在のマップのスキャンを続けます。これは、+ エントリの後にさらにエントリを追加できることを意味します。

```
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。この場合、次のコマンドを実行すると、bee のファイルシステム /export1 がマウントされます。

```
% ls /execute/src
```

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、ユーザーの環境ニーズに合わせることができます。ユーザー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になるかどうかは、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って起動され、マスターマップ 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 を選択し、autofs が必要なマップを検出できない場合で、1 つまたは複数の下線を含むマップ名を検出したときには、それらの下線がドットに変更されます。こうすることにより、NIS の古いファイル名を利用することができます。次に autofs はもう 1 度マップを調べます。この手順を図 16–4 に示します。

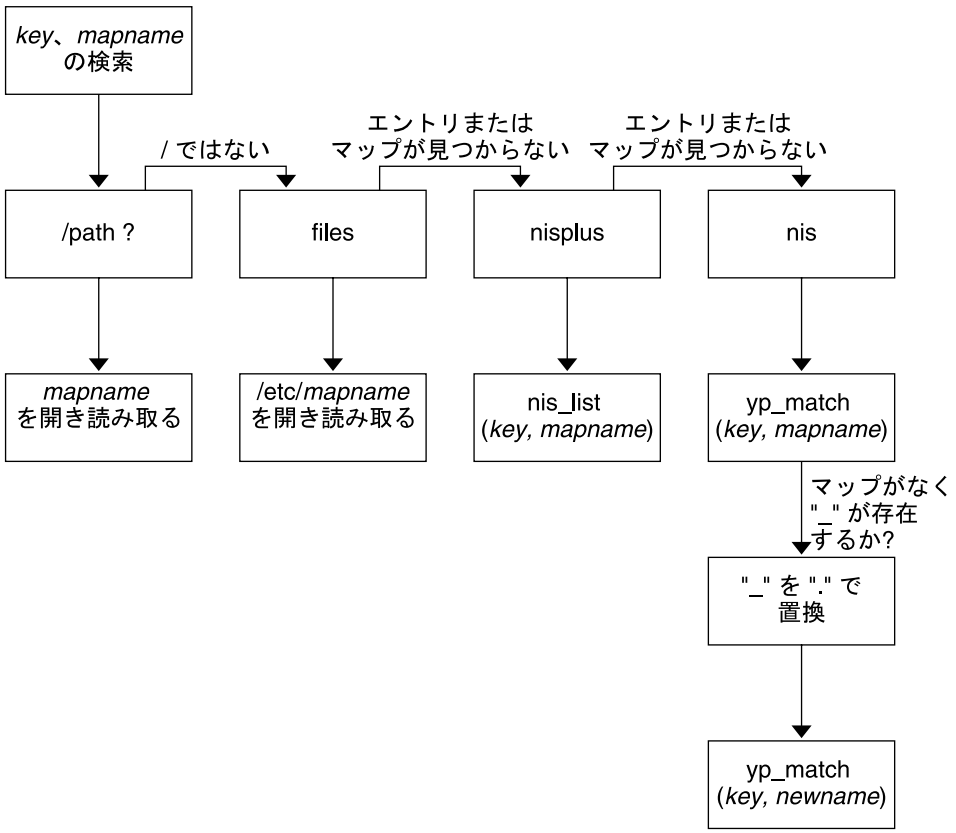


図 **16–4** autofs によるネームサービスの使用

このセッションでは、画面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

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。置き換えに使用する文字や、autofs のマップ構文解析機能から他の文字を保護するための文字もあります。

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は、文字列置換を使用できます。

```
john        willow:/home/john
mary        willow:/home/mary
joe         willow:/home/joe
able        pine:/export/able
baker       peach:/export/baker
```

この場合、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。アンパサンド文字を使用すると、前述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
```

また、このエントリは、次のようにさらに簡単にすることができます。

```
/usr/man                        willow,cedar,poplar:&
```

アンパサンド文字による置換では、キー文字列全体を使用していることに注意してください。そのため、直接マップ内のキーの最初の文字が / である場合は、そのスラッシュ (/) も引き継がれます。たとえば、次のように指定することはできません。

```
/progs                  &1,&2,&3:/export/src/progs
```

これは、autofs が、この例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用して、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、コンマ、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 - "
```

第 17 章

SLP (トピック)

以下の各章で、Solaris 9 オペレーティング環境におけるサービスロケーションプロトコル (SLP) の構成と配置について説明します。

第 18 章	SLP アーキテクチャと実装について説明します。
第 19 章	SLP 構成と SLPを有効にする方法について説明します。
第 20 章	SLP のエージェントとプロセスを構成する方法について説明します。
第 21 章	SLP のレガシーサービスの通知について説明します。
第 22 章	SLP のステータスコードとメッセージタイプをリストします。

第 18 章

SLP (概要)

サービスロケーションプロトコル (SLP) は、SLP が使用できるネットワークサービスを検出しそれに対応するための、移植性が高くプラットフォームに依存しないフレームワークを提供します。この章では、SLP のアーキテクチャの概要と、IP イントラネットに対応する SLP の Solaris 9 での実装について説明します。

SLP のアーキテクチャ

この節では、SLP の基本的な処理を示し、SLP の管理で使用されるエージェントとプロセスについて説明します。

SLP は、次のサービスを自動的に行い、設定はほとんどあるいはまったく必要ありません。

- クライアントアプリケーションがサービスへのアクセスに必要な情報を要求する
- プリンタ、ファイルサーバー、ビデオカメラ、HTTP サーバーなどのネットワークのハードウェアデバイスやソフトウェアサーバーにサービスを通知する
- 主サーバーの障害からの管理された回復

また、SLP の動作を管理、調整するために、必要に応じて次のことを実行できます。

- サービスとユーザーを論理グループや機能グループから構成されるスコープに編成する
- SLP のロギングを有効にして、ネットワーク上の SLP 動作の監視と障害追跡を行う
- SLP のタイミングパラメータを調整して、パフォーマンスの向上とスケーラビリティの拡張を行う

- SLP がマルチキャストルーティングに対応していないネットワークに配置されている場合、マルチキャストメッセージの送信や処理を行わないように SLP を構成する
- SLP のディレクトリエージェントを配置して、スケーラビリティとパフォーマンスを改善する

SLP 設計の概要

SLP ライブラリは、サービスをネットワークで検出するための情報を、サービスを通知するネットワーク対応のエージェントに与えます。SLP エージェントは、サービスの種類と場所に関する最新情報を保持します。これらのエージェントはプロキシ登録を使用することで、SLP が直接使用できないサービスを通知することもできます。詳細は、第 21 章を参照してください。

クライアントアプリケーションは、SLP ライブラリに依頼して、サービスを通知するエージェントに直接要求を出してもらいます。

SLP エージェントとプロセス

次の表では、SLP エージェントについて説明します。ここで使用する用語の詳細な定義は、用語集を参照してください。

表 18–1 SLP エージェント

SLP エージェント	説明
ディレクトリエージェント (DA)	サービスエージェント (SA) が登録する SLP 通知をキャッシュするプロセス。DA は、要求に応じて、サービス通知をユーザーエージェント (UA) に転送する
サービスエージェント (SA)	サービス通知を配信するためやサービスをディレクトリエージェント (DA) に登録するために、サービスの代理として動作する SLP エージェント
ユーザーエージェント (UA)	サービス通知情報を取得するために、ユーザーやアプリケーションの代理として動作する SLP エージェント
スコープ	サービスに対する管理上または論理上のグループ

次の図は、SLP アーキテクチャを実装する、基本的なエージェントおよびプロセスを示しています。図は、SLP のデフォルトの配置を表しています。特別な構成はまったく行われていません。UA と SA の 2 つのエージェントだけが必要です。SLP フレームワークでは、UA がサービス要求を SA にマルチキャストすることを許可しています。SA は、UA に対して応答をユニキャストします。たとえば、UA がサービス要求メッセージを送信すると、SA はサービス応答メッセージを返します。サービス応答には、クライアントの要求と一致するサービスの場所が含まれています。属性やサービスタイプに関する要求や応答も可能です。詳細は、第 22 章を参照してください。

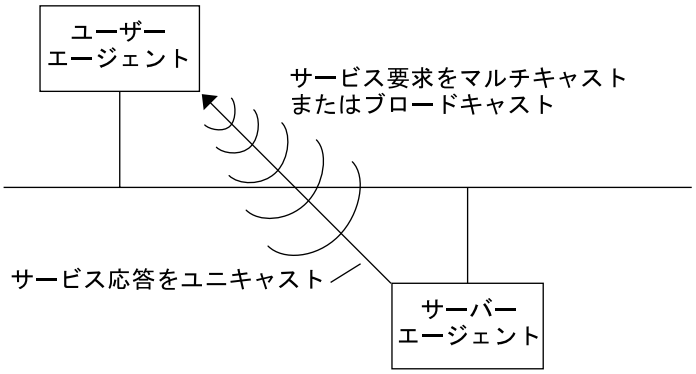


図 **18–1** SLP の基本的なエージェントとプロセス

以下の図は、フレームワークに DA が配置された場合の、SLP アーキテクチャを実装する基本的なエージェントとプロセスを示しています。

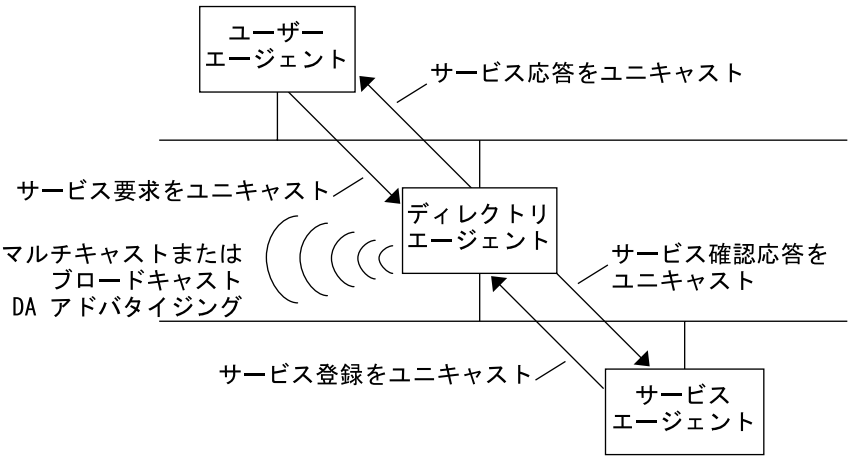


図 **18–2** DA を使って実装される SLP アーキテクチャのエージェントとプロセス

DA を配置すると、ネットワークにはより少ないメッセージが送られるので、UA は情報をすばやく受け取ることができます。DA は、ネットワークのサイズが増大する場合やマルチキャストルーティングがサポートされていない場合に必要です。DA は登録されたサービス通知のキャッシュの役割を果たします。SA は DA に対して、通知するすべてのサービスをリストした登録メッセージ (`SrvReg`) を送り、その応答として確認応答 (`SrvAck`) を受け取ります。サービス通知は DA によって更新されるか、通知に設定された有効期限に従って期限切れになります。UA が DA を検出すると、UA は要求を SA にマルチキャストするのではなく、DA にユニキャストします。

Solaris SLP メッセージについての詳細は、第 22 章を参照してください。

SLP の実装

Solaris SLP の実装では、SLP の SA、UA、DA、SA サーバー、スコープなどのアーキテクチャ部品 (表 18–1 を参照) の一部が `slpd` にマップされ、一部がアプリケーションプロセスにマップされます。SLP デーモン (`slpd`) は、特定のオフホストの SLP 相互作用を構成して、次のことを実行します。

- ネットワーク上のすべての DA に対し、ディレクトリエージェントの受動的検出と能動的検出を使用する
- ローカルホスト上の UA と SA が使用するために DA の更新テーブルを保持する
- レガシーサービス通知に対してプロキシ SA サーバーとして機能する (プロキシ登録)

`net.slpisDA` プロパティを設定し、`slpd` が DA として機能するように構成することもできます。第 20 章を参照してください。

SLP デーモンについては、`slpd(1M)` のマニュアルページを参照してください。

`slpd` の他に、C/C++ クライアントライブラリと Java クライアントライブラリ (`libslp.so` および `slp.jar`) が、UA クライアントと SA クライアントに SLP のフレームワークへのアクセスを提供します。クライアントライブラリは、次の機能を提供します。

- サービス通知の登録と登録解除が可能なネットワークサービスを提供するソフトウェア
- サービス通知にクエリーを発行することによってサービスを要求できるクライアントソフトウェア
- 登録と要求に使用できる SLP スコープのリスト

`slpd` とクライアントライブラリ (前述のサービスを提供する) 間のプロセス間通信を可能にするには、特別な構成は必要ありません。ただし、ライブラリが機能するように、先に `slpd` プロセスを実行してからクライアントライブラリをロードする必要があります。

次の図で、サービスプロバイダプログラム内の SLP クライアントライブラリは、SA の機能を使用します。サービスプロバイダプログラムは SLP クライアントライブラリを使用して、サービスを `slpd` に登録または登録解除します。サービスクライアントプログラムの SLP クライアントライブラリは、UA の機能を使用します。サービスクライアントプログラムは SLP クライアントライブラリを使用して、要求を出します。SLP クライアントライブラリは、SA に要求をマルチキャストするか、DA に要求をユニキャストします。この通信はアプリケーションから見て透過です。ただし、ユニ

キャスト方式の要求発行はより高速になります。クライアントライブラリの動作は、SLP のさまざまな構成プロパティの設定によって影響を受けます。第 20 章 を参照してください。`slpd` プロセスは、マルチキャスト要求への応答、DA への登録など、SA の全機能を処理します。

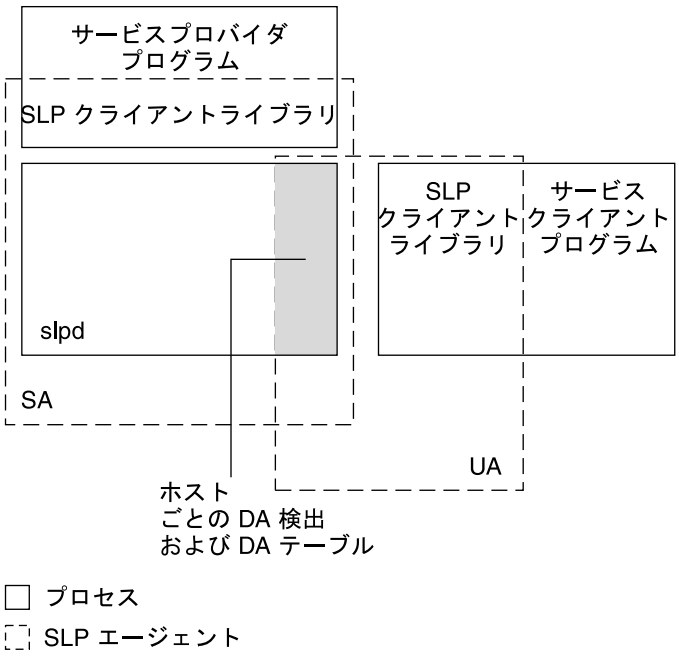


図 **18–3** SLP の実装

SLP の参考資料

SLP の詳細は、次の文書を参照してください。

- 『*Service Location Protocol for Enterprise Networks*』Kempf、James、Pete St. Pierre 著、John Wiley & Sons, Inc. 発行 (ISBN 番号 :0–47–3158–7)
- 『*Authentication Management Infrastructure Administration Guide*』(パート番号 :805–1139–03)
- 『*Service Location Protocol, Version 2, RFC 2608*』、Guttman、Erik、Chareles Perkins、John Veizades、Michael Day 著、Internet Engineering Task Force (IETF) 発行。[http://www.ietf.org/rfc/rfc2608.txt]
- 『*An API for Service Location, RFC 2614*』Kempf, James, Erik Guttman 著、Internet Engineering Task Force (IETF) 発行。[http://www.ietf.org/rfc/rfc2614.txt]

第 19 章

SLP の計画と有効化 (手順)

この章では、SLP の計画と有効化について説明します。次の節では、SLP の構成と SLP を有効にするためのプロセスを取り上げています。

SLP 構成の検討事項

Solaris 9 では、オペレーティング環境といっしょにインストールされるように、SLP デーモンがデフォルトのプロパティで構成済みです。デフォルトの設定で正しく動作する場合、SLP の配置において、ほとんど管理は必要ありません。

ただし場合によっては、デフォルトの SLP プロパティを変更して、SLP のネットワーク動作を調整することや各種の SLP 機能を有効にすることが必要になります。たとえば、いくつかの構成を変更して、SLP のロギングを有効にすることができます。SLP のログ情報と snoop トレースの情報によって、追加の構成が必要かどうかを判断できます。

SLP 構成プロパティは、/etc/inet ディレクトリ内の slp.conf ファイルにあります。デフォルトのプロパティ設定を変更する場合は、第 20 章の該当する手順を参照してください。

SLP 構成プロパティの設定を変更する前に、ネットワーク管理で大切な次のことがらを検討してください。

- 動作しているネットワーク技術の種類
- ネットワーク技術が円滑に処理できるトラフィック量
- ネットワークで使用できるサービスの数と種類

- ネットワーク上のユーザー数、ユーザーが必要とするサービス、もっとも頻繁にアクセスするサービスに関係するユーザーの場所

再構成の判断

SLP 対応の snoop ユーティリティと SLP ログユーティリティを使用して、再構成が必要かどうかや、変更する必要があるプロパティを判断できます。たとえば、次の目的のために特定のプロパティを再構成する場合があります。

- 各種の応答時間および帯域幅の性質が混在するネットワークメディアを調整する
- ネットワークの障害または計画されていないパーティション分割から回復させる
- DA を追加して SLP マルチキャストの急増を軽減する
- 新規のスコープを実装して、もっとも頻繁にアクセスするサービスにユーザーを編成する

snoop を使用して SLP 動作を監視する

snoop ユーティリティは受動的に機能する管理ツールで、ネットワークのトラフィック情報を提供します。ユーティリティ自身が発するトラフィックは最小限で、ネットワーク上のすべての動作を監視できます。

snoop ユーティリティは、実際の SLP メッセージトラフィックのトレースを行います。たとえば、snoop に `slp` コマンド行引数を付けて実行すると、登録および登録解除の SLP トレース情報が表示されます。このトレース情報を使用して、登録されているサービスの種類および登録動作の量をチェックできるので、ネットワークの負荷を測定できます。

snoop ユーティリティは、SLP ホスト間のトラフィックフローの監視にも役立ちます。snoop に `slp` コマンド行引数を付けて実行し、次の種類の SLP 動作を監視することで、ネットワークまたはエージェントの再構成が必要かどうかを判断できます。

- 特定の DA を使用しているホスト数。この情報により、負荷を均等にするために DA をさらに追加して配置するかどうかを判断できる
- 特定の DA を使用しているホスト数。この情報により、特定のホストに新規または別のスコープを構成すべきかどうかを判断できる
- UA がタイムアウトを要求しているか、あるいは DA の確認応答が遅いかどうか。UA のタイムアウトや再伝送を監視することで、DA が過負荷になっているかどうかを判断できる。DA が SA に登録の確認応答を送るのに数秒以上かかっているかどうかも確認できる。この情報により、必要に応じて、DA を追加したりスコープの構成を変更したりして、DA にかかるネットワーク負荷を調整する

snoop に -V (詳細) コマンド行引数を付けて実行すると、登録の有効期限や SrvReg の新規フラグの値を得ることができるので、再登録の数を削減すべきかどうかを判断できます。

snoop を使用して、次のような別の種類の SLP トラフィックをトレースすることもできます。

- UA クライアントと DA 間のトラフィック
- UA クライアントのマルチキャストとそれに対する SA の応答との間のトラフィック

snoop については、snoop(1M) のマニュアルページを参照してください。

ヒント – トラフィックおよび輻輳の統計情報を表示するには、netstat コマンドを snoop と併せて使用します。netstat の詳細は、netstat(1M) のマニュアルページを参照してください。

▼ snoop を使用して SLP トレースを実行する方法

1. スーパーユーザーになります。
2. snoop に slp コマンド行引数を付けて実行します。

   簡易モード :

   ```
   # snoop slp
   ```

   snoop をデフォルトの簡易モードで実行すると、進行中の出力が画面に表示されます。SLP メッセージは SLP トレースあたり 1 行に収まるように切り捨てられます。

   詳細モード :

   ```
   # snoop -v slp
   ```

   snoop を詳細モードで実行すると、進行中の出力がすべて画面に表示されます。出力される情報は次のとおりです。

   - サービス URL の完全なアドレス
   - すべてのサービス属性
   - 登録の有効期限
   - すべてのセキュリティパラメータとフラグ (存在する場合)

注 – slp コマンド行引数を snoop の他のオプションとともに使用できます。

---

snoop slp トレースの分析

次の例では、slpd は、*slphost1* 上で SA サーバーとしてデフォルトモードで動作しています。SLP デーモンは、*slphost2* をエコーサーバーとして初期化して登録しています。その後、snoop slp プロセスが *slphost1* 上で呼び出されます。

注 – トレース結果を説明しやすくするために、次の snoop からの出力結果にトレース行番号を付けています。

```
[1]slphost1 -> 239.255.255.253 SLP V@ SrvRqst [24487] service:directory-agent []
[2]slphost2 -> slphost1 SLP V2 DAAdvert [24487] service:directory-agent://129
[3]slphost1 -> 239.255.255.253 SLP V2 SrvRqst [24487] service:directory-agent []
[4]slphost1 -> 239.255.255.253 SLP V2 SrvRqst [24487] service:directory-agent []
[5]slphost1 -> slphost2 SLP V2 SrvReg [24488/tcp]service:echo.sun:tcp://slphost1:
[6]slphost2 -> slphost1 SLP V2 SrvAck [24488/tcp] ok
[7]slphost1 -> slphost2 SLP V2 SrvDereg [24489/tcp] service:echo.sun:tcp://slphost1:
[8]slphost2 -> slphost1 SLP V2 SrvAck [24489/tcp] ok
```

1. *slphost1* 上の slpd が、ディレクトリエージェントを探すために SLP マルチキャストグループアドレスにマルチキャストして、ディレクトリエージェントを能動検出していることを示しています。能動検出に対するメッセージ番号 (24487) は、トレース表示では角括弧内に示されます。
2. トレース 1 からの能動検出要求 24487 に対し、ホスト *slphost2* 上で DA として動作している slpd が応答したことを示します。*slphost2* からのサービス URL は 1 行に収まるように切り捨てられています。トレース 1 および 2 のメッセージ番号が一致していることからわかるように、DA はマルチキャストディレクトリエージェント検出メッセージに応答して、DA 通知を送っています。
3. 追加の DA に対する *slphost1* 上の UA からのマルチキャストを示します。*slphost2* はすでに要求に応答しているため、ふたたび応答することはなく、他のどの DA も応答しません。
4. 前の行で示したマルチキャストを繰り返しています。
5. *slphost1* 上の slpd は、SA クライアントが作成した登録をホスト *slphost2* 上の DA に転送します。エコーサーバーに対するユニキャストサービス登録 (SrvReg) が、*slphost1* によって *slphost2* 上の DA に行われています。
6. *slphost2* が *slphost1* の SrvReg に対してサービス確認応答 (SrvAck) で応答していることを表し、登録が完了したことを示しています。

   SA クライアントを稼働しているエコーサーバーと *slphost1* 上の SLP デーモンとの間のトラフィックは、snoop トレースでは表示されません。表示されないのは、snoop 動作がネットワークのループバックを超えて実行されているからです。
7. *slphost1* 上のエコーサーバーが、エコーサービス通知の登録を解除します。*slphost1* 上の SLP デーモンは、登録解除を *slphost2* 上の DA に転送します。
8. *slphost2* が *slphost1* に対してサービス確認応答 (SrvAck) で応答していることを表し、登録解除が完了したことを示しています。

   トレース行 5、6、7、8 のメッセージ番号に追加されている /tcp パラメータは、メッセージ交換が TCP で発生したことを示しています。

次に進む手順

SLP トラフィックを監視後、snoop トレースから集められた情報を使用して、SLP デフォルトの再構成が必要かどうかを判断できます。SLP プロパティ値の設定については、第 20 章を参照してください。SLP メッセージとサービス登録については、第 22 章を参照してください。

SLP の有効化

SLP を有効にするには、SLP デーモン slpd を実行します。SLP は、SLP デーモン slpd の実行で有効になります。slpd を起動するためにサポートされているインタフェースが /etc/init.d/slpd スクリプトで、SLP 構成ファイル /etc/inet/slp.conf が存在すればデーモンを起動します。Solaris オペレーティング環境には、/etc/inet/slp.conf.example というファイルがあります。ブート時に SLP を有効にするには、このファイル名を /etc/inet/slp.conf に変更します。

第 20 章

SLP の管理 (手順)

この章では、SLP のエージェントとプロセスを構成するための情報と作業手順について説明します。

SLP プロパティの構成

SLP 構成プロパティは、ネットワークの相互作用、SLP エージェントの特性、状態、およびログを制御します。ほとんどの場合、これらのプロパティのデフォルトの構成は変更する必要がありません。ただし、ネットワークの媒体またはトポロジが変更される場合は、次の目的を達成するためには、この章の手順を使用します。

- ネットワークの応答時間を補正する
- ネットワークの輻輳を軽減する
- エージェントの追加、または IP アドレスの再割り当てを行う
- SLP ログを起動する

SLP 構成ファイル `/etc/inet/slp.conf` を編集して、次の表に示す処理を行うことができます。

表 **20–1** SLP 構成の操作

操作	説明
slpd が DA サーバーと SA サーバーのどちらで機能するかを指定する。SA サーバーがデフォルト	net.slpisDA プロパティに True を設定する
マルチキャストメッセージのタイミングを設定する	net.slp.DAHeartBeat プロパティを設定して、請求されていない DA 通知を DA がマルチキャストする回数を制御する
DA ロギングを使用可能にしてネットワークトラフィックを監視する	net.slp.traceDATraffic プロパティに True を設定する

SLP 構成ファイルの基本要素

/etc/inet/slp.conf ファイルは、SLP デーモンを再起動するたびにすべての SLP 動作を定義して起動します。構成ファイルは次の要素から成ります。

- 構成プロパティ
- コメント行と注釈

構成プロパティ

net.slp.isDA や net.slp.DAHeartBeat などのすべての基本的な SLP プロパティは、次の書式で名前が付けられています。

```
net.slp.<keyword>
```

SLP の動作は、slp.conf ファイル内のプロパティの値またはプロパティの組み合わせによって定義されます。プロパティは、SLP 構成ファイル内でキーと値の対で構成されています。次の例に示すように、キーと値の対は、プロパティ名とその設定値で構成されています。

```
<property name>=<value>
```

各プロパティのキーはプロパティ名です。値はプロパティに、数値 (間隔または時間)、真偽の状態、または文字列値のパラメータを設定します。プロパティの値は次のデータ型の 1 つで構成されます。

- 真偽設定 (ブール型)
- 整数
- 整数のリスト
- 文字列
- 文字列のリスト

コメント行と注釈

slp.conf ファイルにコメントを追加して、その行の性質および機能を説明できます。コメント行はファイルに任意に書き込めるので、管理する上で役立ちます。

注 – 構成ファイル内の設定には、大文字と小文字の区別がありません。詳細は、Guttman, Erik, James Kempf, Charles Perkins 著、Internet Engineering Task Force (IETF) 発行の『Service Templates and service: scheme RFC 2609』を参照してください。 (http://www.ietf.org/rfc/rfc2609.txt)

▼ SLP 構成の変更方法

SLP 構成ファイルのプロパティ設定を変更するには、次の手順を実行します。SLP が使用できるクライアントまたはサービスソフトウェアは、SLP API を使用して、SLP 構成も変更できます。API については、Internet Engineering Task Force (IETF) 発行の『An API for Service Location, RFC 2614』を参照してください。
(http://www.ietf.org/rfc/rfc2614.txt)

1. スーパーユーザーになります。
2. ホスト上の slpd とすべての **SLP** 動作を停止します。

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

3. 構成の設定を変更する前に、デフォルトの /etc/inet/slp.conf ファイルのバックアップをとります。
4. 必要に応じて、/etc/inet/slp.conf ファイルのプロパティ設定を編集します。

   SLP プロパティの設定については、270 ページの「構成プロパティ」 を参照してください。slp.conf プロパティを変更する別の例については、後述の各節を参照してください。slp.conf(4) のマニュアルページを参照してください。
5. 変更を保存し、ファイルを閉じます。
6. 変更を反映するには、slpd を再起動します。

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

注 – slpd を停止または起動するとき、SLP デーモンは構成ファイルから情報を取得します。

たとえば、slpd.conf ファイルの net.slp.isDA プロパティに True を設定して、slpd が DA サーバーとして動作するように SA サーバーのデフォルトを変更できます。

```
net.slp.isDA=True
```

各領域で、各種のプロパティが構成の異なる場合を制御します。以降の各節では、SLP 構成で使用するデフォルトのプロパティ設定を変更するさまざまなシナリオについて説明します。

DA 通知と検出頻度の変更

次のような場合は、DA 通知と検出要求のタイミングを制御するプロパティを変更できます。

- SA または UA が `slp.conf` ファイルの `net.slp.DAAddresses` プロパティから静的に DA 構成情報を取得するように設定したい場合は、DA 検出を無効にできる
- ネットワークが頻繁にパーティション分割を行う場合は、受動的な通知および定期的な能動的検出の頻度を変更できる
- UA と SA クライアントがダイアルアップ接続の一方の側で DA にアクセスしている場合は、DA のハートビート頻度と能動的検出の間隔を減らすことで、ダイアルアップ回線の起動回数を少なくできる
- ネットワークが輻輳している場合は、マルチキャストを制限できる

この節の手順では、次のプロパティを変更する方法について説明します。

表 **20–2** DA 通知タイミングと検出要求のプロパティ

プロパティ	説明
`net.slp.passiveDADetection`	請求されていない DA 通知を `slpd` が受信するかどうかを示すブール値
`net.slp.DAActiveDiscoveryInterval`	`slpd` が、新しい DA に対して DA の能動的検出を実行する頻度を示す値
`net.slp.DAHeartBeat`	請求されていない DA 通知を DA がマルチキャストする頻度を示す値

UA と SA を静的に構成された DA に限定する

UA と SA が `slp.conf` ファイル内の静的な構成情報から DA アドレスだけを取得するように制限することが必要な場合があります。次の手順では、`slpd` が `net.slp.DAAddresses` プロパティから DA 情報だけを取得するように 2 つのプロパティを変更できます。

▼ UA と SA を静的に構成された DA に限定する方法

次の手順に従って、net.slp.passiveDADetection および net.slp.DAActiveDiscoveryInterval プロパティを変更します。

注 – この手順は、静的な構成を使用するように制限されている UA と SA を実行するホストにだけ使用してください。

---

1. スーパーユーザーになります。
2. ホスト上の slpd とすべての **SLP** 動作を停止します。

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

3. 構成の設定を変更する前に、デフォルトの /etc/inet/slp.conf ファイルのバックアップをとります。
4. slp.conf ファイル内の net.slp.passiveDADetection プロパティに False を設定して、受動的検出を無効にします。この設定により、slpd は請求されていない **DA** 通知を無視します。

   ```
   net.slp.passiveDADetection=False
   ```

5. net.slp.DAActiveDiscoveryInterval に -1 を設定して、初期および定期の能動的検出を無効にします。

   ```
   net.slp.DAActiveDiscoveryInterval=-1
   ```

6. 変更を保存し、ファイルを閉じます。
7. 変更を反映するには、slpd を再起動します。

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

ダイアルアップネットワークに対する DA 検出の構成

UA または SA がダイアルアップネットワークによって DA から切り離されている場合は、DA 検出を構成して、検出要求と DA 通知の数を削減するか、完全になくすことができます。ダイアルアップネットワークでは、通常起動時に課金されます。余分な通話を最小限に抑えることにより、ダイアルアップネットワークの使用コストを削減できます。

注 – 272 ページの「UA と SA を静的に構成された DA に限定する」 で説明している方法で、DA 検出を完全に無効にすることができます。

▼ ダイアルアップネットワークに対する DA 検出の構成方法

次の手順に従って、DA ハートビートの期間と能動的検出の間隔を長くすることで、請求されていない DA 通知と能動的検出を削減できます。

1. スーパーユーザーになります。
2. ホスト上の `slpd` とすべての **SLP** 動作を停止します。

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

3. 構成の設定を変更する前に、デフォルトの `/etc/inet/slp.conf` ファイルのバックアップをとります。
4. `slpd.conf` ファイル内の `net.slp.DAHeartbeat` プロパティの値を大きくします。

   ```
   net.slp.DAHeartbeat=value
   ```

value	32 ビットの整数で、DA 通知の受動的ハートビートに対して秒数を設定する デフォルト値は、10800 秒 (3 時間) 値の範囲は、2000 から 259200000 秒

   たとえば、DA を実行しているホストに対して、DA のハートビートを約 18 時間に設定できます。

   ```
   net.slp.DAHeartbeat=65535
   ```

5. `slpd.conf` ファイル内の `net.slp.DAActiveDiscoveryInterval` プロパティの値を大きくします。

   ```
   net.slp.DAActiveDiscoveryInterval value
   ```

value	32 ビットの整数で、DA の能動的検出クエリーに対して秒数を設定する デフォルトの値は、900 秒 (15 分) 値の範囲は、300 から 10800 秒

たとえば、UA と SA を実行しているホストに対して、DA の能動的検出の間隔を 18 時間に設定できます。

```
net.slp.DAActiveDiscoveryInterval=65535
```

6. 変更を保存し、ファイルを閉じます。
7. 変更を反映するには、slpd を再起動します。

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

頻繁なパーティション分割に対する DA のハートビートの構成

スコープをサポートするすべての DA に登録するには、SA が必要です。DA は、slpd が能動的検出を行なった後で現れることがあります。DA が slpd スコープをサポートする場合、SLP デーモンはホスト上のすべての通知を DA に登録します。

slpd が DA を検出する 1 つの方法は、起動時に DA が送り出す初期の請求されていない通知を使用します。SLP デーモンは定期的な請求されていない通知 (ハートビート) を使用して、DA がまだアクティブであるかどうかを判断します。ハートビートが出現しない場合、SLP デーモンは自分が使用する DA を削除し、これを UA に申し出ます。

最後に、DA にシャットダウン要求が出されると、DA は特別な DA 通知を転送して、受信中の SA サービスに DA がサービスから抜け出すことを知らせます。SLP デーモンもこの特別な通知を使用して、キャッシュからアクティブでない DA を削除します。

ネットワークが頻繁にパーティション分割を行い、SA の期限が長い場合、ハートビートの通知を受けなければ、slpd はパーティションの分割中に DA をキャッシュから削除できます。ハートビートの頻度を減らすことにより、使用中止になった DA がパーティションの修正後にキャッシュに復元されるまでの遅延時間を縮小できます。

▼ 頻繁なパーティション分割に対して DA のハートビートを構成する方法

次の手順に従って、net.slp.DAHeartBeat プロパティを変更し、DA のハートビート期間を短くします。

1. スーパーユーザーになります。
2. ホスト上の slpd とすべての **SLP** 動作を停止します。

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

3. 構成の設定を変更する前に、デフォルトの /etc/inet/slp.conf ファイルのバックアップをとります。

4. **DA** のハートビートの値を **1** 時間 **(3600** 秒**)** に縮小します。デフォルトでは、**DA** のハートビート期間は **3** 時間 **(10800** 秒**)** に設定されています。

```
net.slp.DAHeartBeat=3600
```

5. 変更を保存し、ファイルを閉じます。

6. 変更を反映するには、`slpd` を再起動します。

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

注 – DA 検出が完全に無効になっている場合、ホスト上で実行されている UA と SA が正しい DA にアクセスするように、`net.slp.DAAddresses` プロパティを `slp.conf` に設定する必要があります。

ネットワーク輻輳の軽減

ネットワークが非常に混雑している場合、マルチキャストの量を制限できます。ネットワークに DA を配置していない場合は、DA を配置すると SLP 関連のマルチキャストの量を大幅に削減できます。

ただし、DA の配置後でも DA 検出のためのマルチキャストは必要です。 DA 検出に必要なマルチキャストの量は、274 ページの「ダイアルアップネットワークに対する DA 検出の構成方法」 で説明している方法で削減できます。272 ページの「UA と SA を静的に構成された DA に限定する」 で説明している方法で、DA 検出のためのマルチキャストを完全になくすことができます。

異なるネットワーク媒体、トポロジ、または構成の調整

この節では、次のプロパティを変更して SLP のパフォーマンスを調整する場合の可能なシナリオについて説明します。

表 **20–3** SLP パフォーマンスのプロパティ

プロパティ	説明
`net.slp.DAAttributes`	DA が通知を受け取る最短の更新間隔
`net.slp.multicastTTL`	マルチキャストパケットの有効期限

表 **20–3** SLP パフォーマンスのプロパティ　(続き)

プロパティ	説明
net.slp.MTU	ネットワークパケットのサイズ (バイト)。サイズには、IP と TCP または UDP の各ヘッダーが含まれている
net.slp.isBroadcastOnly	ブロードキャストを DA サービス検索および DA ベースでないサービス検索に使用する必要があるかどうかを示すために設定されるブール値

SA 再登録の削減

SA は、期限が切れる前に定期的にサービス通知を更新する必要があります。DA が多くの UA と SA の非常に重い負荷を処理している場合は、頻繁な更新により DA が過負荷になることがあります。DA が過負荷になると、UA の要求がタイムアウトして欠落します。UA 要求のタイムアウトには多くの原因が考えられます。DA の過負荷が問題であると仮定する前に、snoop トレースを使ってサービス登録に登録されているサービス通知の有効期限を確認してください。有効期限が短く、再登録が頻繁に発生している場合は、再登録が頻繁すぎることがタイムアウトの原因と考えられます。

注 – サービス登録は、FRESH フラグが設定されていなければ再登録になります。サービス登録メッセージについては、第 22 章を参照してください。

▼ SA 再登録を削減する方法

次の手順に従って、SA の最小更新間隔を長くすることで、再登録回数を削減します。

1. スーパーユーザーになります。
2. ホスト上の slpd とすべての **SLP** 動作を停止します。

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

3. 構成の設定を変更する前に、デフォルトの /etc/inet/slp.conf ファイルのバックアップをとります。
4. net.slp.DAAttributes プロパティの min-refresh-interval 属性の値を大きくします。

   デフォルトの最短再登録期間はゼロ (0) です。デフォルトがゼロの場合、SA はいつでも自由に再登録できます。次の例では、間隔は 3600 秒 (1 時間) に増やしています。

   ```
   net.slp.DAAttributes(min-refresh-interval=3600)
   ```

5. 変更を保存し、ファイルを閉じます。

6. 変更を反映するには、slpd を再起動します。

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

マルチキャストの有効期限プロパティの構成

マルチキャストの有効期限プロパティ (net.slp.multicastTTL) によって、マルチキャストパケットがイントラネット内で伝達される範囲が決まります。マルチキャスト TTL は net.slp.multicastTTL プロパティを 1 ～ 255 までの整数に設定することにより構成されます。マルチキャスト TTL のデフォルト値は 255 で、これは理論的にはパケット経路が無制限であることを意味します。しかし、TTL を 255 とすると、マルチキャストパケットがイントラネットを超えて管理ドメインの端にある境界ルーターまで進む原因になります。マルチキャストパケットがインターネットのマルチキャストバックボーンまたは ISP に漏れないようにするには、境界ルーター上のマルチキャストが正しく構成されている必要があります。

マルチキャスト TTL のスコープ設定は、TTL 比較が作成されることを除いて、標準的な IP の TTL と似ています。マルチキャストを実行できるルーター上の各インタフェースには、TTL 値が割り当てられています。マルチキャストパケットが着信すると、ルーターはパケットの TTL をインタフェースの TTL と比較します。パケットの TTL がインタフェースの TTL 値と同じかそれより大きい場合は、標準的な IP の TTL の場合と同じように、パケットの TTL を 1 減らします。TTL がゼロになると、そのパケットは破棄されます。SLP マルチキャストに TTL スコープを使用する場合、パケットをイントラネットの特定のサブセクションに限定するために、ルーターが正しく構成されている必要があります。

▼ マルチキャストの有効期限プロパティの構成方法

次の手順に従って、net.slp.multicastTTL プロパティを設定し直します。

1. スーパーユーザーになります。
2. ホスト上の slpd とすべての **SLP** 動作を停止します。

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

3. 構成の設定を変更する前に、デフォルトの /etc/inet/slp.conf ファイルのバックアップをとります。
4. slpd.conf ファイル内の net.slp.multicastTTL プロパティを変更します。

```
net.slp.multicastTTL=value
```

value	マルチキャスト TTL を定義する 255 以下の正の整数

注 – TTL 値を減らしてマルチキャストの伝達範囲を縮小することができます。TTL の値が 1 の場合、パケットはそのサブネットに限定されます。TTL の値が 32 の場合は、パケットはそのサイトに限定されます。「サイト」は、マルチキャスト TTL について記述されている RFC 1075 では定義されていません。32 以上の値は、インターネット上の論理的な経路を指すので使用しないでください。32 未満の値は、各ルーターが TTL で正しく構成されていれば、マルチキャストをアクセス可能なサブネットのセットに限定するために使用できます。

5. 変更を保存し、ファイルを閉じます。
6. 変更を反映するには、`slpd` を再起動します。

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

パケットサイズの構成

SLP のデフォルトのパケットサイズは 1400 バイトです。ほとんどのローカルエリアネットワークにはこのサイズで十分です。無線ネットワークまたは広域ネットワークの場合は、メッセージの断片化を防いだりネットワークのトラフィックを削減したりするために、パケットサイズを縮小できます。より大きなパケットを持つローカルエリアネットワークの場合は、パケットサイズを大きくするとパフォーマンスを向上できます。ネットワークの最小パケットサイズを確認して、パケットサイズの縮小が必要かどうかを判断できます。ネットワーク媒体のパケットサイズがより小さい場合は、パケットサイズに合わせて `net.slp.MTU` の値を小さくできます。

ネットワーク媒体のパケットサイズがより大きい場合は、パケットサイズに合わせて値を大きくできます。ただし、SA からのサービス通知または UA からのクエリーが頻繁にデフォルトのパケットサイズをオーバーフローするのでなければ、`net.slp.MTU` の値を変更する必要はありません。`snoop` を使用して、UA 要求がデフォルトのパケットサイズを頻繁にオーバーフローし、UDP ではなく TCP を使用するためにロールオーバーしているかどうかを判断できます。

`net.slp.MTU` プロパティは、リンク層ヘッダー、IP ヘッダー、UDP または TCP ヘッダー、SLP メッセージを含めた、IP パケットの全体サイズを測定します。

▼ パケットサイズの構成方法

次の手順に従って、`net.slp.MTU` プロパティを調整することで、デフォルトのパケットサイズを変更します。

1. スーパーユーザーになります。
2. ホスト上の `slpd` とすべての **SLP** 動作を停止します。

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

3. 構成の設定を変更する前に、デフォルトの /etc/inet/slp.conf ファイルのバックアップをとります。
4. slpd.conf ファイル内の net.slp.MTU プロパティを変更します。

   net.slp.MTU=*value*

value	16 ビットの整数で、ネットワークのパケットサイズ (バイト単位) デフォルト値は、1400 値の範囲は、128 から 8192

5. 変更を保存し、ファイルを閉じます。
6. 変更を反映するには、slpd を再起動します。

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

ブロードキャスト専用ルーティングの構成

SLP は、DA が存在しない場合に、マルチキャストを使ってサービス検出や DA 検出を行うように設計されています。使用するネットワークが、マルチキャストルーティングを配置しない場合は、net.slp.isBroadcastOnly プロパティに True を設定することで、SLP がブロードキャストを使用するように構成できます。

マルチキャストと異なり、ブロードキャストパケットはデフォルトの場合サブネットを越えて伝達しません。このため、マルチキャストを行わないネットワークでは、DA を使用しないサービス検出は、単一のサブネット上でしか機能しません。さらに、ブロードキャストが使用されているネットワークに DA およびスコープを配置する場合は、特別な考慮が求められます。マルチホームホスト上の DA は、マルチキャストが使用できない複数のサブネット間でサービス検出をブリッジできます。マルチホームホスト上の DA の配置については、296 ページの「DA の配置とスコープ名の割り当て」 を参照してください。

▼ ブロードキャスト専用ルーティングの構成方法

次の手順に従って、net.slp.isBroadcastOnly プロパティを True に変更します。

1. スーパーユーザーになります。
2. ホスト上の slpd とすべての **SLP** 動作を停止します。

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

3. 構成の設定を変更する前に、デフォルトの /etc/inet/slp.conf ファイルのバックアップをとります。
4. slpd.conf ファイル内の net.slp.isBroadcastOnly プロパティを True に変更します。

```
net.slp.isBroadcastOnly=True
```

5. 変更を保存し、ファイルを閉じます。
6. 変更を反映するには、slpd を再起動します。

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

SLP 検出要求のタイムアウトの変更

SLP 検出要求のタイムアウトを変更する必要があるのは、次の 2 つの場合です。

- SLP エージェントが複数のサブネット、ダイアルアップ回線、または別の WAN によって切り離されている場合は、ネットワークの応答時間が長く、デフォルトのタイムアウトでは要求や登録を完了できないことがある。逆に、ネットワークの応答時間が短い場合は、タイムアウトを短くすることにより、パフォーマンスを向上できることがある
- トラフィックが多いネットワークまたは衝突率の高いネットワークの場合、SA および UA がメッセージを送る前に待たなければならない最長の時間が不足して、衝突のないトランザクションを確保できない場合がある

デフォルトのタイムアウトの変更

ネットワークの応答時間が長いと、UA および SA が要求と登録を行う場合、応答を受け取る前にタイムアウトになる原因になります。複数のサブネット、ダイアルアップ回線、または WAN によって UA が SA から切り離されている場合、または UA と SA の両方が DA から切り離されている場合、応答時間が問題となることがあります。応答時間が問題であるかどうかを判断するには、UA および SA の要求と登録でタイムアウトが起こったために SLP 要求が失敗しているかどうかを確認します。ping コマンドを使って実際の応答時間を測定することもできます。

次の表は、タイムアウトを制御する構成プロパティを示します。この節で説明する手順で、これらのプロパティを変更できます。

表 **20–4** タイムアウトプロパティ

プロパティ	説明
net.slp.multicastTimeouts net.slp.DADiscoveryTimeouts net.slp.datagramTimeouts	これらのプロパティは、メッセージ転送が中止されるまで、マルチキャストやユニキャストが繰り返し実行する UDP メッセージの転送に使用できるタイムアウトのリストを制御する
net.slp.multicastMaximumWait	このプロパティは、マルチキャストメッセージが中止されるまで、転送される最長時間を制御する
net.slp.datagramTimeouts	このプロパティにリストされる値の合計を示す DA タイムアウトの上限。UDP ダイアグラムは、応答を受け取るかタイムアウトの上限になるまで、DA に繰り返し送られる

マルチキャストサービスの検出中または DA の検出中に頻繁にタイムアウトが発生する場合は、net.slp.multicastMaximumWait プロパティをデフォルト値の 15000 ミリ秒 (15 秒) から増やしてください。最大待ち時間を長くすることにより、応答時間の長いネットワーク上で要求に対してより長い時間が許可されます。net.slp.multicastMaximumWait プロパティの値を増やした後は、net.slp.multicastTimeouts と net.slp.DADiscoveryTimeouts も変更する必要があります。これらのプロパティのタイムアウト値の合計が net.slp.multicastMaximumWait 値と等しくなるようにしてください。

▼ デフォルトのタイムアウトの変更方法

次の手順に従って、タイムアウトを制御する SLP プロパティを変更します。

1. スーパーユーザーになります。
2. ホスト上の slpd とすべての **SLP** 動作を停止します。

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

3. 構成の設定を変更する前に、デフォルトの /etc/inet/slp.conf ファイルのバックアップをとります。
4. slpd.conf ファイル内の net.slp.multicastMaximumWait プロパティを変更します。

   net.slp.multicastMaximumWait=*value*

value

32 ビットの整数で、net.slp.multicastTimeouts と net.slp.DADiscoveryTimeouts に設定する値の合計値を示す

デフォルト値は、15000 ミリ秒 (15 秒)

値の範囲は、1000 から 60000 ミリ秒

たとえば、マルチキャスト要求で 20 秒 (20000 ミリ秒) 必要だと判断したら、net.slp.multicastTimeouts プロパティと net.slp.DADiscoveryTimeouts プロパティにリストされている値の合計が 20000 ミリ秒になるように調整します。

```
net.slp.multicastMaximumWait=20000
net.slp.multicastTimeouts=2000,5000,6000,7000
net.slp.DADiscoveryTimeouts=3000,3000,6000,8000
```

5. slpd.conf ファイル内の net.slp.datagramTimeouts プロパティを必要に応じて変更します。

```
net.slp.datagramTimeouts=value
```

value

32 ビット整数のリストで、ユニキャストのデータグラム転送を DA に実行するためのタイムアウト (ミリ秒)

デフォルト値は、3000,3000,3000

たとえば、頻繁なタイムアウトの発生を回避するために、データグラムのタイムアウトを 20000 ミリ秒に増やすことができます。

```
net.slp.datagramTimeouts=2000,5000,6000,7000
```

高パフォーマンスのネットワークでは、逆に UDP データグラム転送のマルチキャストまたはユニキャストのタイムアウトの上限を小さくできます。 タイムアウトの上限を小さくすることで、SLP 要求を満たすために必要な応答時間を短縮できます。

6. 変更を保存し、ファイルを閉じます。

7. 変更を反映するには、slpd を再起動します。

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

ランダム待ち時間の上限の構成

トラフィックの重いネットワークや衝突率の高いネットワークでは、DA との通信が影響を受けることがあります。衝突率が高い場合、送信エージェントは、UDP データグラムを再転送する必要があります。再転送が発生しているかどうかは、snoop を使用して、SA サーバー として slpd を実行しているホスト、および DA として slpd を実行しているホストのネットワークトラフィックを監視することにより判断できます。 SA サーバーとして slpd を実行しているホストから同じサービスについて複数のサービス登録メッセージが snoop トレースに現れる場合は、衝突の問題があると考えられます。

衝突は、ブート時の主要な問題となる場合があります。DA が最初に起動されると、DA は請求されていない通知を送り出し、SA はそれらの登録に応答します。SLP は、DA 通知を受け取ってから応答するまでにランダムな時間だけ、SA を待たせます。このランダムな待ち時間は、net.slp.randomWaitBound によって制御される最大値を使って均等に分散されます。デフォルトのランダム待ち時間の上限は 1000 ミリ秒 (1 秒) です。

▼ ランダム待ち時間の上限の構成方法

次の手順に従って、slp.conf ファイルの net.slp.RandomWaitBound プロパティを変更します。

1. スーパーユーザーになります。
2. ホスト上の slpd とすべての **SLP** 動作を停止します。

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

3. 構成の設定を変更する前に、デフォルトの /etc/inet/slp.conf ファイルのバックアップをとります。
4. slpd.conf ファイル内の net.slp.RandomWaitBound プロパティを変更します。

   net.slp.RandomWaitBound=*value*

value	DA に接続するまでのランダム待ち時間の計算に使用される上限 デフォルト値は、1000 ミリ秒 (1 秒) 値の範囲は、1000 から 3000 ミリ秒

   たとえば、ランダム待ち時間を 5000 ミリ秒 (5 秒) に延長できます。

   ```
   net.slp.randomWaitBound=5000
   ```

   ランダム待ち時間の上限を長くすると、登録で遅延が長くなります。SA は新しく検出された DA をより時間をかけて登録できるので、衝突とタイムアウトを回避することができます。
5. slpd.conf ファイル内の net.slp.datagramTimeouts プロパティを必要に応じて変更します。

   net.slp.datgramTimeouts=*value*

value	32 ビット整数のリストで、ユニキャストのデータグラム転送を DA に実行するためのタイムアウト (ミリ秒) デフォルト値は、3000,3000,3000

たとえば、頻繁なタイムアウトの発生を回避するために、データグラムのタイムアウトを 20000 ミリ秒に増やすことができます。

```
net.slp.datagramTimeouts=2000,5000,6000,7000
```

高パフォーマンスのネットワークでは、逆に UDP データグラム転送のマルチキャストまたはユニキャストのタイムアウトの上限を小さくできます。 この設定により、SLP 要求を満たす際に、応答時間を短縮できます。

6. 変更を保存し、ファイルを閉じます。
7. 変更を反映するには、`slpd` を再起動します。

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

スコープの配置

スコープを使用すると、論理的、物理的、および管理上のユーザーのグループによるサービスへの対応が可能です。スコープを使用することで、サービス通知へのアクセスの管理が可能になります。

`net.slp.useScopes` プロパティを使用すると、スコープを作成できます。たとえば、次のように構成すると、ホスト上の `/etc/inet/slp.conf` ファイルに、`newscope` という名前の新規のスコープが追加されます。

```
net.slp.useScopes=newscope
```

スコープの概念を理解しやすくするために、会社にプリンタや FAX などのネットワーク接続されたオフィス機器の小部屋がビルディング 6 の 2 階の南の大部屋の端にあるとします。これらのオフィス機器は 2 階のすべてのユーザーに提供されている場合や、使用が特定の部署のメンバーに限定されている場合があります。スコープはこれらの機器に対するサービス通知へのアクセスに対応する手段を提供します。

オフィス機器をマーケティング部専用にすると、`mktg` という名前のスコープを作成することができます。別の部署に所属しているオフィス機器は、別のスコープ名で構成できます。

また、部署が分散している場合もあります。たとえば、機械工学部門と CAD/CAM 部門が 1 階と 2 階に分かれているとします。この場合でも、両者に同じスコープを割り当てることにより、1 階と 2 階にあるホストに 2 階のマシンを提供できます。ネットワークとユーザーに都合よく動作するように、スコープはどのように配置してもかまいません。

注 – 特定のスコープを持つ UA は、別のスコープで通知されたサービスを実際に使用できないわけではありません。スコープの構成は、UA が検出するサービス通知を制御するだけです。サービス自体が、なんらかのアクセス制御の制限を行う必要があります。

スコープを構成する場合

SLP はスコープ構成をまったく行わなくても十分機能します。Solaris オペレーティング環境では、SLP のデフォルトのスコープは `default` です。構成されているスコープがない場合は、`default` がすべての SLP メッセージのスコープになります。

次の環境のどれかに当てはまれば、スコープを構成できます。

- サポートしている組織が、所属メンバーに対するサービス通知アクセスを制限する場合
- サポートしている組織が、特定のユーザーが特定領域のサービスにアクセスするように物理的に配置されている場合
- ユーザーが認識できるサービス通知を分割する必要がある場合

最初の場合の例を 273 ページの「ダイアルアップネットワークに対する DA 検出の構成」 に挙げました。2 番目の例は、組織が 2 つの建物に分かれていて、1 つの建物のユーザーはその建物のローカルサービスにアクセスするようにしたい場合です。ビルディング 1 のユーザーはスコープ `B1` を使用して、ビルディング 2 のユーザーはスコープ `B2` を使って構成できます。

スコープを構成する場合の検討事項

`slpd.conf` ファイル内の `net.slp.useScopes` プロパティを変更する場合は、ホスト上のすべてのエージェントにスコープを構成します。ホストが SA を実行している場合や DA として機能している場合に、その SA と DA に `default` 以外のスコープを構成するには、このプロパティを構成する必要があります。UA だけがマシン上で動作し、UA が、`default` 以外のスコープをサポートしている SA と DA を検出する必要がある場合は、UA が使用するスコープを制限するのでなければ、プロパティを構成する必要はありません。プロパティを構成しない場合、UA は、`slpd` を通じて、使用可能な DA とスコープを自動的に検出します。SLP デーモンは、能動的および受動的 DA 検出を使用して DA を見つけるか、DA が動作していない場合は SA 検出を使用して DA を見つけます。プロパティを構成する場合、UA は構成されたスコープを使用するだけで、構成されたスコープを破棄することはありません。

スコープを構成することを決定した場合、ネットワークのすべての SA にスコープが構成されていることを確認できなければ、構成されたスコープのリストに `default` スコープを保存することを考えてください。構成されていない SA があると、構成されたスコープを持つ UA はそれらの SA を見つけることができません。これは、構成されていない SA が自動的に `default` スコープを持つのに対し、UA は構成されたスコープを持つためです。

`net.slp.DAAddresses` プロパティを設定して DA も構成しようとする場合は、構成される DA によってサポートされるスコープが、`net.slp.useScopes` プロパティで構成したスコープと同じであることを確認してください。スコープが同じでない場合は、再起動時に `slpd` がエラーメッセージを出力します。

▼ スコープの構成方法

次の手順に従って、スコープ名を `slp.conf` ファイルの `net.slp.useScopes` プロパティに追加します。

1. スーパーユーザーになります。
2. ホスト上の `slpd` とすべての **SLP** 動作を停止します。

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

3. 構成の設定を変更する前に、デフォルトの `/etc/inet/slp.conf` ファイルのバックアップをとります。
4. `slpd.conf` ファイル内の `net.slp.useScopes` プロパティを変更します。

   ```
   net.slp.useScopes=<scope names>
   ```

scope names	文字列のリストで、DA または SA が要求時に使用を許されるスコープを示すか、DA がサポートする必要があるスコープを示す デフォルトの値は、SA と DA の場合は Default、UA の場合は未設定

注 – スコープ名は、次の文法上のガイドラインに従って構成します。

- 大文字または小文字の英数字
- 句読点 (’’、 \、!、<、 =、>、および ~ を除く)
- 名前の一部と考えられるスペース
- 非 ASCII 文字

  ASCII でない文字をエスケープするには、バックスラッシュを使用します。たとえば、UTF-8 コード体系では、フランス語の *aigue* アクセントのある文字 *e* を表すために、16 進コード `0xc3a9` を使用します。プラットフォームが UTF-8 をサポートしていない場合は、UTF-8 の 16 進コード `\c3\a9` をエスケープシーケンスとして使用します。

ここでは、例として、`bldg6` にスコープ `eng` と `mktg` を作成することを考えます。この場合は、`net.slp.useScopes` 行を次のように変更します。

```
net.slp.useScopes=eng,mktg,bldg6
```

5. 変更を保存し、ファイルを閉じます。
6. 変更を反映するには、`slpd` を再起動します。

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

DA の配置

この節では、SLP を実行しているネットワークでの計画的な DA の配置について説明します。

配置された DA または構成されたスコープがなくても、基本のエージェントである UA と SA だけで SLP は十分機能します。特定の構成を持たないすべてのエージェントは自動的に `default` スコープを使用します。DA はサービス通知のキャッシュとして機能します。DA を配置すると、ネットワークに送られるメッセージ数が削減されるため、メッセージ応答の受け取りに必要な時間も短縮されます。これにより、SLP をより大規模なネットワークに対応させることができます。

SLP DA を配置する理由

DA を配置する主な目的は、サービス検出によって生じるマルチキャストトラフィックの量とユニキャスト応答の収集に関係する遅延を削減することです。多くの UA と SA を持つ大規模なネットワークでは、サービス検出によって生じるマルチキャストの

量が非常に大きくなるので、ネットワークのパフォーマンスが低下します。1 つまたは複数の DA を配置すると、UA はサービスについて DA にユニキャストし、SA はユニキャストを使用して DA に登録する必要があります。DA を使用したネットワークでは、SLP 登録されたマルチキャストは、能動的および受動的 DA 検出のマルチキャストだけです。

SA は、マルチキャストのサービス要求を受け取るのではなく、共通のスコープのセット内で検出した任意の DA に自動的に登録します。ただし、DA がサポートしていないスコープ内のマルチキャスト要求には、SA が直接応答します。

UA から出されたサービス要求は、UA のスコープ内に DA が配置されている場合は、ネットワーク上へのマルチキャストではなく DA に対するユニキャストです。そのため、UA のスコープ内に DA を配置すると、マルチキャストが削減されます。通常の UA 要求を行うマルチキャストをなくすことにより、クエリー応答の受け取りに必要な時間が秒単位からミリ秒単位に大幅に縮小します。

DA は SA および UA の動作の中心として機能します。スコープの集合に対して 1 つまたは複数の DA を配置することにより、SLP の動作を監視するための集中的なポイントが提供されます。DA ログを起動することにより、ネットワークに散在している複数の SA から取り寄せたログをチェックするよりも、登録および要求の監視が容易になります。負荷を均等にする必要に合わせて、1 つまたは複数の特定のスコープに対して DA をいくつでも配置できます。

マルチキャストルーティングが使用できないネットワークでは、SLP がブロードキャストを使用するように構成できます。しかし、ブロードキャストは各ホストにメッセージを処理するように要求するため、非常に効率が悪くなります。また、ブロードキャストは通常、ルーターを超えて伝達されません。この結果、マルチキャストルーティングに対応していないネットワークでは、同じサブネットでしかサービスを検出できません。マルチキャストルーティングに一部しか対応していない場合は、ネットワーク上でサービスを検出する機能に矛盾が生じます。マルチキャストメッセージは DA の検出に使用されます。したがって、マルチキャストルーティングに一部しか対応していない場合は、UA と SA はサービスを SA のスコープ内にある既知の DA に登録することが暗黙の了解になっています。たとえば、UA が `DA1` と呼ばれる DA にクエリーを出し、SA がサービスを `DA2` に登録している場合、UA はサービスの検出に失敗します。マルチキャストが使用できないネットワーク上の SLP の配置については、280 ページの「ブロードキャスト専用ルーティングの構成」 を参照してください。

サイト全体がマルチキャストルーティングに対応していないネットワークでは、`net.slp.DAAdresseses` プロパティを使用して、SLP の UA と SA が DA 位置に関して矛盾のないリストを持つように構成する必要があります。

最後に、Solaris SLPv2 の DA は SLPv1 との相互運用性をサポートしています。SLPv1 の相互運用は Solaris DA ではデフォルトで有効になっています。ネットワークにプリンタなどの SLPv1 デバイスが接続されている場合、またはサービス検出で SLPv1 を使用している Novell Netware 5 と相互運用する必要がある場合、DA を配置する必要があります。DA が配置されていないと、Solaris SLP の UA は SLPv1 によって通知されたサービスを見つけることができません。

DA を配置する場合

次の条件のどれかに当てはまる場合は、エンタープライズに DA を配置します。

- snoop で測定した、ネットワーク上での SLP のマルチキャストのトラフィックが帯域幅の 1% を超える
- UA クライアントがサービス要求のマルチキャスト中に長時間遅延またはタイムアウトする
- 1 台または複数台のホスト上にある特定のスコープに対して、SLP サービス通知の監視を集中する
- ネットワークが、サービスを共有する複数のサブネットから構成され、マルチキャストに対応していない
- ネットワークが前バージョンの SLP (SLPv1) をサポートするデバイスを使用している、または SLP サービス検出で Novell Netware 5 と相互運用したい

▼ DA を配置する方法

次の手順に従って、slp.conf ファイルの net.slp.isDA プロパティに True を設定します。

注 – 1 台のホストにつき 1 つの DA だけが割り当てられます。

---

1. スーパーユーザーになります。
2. ホスト上の slpd とすべての **SLP** 動作を停止します。

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

3. 構成の設定を変更する前に、デフォルトの /etc/inet/slp.conf ファイルのバックアップをとります。
4. slpd.conf ファイル内の net.slp.isDA プロパティに True を設定します。

   ```
   net.slp.isDA=True
   ```

5. 変更を保存し、ファイルを閉じます。
6. 変更を反映するには、slpd を再起動します。

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

DA を配置する場所

この節は、DA を配置する場所について状況ごとにヒントを示します。

- マルチキャストルーティングが使用できず、DA がサブネット間のサービス検出をブリッジする必要がある場合

  この場合は、インタフェースとサービスを共有するすべてのサブネットを持つホスト上に DA を配置してください。IP パケットがインタフェースの間を経路指定されない場合を除き、`net.slp.interfaces` 構成プロパティを設定する必要はありません。`net.slp.interfaces` プロパティの設定については、292 ページの「マルチホームの構成」 を参照してください。

- DA が拡張に備えて配置されており、考慮すべき主要な事柄がエージェントのアクセスの最適化である場合

  UA は通常、DA に対してサービスを大量に要求します。SA がサービスを DA に登録すると、SA は通知を定期的に適切な頻度で更新できます。その結果、UA から DA へのアクセスの方が SA のアクセスよりはるかに頻繁になります。通常、サービス通知の数も要求の数より小さくなります。このため、UA のアクセスに対して DA の配置が最適化されている場合、多くの DA を配置することは効率化をうながします。

- UA のアクセスを最適化するために、ネットワーク上でトポロジ的に UA の近くになるように DA を配置する場合

  UA クライアントと SA クライアントの両方が共有しているスコープを使用して、DA を構成してください。

複数の DA を配置して負荷を均等にする

負荷を均等にする手段として、同じスコープの集合体について複数の DA を配置できます。次の状況のどれかに当てはまれば、DA を配置できます。

- DA に対する UA 要求がタイムアウトしているか、あるいは `DA_BUSY_NOW` エラーが返っている
- DA ログが、多くの SLP 要求が欠落していることを示す
- スコープ内でサービスを共有しているユーザーのネットワークが、複数の建物や物理的なサイトに渡っている

SLP トラフィックの `snoop` トレースを行うことによって、どれくらいの UA 要求で `DA_BUSY_NOW` エラーが返されるかを判断することができます。返される UA 要求の数が多い場合は、DA から物理的およびトポロジ的に離れている建物内の UA は、応答が遅かったり過度にタイムアウトしたりすることがあります。このような場合、建物内の UA クライアントの応答を改善するために、建物ごとに DA を配置できます。

建物に接続しているリンクは、建物内のローカルエリアネットワークよりも遅いことがあります。ネットワークが複数の建物または物理的なサイトに渡っている場合は、`/etc/inet/slp.conf` ファイル内の `net.slp.DAAddresses` プロパティを特定のホスト名またはアドレスのリストに設定して、指定した DA だけに UA がアクセスするようにします。

特定の DA がサービス登録に対して大量のホストメモリーを消費している場合は、DA がサポートするスコープ数を減らすことによって、SA 登録の数を削減します。登録数の多いスコープを 2 つのスコープに分け、別のホストに新たに DA を配置して、新しく作成した 2 つのスコープの 1 つをサポートすることができます。

マルチホーム

マルチホームサーバーは、複数の IP サブネット上でホストとして機能します。そのようなサーバーに複数のネットワークインタフェースカードが装着されると、ルーターとして機能できます。マルチキャストパケットを含む IP パケットは、このインタフェース間を経路指定されます。場合によっては、インタフェース間の経路指定ができないことがあります。この節では、そのような場合に SLP を構成する方法について説明します。

マルチホームの構成

構成を行わない場合、デフォルトのネットワークインタフェース上で、`slpd` はマルチキャストと UDP/TCP ユニキャストに対して待機しています。ユニキャストルーティングとマルチキャストルーティングがマルチホームマシンのインタフェース間で使用できる場合は、追加の構成を行う必要はありません。追加の構成が必要ないのは、別のインタフェースに到達するマルチキャストパケットがデフォルトで正確に経路指定されているからです。その結果、DA または他のサービス通知のマルチキャスト要求は、`slpd` に届きます。経路指定がなんらかの理由で調整されていない場合は、構成が必要です。

経路指定されていない複数のネットワークインタフェースに対して構成を行う場合

マルチホームマシンの構成が必要と考えられるのは、主に次の場合です。

- ユニキャストルーティングはインタフェース間で使用できるが、マルチキャストルーティングは使用できない
- ユニキャストルーティングとマルチキャストルーティングの両方がインタフェース間で使用できない

マルチキャストルーティングがインタフェース間で使用できない場合は、通常、マルチキャストがネットワークに配置されていないことが原因です。この場合は通常、それぞれのサブネット上の DA ベースでないサービス検出および DA 検出にブロードキャストが使用されます。ブロードキャストは、`net.slp.isBroadcastOnly` プロパティを `True` に設定することによって構成します。

経路指定されていない複数のネットワークインタフェースの構成 (作業マップ)

表 **20–5** 経路指定されていない複数のネットワークインタフェースの構成

作業	説明	参照先
net.slp.interfaces プロパティを構成する	このプロパティを設定することで、slpd は、指定されたインタフェース上でユニキャストとマルチキャスト/ブロードキャストの SLP 要求を待機できる	293 ページの「net.slp.interfaces プロパティの構成」
サブネット上の UA が到達可能なアドレスを持つサービス URL を取得できるように、プロキシサービス通知を配置する	マルチホームホストではなく単一のサブネットに接続された slpd を実行しているマシンにプロキシ通知を限定する	295 ページの「マルチホームホスト上のプロキシ通知」
UA と SA 間で確実に到達できるように DA を配置してスコープを構成する	マルチホーム上の net.slp.interfaces プロパティを単一インタフェースのホスト名またはアドレスで構成する。 マルチホームホスト上で DA を実行し、各サブネット上の SA と UA は別のホストを使用するように構成する	296 ページの「DA の配置とスコープ名の割り当て」

net.slp.interfaces プロパティの構成

net.slp.interfaces プロパティが設定されている場合、slpd は、ユニキャストとマルチキャスト/ブロードキャストの SLP 要求を、デフォルトのインタフェース上ではなく、プロパティにリストされたインタフェース上で待機します。

通常、net.slp.interfaces プロパティを設定すると同時に、net.slp.isBroadcastOnly プロパティも設定することでブロードキャストを有効にします。これは、ネットワークにマルチキャストが配置されていないために行います。ただし、マルチキャストは配置されているが、この特定のマルチホームホスト上で経路指定されていない場合、マルチキャスト要求は、複数のインタフェースから slpd に到達できます。このような状況は、パケットの経路指定が、別のマルチホームホストまたはインタフェースからサービスを受けるサブネットに接続されているルーターによって制御されている場合に起こります。

この場合、SA サーバーまたは要求を送っている UA は、マルチホームの slpd から 2 つの応答を受け取ります。これらの応答はクライアントライブラリによってフィルタにかけられて除かれるので、クライアントには見えません。ただし、この応答は、snoop トレースで見ることができます。

注 – ユニキャストルーティングがオフになっている場合、マルチホームホスト上の SA クライアントによるサービス通知がすべてのサブネットに到達できないことがあります。サービスが到達できない場合、SA クライアントは次のことを実行できます。

- 個々のサブネットにつき 1 つのサービス URL を通知する
- 特定のサブネットからの要求が到達可能な URL で確実に応答されるようにする

SA クライアントライブラリには、到達可能な URL が確実に通知されるようにするためのしくみはありません。したがって、到達可能な URL が確実に通知されるようにするには、経路指定のないマルチホームホストを処理できるかどうかにかかわらず、サービスプログラムに任せる必要があります。

ユニキャストルーティングが無効なマルチホームホストにサービスを配置する前に、`snoop` を使ってサービスが複数のサブネットから出された要求を正確に処理しているかどうかを判断してください。さらに、マルチホームホストに DA を配置することを計画している場合は、296 ページの「DA の配置とスコープ名の割り当て」 を参照してください。

▼ `net.slp.interfaces` プロパティの構成方法

次の手順に従って、`slp.conf` ファイルの `net.slp.interfaces` プロパティを変更します。

1. スーパーユーザーになります。
2. ホスト上の `slpd` とすべての **SLP** 動作を停止します。

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

3. 構成の設定を変更する前に、デフォルトの `/etc/inet/slp.conf` ファイルのバックアップをとります。
4. `slpd.conf` ファイル内の `net.slp.interfaces` プロパティを変更します。

   ```
   net.slp.interfaces=value
   ```

value	IPv4 アドレスまたはネットワークインタフェースカードのホスト名のリストで、そこに存在する DA や SA はポート 427 上でマルチキャスト、ユニキャスト UDP、および TCP の各メッセージを待機する必要がある

たとえば、3 枚のネットワークカードを持ち、マルチキャストルーティングがオフになっているサーバーが、3 つのサブネットに接続されているとします。さらに、その 3 つのネットワークインタフェースが IP アドレス `192.147.142.42`、`192.147.143.42`、および `192.147.144.42` を持っており、サブネットマスクが

255.255.255.0 であるとします。次のプロパティの設定を行うと、slpd は、3 つすべてのインタフェース上でユニキャストとマルチキャスト/ブロードキャストのメッセージを待機します。

```
net.slp.interfaces=192.147.142.42,192.147.143.42,192.147.144.42
```

注 – net.slp.interfaces プロパティには、IP アドレスまたは解決可能なホスト名を設定できます。

---

5. 変更を保存し、ファイルを閉じます。
6. 変更を反映するには、slpd を再起動します。

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

マルチホームホスト上のプロキシ通知

複数のインタフェースを持つホストが slpd およびプロキシ登録を使ってサービスを通知する場合は、slpd によって通知されるサービス URL に到達可能なホスト名またはアドレスが含まれている必要があります。インタフェース間でユニキャストルーティングが有効な場合は、すべてのサブネット上のホストは別のサブネット上のホストに到達できます。任意のサブネット上のサービスに対してプロキシ登録も行うことができます。ただし、ユニキャストルーティングが無効な場合は、1 つのサブネット上のサービスクライアントはマルチホームホストを通じて別のサブネット上のサービスに到達することはできません。ただし、別のルーターを通じて到達できることはあります。

たとえば、デフォルトのホスト名が bigguy のホストが、経路指定されていない異なる 3 つのサブネット上に 3 枚のインタフェースカードを持っているとします。これらのサブネット上のホスト名は、IP アドレス 192.147.142.42 を持つ bigguy、IP アドレス 192.147.143.42 を持つ bigguy1、IP アドレス 192.147.144.42 を持つ bigguy2 です。ここで、レガシープリンタ oldprinter がサブネット 143 に接続され、すべてのインタフェース上で待機するために、URL service:printing:lpr://oldprinter/queue1 が net.slp.interfaces で構成されているとします。oldprinter の URL はすべてのインタフェース上でプロキシ通知されます。サブネット 142 と 144 上のマシンは、サービス要求に対する応答でこの URL を受信しますが、oldprinter サービスにアクセスすることはできません。

この問題の解決方法は、マルチホームホスト上ではなく、サブネット 143 だけに接続されたマシン上で動作している slpd を使ってプロキシ通知を行うことです。サブネット 143 上のホストだけがサービス要求に対する応答でこの通知を取得できます。

DA の配置とスコープ名の割り当て

マルチホームホストを持つネットワーク上で DA の配置とスコープ名の割り当てを行う場合は、クライアントがアクセス可能なサービスを確実に取得できるように注意してください。経路指定が無効で `net.slp.interfaces` プロパティが構成されている場合は特に注意してください。また、マルチホームマシン上のインタフェース間でユニキャストルーティングが有効な場合は、特別な DA やスコープを構成する必要はありません。これは、DA にキャッシュされている通知が任意のサブネットからアクセス可能なサービスを識別するためです。ただし、ユニキャストルーティングが無効な場合は、DA をうまく配置しないと問題になることがあります。

前述の例で何が問題になりうるかを見るために、`bigguy` が DA を実行し、すべてのサブネット上のクライアントが同じスコープを持つ場合に何が起こるかを考えてみます。サブネット `143` 上の SA はサービス通知を DA に登録します。サブネット `144` 上の UA は、サブネット `143` 上のホストに到達できなくても、それらのサービス通知を入手できます。

この問題の 1 つの解決方法は、マルチホームホスト上ではなく、各サブネット上で DA を実行することです。この場合は、マルチホームホスト上の `net.slp.interfaces` プロパティを、単一のインタフェースホスト名またはアドレスを使って構成するか、構成しないでそのままにし、強制的にデフォルトのインタフェースを使用するようにします。この解決方法の欠点は、通常大規模なマシンであり、DA として高機能であるマルチホームホストを DA に設定できないことです。

もう 1 つの解決方法は、マルチホームホスト上で DA を実行するが、各サブネット上の SA および UA が異なるスコープを持つようにスコープを構成することです。たとえば、前述の場合、`142` サブネット上の UA と SA がスコープ `scope142` を持つようにスコープを構成することができます。`143` サブネット上の UA と SA は、`scope143` という別のスコープを持ち、`144` サブネット上の UA と SA は `scope144` という別のスコープを持つように構成することができます。3 つのインタフェースを持つ `bigguy` 上の `net.slp.interfaces` プロパティを構成して、DA を 3 つのサブネット上の 3 つのスコープに作用させることができます。

経路指定されていない複数のネットワークインタフェースを構成する場合の検討事項

マルチホームホスト上の DA がサブネット間のサービス通知をブリッジできるように `net.slp.interfaces` プロパティを構成することができます。このような構成は、ネットワークでマルチキャストルーティングがオフで、マルチホームホスト上のインタフェース間でユニキャストルーティングが有効な場合に便利です。ユニキャストはインタフェース間を経路指定しているため、サービスが置かれているサブネットと異なるサブネット上のホストは、サービス URL を受信すればそのサービスに接続することができます。DA がない場合は、特定のサブネット上の SA サーバーが同じサブネット上に出されたブロードキャストだけを受信するので、そのサブネット以外にサービスを置くことはできません。

net.slp.interfaces プロパティの構成が必要な場合は、マルチキャストがネットワークに配置されておらず、代わりにブロードキャストが使用されている場合です。その他の場合は、不必要な応答の重複や到達できないサービスを避けるために、入念に検討および計画を行ってください。

第 21 章

レガシーサービスの組み込み

レガシーサービスとは、SLP の開発および実装が旧式になっているネットワークサービスのことです。たとえば、ラインプリンタデーモン (`lpsched`)、NFS ファイルサービス、NIS または NIS+ ネームサービスなどの Solaris サービスは、SLP で使用する内部 SA を持っていません。この章では、レガシーサービスを通知する場合とその方法について説明します。

レガシーサービスを通知する場合

レガシーサービス通知では、SLP UA を使用可能にすることで、ネットワーク上の次のようなデバイスやサービスを検出できます。

- SLP SA を含まないハードウェアデバイスやソフトウェアサービス。たとえば、SLP UA を持つアプリケーションが、SLP SA を含まないプリンタやデータベースを検出する必要がある場合、レガシー通知が必要になります。

レガシーサービスの通知

レガシーサービスは、次の方法で通知できます。

- SLP SA を組み込むようにサービスを変更する
- SLP が有効でないサービスの代わりにサービスを通知する小さなプログラムを書く
- プロキシ通知を使用して、`slpd` にサービスを通知させる

サービスの変更

ソフトウェアサーバーのソースコードを使用できる場合は、SLP SA を組み込むことができます。SLP 用の C 言語の API と Java の API は比較的簡単に使用できます。C 言語の API のマニュアルページと Java の API のマニュアルを参照してください。サービスがハードウェアデバイスの場合は、製造元が SLP を組み込む PROM を更新していることがあります。詳細は、デバイスの製造元に問い合わせてください。

SLP が使用できないサービスの通知

ソースコードや更新された SLP を含む PROM が使用できない場合は、SLP クライアントライブラリを使ってサービスを通知する小さなアプリケーションを書くことができます。このアプリケーションは小さなデーモンとして機能し、サービスの起動・停止に使用する場合と同じシェルスクリプトで起動・停止します。

SLP プロキシ登録

Solaris `slpd` は、プロキシ登録ファイルを使用したレガシーサービスの通知をサポートしています。プロキシ通知ファイルは、移植性のあるフォーマットで書かれたサービス通知のリストです。

▼ SLP プロキシ登録を有効にする方法

1. ホストのファイルシステムまたは **HTTP** でアクセス可能なネットワーク上の任意のディレクトリに、プロキシ登録ファイルを作成します。
2. サービスについてサービスタイプのテンプレートが存在するかどうかを確認します。

   テンプレートは、サービスタイプのサービス URL と属性を記述したものです。テンプレートを使用して、特定のサービスタイプについて通知の構成要素を定義します。

   - サービスタイプテンプレートが存在する場合は、そのテンプレートを使ってプロキシ登録を構成してください。サービスタイプテンプレートについては、RFC 2609 を参照してください。
   - サービスについてサービスタイプテンプレートを使用できない場合は、サービスを正確に記述する属性の集合体を選択してください。通知に対して、デフォルト以外の命名権限を使用してください。デフォルトの命名権限は標準化されたサービスタイプについてだけ許可されてます。命名権限については、RFC 2609 を参照してください。

     たとえば、*BizApp* という会社にソフトウェアバグの追跡に使用されるローカルデータベースがあるとします。データベースを通知するために、この会社は、サービスタイプ `service:bugdb.bizapp` を持つ URL を使用します。この場合、命名権限は `bizapp` になります。

3. 前の手順で作成した登録ファイルの場所を使用して、/etc/inet/slp.conf ファイルの net.slp.serializedRegURL プロパティを構成するには、以下の手順に従います。

4. スーパーユーザーになります。

5. ホスト上の slpd とすべての **SLP** 動作を停止します。

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

6. 構成の設定を変更する前に、デフォルトの /etc/inet/slp.conf ファイルのバックアップをとります。

7. /etc/inet/slp.conf ファイルの net.slp.serializedRegURL プロパティにプロキシ登録の場所を指定します。

```
net.slp.net.slp.serializedRegURL=proxy registration file URL
```

たとえば、直列化登録ファイルが /net/inet/slp.reg である場合、プロパティを次に示すように構成します。

```
net.slp.serializedRegURL=file:/etc/inet/slp.reg
```

8. 変更を保存し、ファイルを閉じます。

9. 変更を反映するには、slpd を再起動します。

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

SLP プロキシ登録による通知

サービス通知は、サービス URL を特定する行、オプションのスコープ行、一連の属性の定義から構成されます。SLP デーモンはファイルからプロキシ通知を読み、その通知を登録し、SA クライアントと同じようにそれらを保持します。次のリストは、プロキシ登録ファイルの例を示します。

この例では、LPR プロトコルをサポートするレガシープリンタと ftp サーバーが通知されています。行番号は説明のために付け加えたもので、実際のファイルには記述されていません。

```
1#Advertise legacy printer.
2
3service:lpr://bizserver/mainspool,en,65535
4scope=eng,corp
5make-model=Laserwriter II
6location-description=B16-2345
7color-supported=monochromatic
8fonts-supported=Courier,Times,Helvetica 9 10
9
10#Advertise FTP server
11
12ftp://archive/usr/src/public,en,65535,src-server
```

```
[13]content=Source code for projects
[14]
```

注 – プロキシ登録ファイルは、ASCII でない文字のエスケープに、構成ファイルと同じ取り決めを使用します。プロキシ登録ファイルのフォーマットについては、RFC 2614 を参照してください。

表 **21–1** SLP プロキシ登録ファイルの説明

行番号	説明
1 と 10	シャープ記号 (#) で始まるコメント行で、ファイルの動作には影響しない。コメント行の最後まですべての文字が無視される
2、9、14	通知の区切りを示す空行
3、12	3 つの必須フィールドと 1 つのオプションフィールドがコンマで区切られたサービス URL ■ 一般的な URL または `service: URL` が通知される。service: URL の指定方法の仕様については、RFC 2609 を参照 ■ 通知の言語を指定する。前述の例では、フィールドは英語 *en* を指定している。この言語は RFC 1766 の言語タグである ■ 登録の有効期限を秒単位で規定する。有効期限は符号なしの 16 ビット整数に限定される。有効期限が最大値 65535 より小さい場合、`slpd` は通知をタイムアウトする。有効期限が 65535 の場合、`slpd` は定期的に通知を更新し、`slpd` が存在するかぎり有効期限は永続するとされる ■ サービスタイプフィールド (省略可) – サービスタイプの定義に使用する。サービス URL が定義されている場合は、URL が通知されるサービスタイプを変更できる。前述のプロキシ登録ファイルの例では、12 行目に一般的な FTP URL が含まれている。オプションのタイプフィールドを使用して、この URL をサービスタイプ名 *src-server* で通知できる。デフォルトでは `service` 接頭辞はタイプ名には付かない
4	スコープの指定 オプション行はトークン `scope` と等号、およびコンマで区切られたスコープ名のリストで構成される。このスコープ名は、`net.slp.useScopes` 構成プロパティで定義されている。ホストに構成されたスコープだけが、このスコープリストに表示される。スコープ行がない場合は、`slpd` が構成されているすべてのスコープに登録が行われている。スコープ行は URL 行のすぐ後になければならない。その他の場所にある場合、スコープ名は属性として認識される

表 **21–1** SLP プロキシ登録ファイルの説明　(続き)

行番号	説明
5 から 8	属性の定義 オプションのスコープ行の後は、サービス通知の大部分は属性と値リストのペアの行で構成される。各ペアは属性タグ、等号、コンマで区切られた属性値のリスト (属性が単一値の場合は単一値) で構成される。前述のプロキシ登録ファイルの例では、8 行目が複数の値を持つ属性リストを示している。これ以外の値リストはすべて単一値を持っている。属性名および値のフォーマットは、ネットワークを通過する SLP メッセージと同じである

レガシーサービスを通知する場合の検討事項

通常、SLP を追加する場合、他のサービスの代理として SLP API で通知する SLP 対応のサービスを書くよりも、ソースコードを変更する方が望ましい方法です。ソースコードの変更は、プロキシ登録を使用するよりも望ましい方法です。ソースコードを変更する場合、サービス固有の機能を追加したり、サービスの使用可否を綿密に追跡したりできます。ソースコードが使用できない場合は、プロキシ登録を使用するより他のサービスの代理として通知する SLP 対応のヘルパーサービスを書く方が望ましい方法です。このヘルパーサービスを、起動と停止の制御に使用されるサービスの開始または停止手順に組み込むことをお勧めします。プロキシ通知は通常、ソースコードが使用できず、スタンドアロンの SA を書くことが実際的ではない場合の 3 番目の選択肢です。

プロキシ通知は、プロキシ登録ファイルを読み取る `slpd` が動作している間だけ保持されます。プロキシ通知とサービスの間には直接的な関係はありません。通知がタイムアウトしたり `slpd` が停止したりすると、プロキシ通知は使用できなくなります。

サービスが停止した場合は、`slpd` を停止する必要があります。直列化登録ファイルを編集してプロキシ通知をコメントにするか削除し、`slpd` を再起動してください。サービスを再起動または再インストールしたときは同じ手順に従ってください。プロキシ通知とサービスの間に関係のないことがプロキシ通知の主な欠点です。

第 22 章

SLP (リファレンス)

この章では、SLP のステータスコードとメッセージタイプについて説明します。SLP のメッセージタイプは、省略形と機能コードを示します。SLP のステータスコードは、説明と機能コードを示します。ステータスコードは、該当する要求を受信しているか (コード 0)、受信側がビジーであるかを示します。

注 – SLP デーモン (`slpd`) は、ユニキャストメッセージに対してだけステータスコードを返します。

SLP のステータスコード

表 **22–1** SLP のステータスコード

ステータスのタイプ	ステータスコード	説明
No Error	0	要求はエラーなしで処理された
LANGUAGE_NOT_SUPPORTED	1	AttrRqst または SrvRqst について、スコープ内にサービスタイプのデータがあるが、指定された言語ではない
PARSE_ERROR	2	メッセージが SLP 構文に従っていない
INVALID_REGISTRATION	3	SrvReg に問題がある。たとえば、有効期限がゼロである、言語タグが欠けているなど

表 **22–1** SLP のステータスコード　(続き)

ステータスのタイプ	ステータスコード	説明
SCOPE_NOT_SUPPORTED	4	SLP メッセージが、要求に応える SA または DA がサポートするスコープリスト内のスコープを含んでいなかった
AUTHENTICATION_UNKNOWN	5	DA または SA がサポートしていない SLP SPI に対する要求を受信した
AUTHENTICATION_ABSENT	6	UA または DA が SrvReg において URL および属性認証を要求したが受信しなかった
AUTHENTICATION_FAILED	7	UA または DA が認証ブロックにおいて認証エラーを検出した
VER_NOT_SUPPORTED	9	メッセージでサポートしていないバージョン番号
INTERNAL_ERROR	10	DA または SA で未知のエラーが発生した。たとえば、オペレーティングシステムがファイルスペースを使い果たした
DA_BUSY_NOW	11	UA または SA は、急増するバックオフを使用して再試行する必要がある。DA が他のメッセージの処理でビジー状態である
OPTION_NOT_UNDERSTOOD	12	DA または SA が必須の範囲から未知のオプションを受信した
INVALID_UPDATE	13	DA が登録されていないサービスに対して、FRESH 設定なしで、あるいは矛盾するサービスタイプで、SrvReg を受信した
MSG_NOT_SUPPORTED	14	SA が AttrRqst または SrvTypeRqst を受信したが、サポートしていない
REFRESH_REJECTED	15	SA が DA に対して、DA の最短更新間隔よりも頻繁に SrvReg または SrvDereg の一部を送った

SLP のメッセージタイプ

表 **22–2** SLP のメッセージタイプ

メッセージタイプ	省略形	機能コード	説明
サービス要求	`SrvRqst`	1	サービスを検出するために UA が発行する。あるいは、能動的 DA 検出において UA あるいは SA サーバーが発行する
サービス応答	`SrvRply`	2	DA あるいは SA がサービス要求に対して応答する
サービス登録	`SrvReg`	3	SA が新規の通知を登録したり、既存の通知を新規の属性および変更された属性で更新したり、URL の有効期限を更新できるようにしたりする
サービス登録解除	`SrvDereg`	4	表しているサービスが無効になった場合にその通知の登録を解除するために SA が使用する
確認応答	`SrvAck`	5	SA のサービス要求またはサービス登録解除メッセージに対する DA の応答
属性要求	`AttrRqst`	6	URL またはサービスタイプが作成し、属性のリストを要求する
属性応答	`AttrRply`	7	属性のリストを返す場合に使用される
DA 通知	`DAAdvert`	8	サービス要求をマルチキャストするための DA の応答
サービスタイプ要求	`SrvTypeRqst`	9	特定の命名権限を持ち、特定のスコープセットにある登録されたサービスタイプについて問い合わせるために使用される
サービスタイプ応答	`SrvTypeRply`	10	サービスタイプ要求に対する応答として返されるメッセージ
SA 通知	`SAAdvert`	11	DA が配置されていないネットワークで、UA は `SAAdvert` を使用して SA およびそのスコープを検出する

第 23 章

メールサービス (トピック)

第 24 章	メールサービスの概要
第 25 章	メールサービスの設定と障害追跡 (トラブルシューティング)
第 26 章	メールサービスの背景情報について
第 27 章	メールサービスの新機能情報について

第 24 章

メールサービス (概要)

電子メールサービスの設定と維持は複雑な作業ですが、どちらもネットワークの日々の運用に必要不可欠です。ネットワーク管理者は、既存のメールサービスを拡張したり、新規のネットワークやサブネットワークにメールサービスを設定する必要があります。メールサービスに関する各章では、ネットワークでメールサービスを計画したり設定したりするために必要な情報を提供します。この章では、sendmail の新機能および参考資料を紹介します。この章ではまた、メールサービスを確立するために必要なソフトウェアおよびハードウェアコンポーネントの概要を説明します。

- 311 ページの「sendmail バージョン 8.12 の新機能」
- 312 ページの「メールサービスのコンポーネントの概要」

第 25 章では、メールサービスの設定および管理方法の手順を説明します。詳細は、316 ページの「メールサービス (作業マップ)」を参照してください。

第 26 章では、メールサービスのコンポーネントについて詳しく説明します。 また、この章では、メールサービスのプログラムとファイル、メールルーティング処理、ネームサービスを使った sendmail の対話型操作についても説明します。

sendmail バージョン 8.12 の新機能

今回の Solaris 9 リリースには sendmail のバージョン 8.12 が含まれています。第 27 章 では、すべての新機能について説明します。以下に、sendmail の重要な変更点の一部を紹介します。

- 400 ページの「新しい構成ファイル submit.cf」
- 402 ページの「コマンド行の新しいオプションまたは推奨されないオプション」
- 403 ページの「構成ファイルの新しい構成オプションと改訂された構成オプション、および関連トピック」
- 420 ページの「sendmail における新しい m4 構成マクロおよび改訂された m4 構成マクロ」

- 424 ページの「配信エージェントの新しいフラグ」
- 424 ページの「配信エージェントの新しい等号 (=)」
- 429 ページの「ファイルへの変更」

第 27 章 では、上記以外の次の変更点についても説明します。

- 429 ページの「`mail.local` の変更点」
- 430 ページの「`mailstats` の変更点」
- 431 ページの「`makemap` の変更点」
- 431 ページの「新しいコマンド `editmap`」
- 432 ページの「他の変更点および機能」

その他の `sendmail` の情報源

次に、上記以外の `sendmail` 関連の参考資料を示します。

- 『*sendmail, Second Edition*』Costales、Bryan 著、O'Reilly & Associates, Inc. 発行、1997
- `sendmail` 関連のホームページ – http://www.sendmail.org
- `sendmail` 関連の FAQ – http://www.sendmail.org/faq
- 新しい `sendmail` 構成ファイルの README – http://www.sendmail.org/m4/readme.html
- `sendmail` の最新 Sun バージョン 8.* への移行ガイド – http://www.sendmail.org/vendor/sun/

メールサービスのコンポーネントの概要

メールサービスを確立するためには、多くのソフトウェアコンポーネントおよびハードウェアコンポーネントが必要になります。次の節ではこれらのコンポーネントとその説明に使用されるいくつかの用語について簡単に説明します。

最初の節 312 ページの「ソフトウェアコンポーネントの概要」 では、メール配信システムのソフトウェア部分を説明するのに使用する用語を定義します。その次の節 313 ページの「ハードウェアコンポーネントの概要」 では、メール構成におけるハードウェアシステムの機能について取り上げます。

ソフトウェアコンポーネントの概要

次の表にメールシステムのソフトウェアコンポーネントを示します。 ソフトウェアコンポーネントすべてに関する完全な説明については、363 ページの「ソフトウェアのコンポーネント」 を参照してください。

コンポーネント	説明
.forward ファイル	ユーザーのホームディレクトリ内で設定して、メールを自動的にリダイレクトしたり、プログラムに送ったりすることができるファイル
メールボックス	メールサーバー上にあり、電子メールメッセージの最終受信先であるファイル
メールアドレス	メールメッセージが配信される受信者とシステムの名称を含むアドレス
メール別名	メールアドレス内で使用されている代替名
メールキュー	メールサーバーによる処理を必要とするメールメッセージの集まり
ポストマスター	メールサービスについての問題を報告し質問を出すために使用される特別なメール別名
sendmail 構成ファイル	メールのルーティングに必要なすべての情報の入ったファイル

ハードウェアコンポーネントの概要

メール構成では次の 3 つの要素が必要ですが、これらは同じシステムで組み合わせることも、別のシステムで提供することもできます。

- メールホスト – 解釈処理が困難なメールアドレスを扱うように構成されたシステム
- 少なくとも 1 台のメールサーバー – 1 つまたは複数のメールボックスを保持するように構成されたシステム
- メールクライアント – メールサーバーからメールにアクセスするシステム

ユーザーがドメイン外のネットワークと通信をするためには、4 番目の要素であるメールゲートウェイを追加する必要があります。

図 24–1 には、一般的な電子メール構成を示しますが、ここでは基本的な 3 つのメール要素とメールゲートウェイが使用されています。

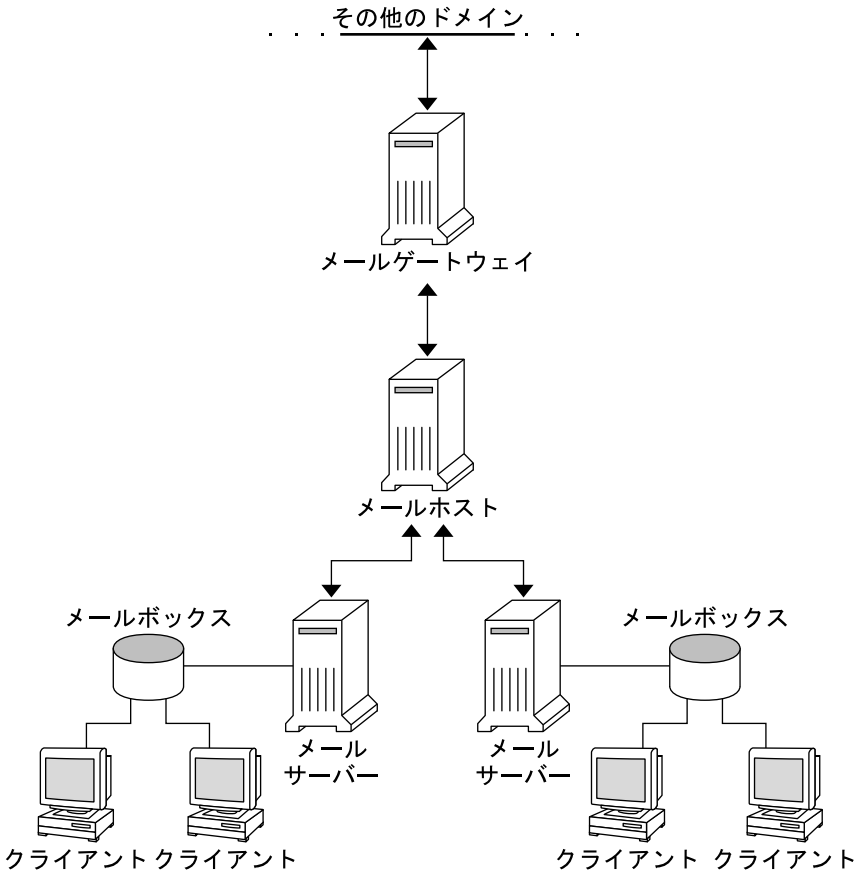


図 **24–1** 一般的な電子メール構成

各要素については、370 ページの「ハードウェアコンポーネント」 を参照してください。

第 25 章

メールサービス (手順)

この章ではメールサービスを設定し、管理する方法について説明します。メールサービスの管理に詳しくない方は、第 24 章 で、メールサービスのコンポーネントの概要と一般的なメールサービス構成 (図 24–1 参照) の説明をお読みください。この章では、次の関連作業について説明します。

メールサービスのコンポーネントについての詳細は、第 26 章 を参照してください。また、この章では、メールサービスのプログラムとファイル、メールルーティング処理、ネームサービスを使った `sendmail` の対話型操作についても説明します。

今回の Solaris 9 リリースに含まれている `sendmail` バージョン 8.12 の新機能については、第 27 章 を参照してください。`mail.local`、`mailstats`、および `makemap` の変更点についての説明もあります。 また、第 27 章 では、新しい保守コマンド `editmap` についても説明します。

メールサービス (作業マップ)

次の表に、この章で説明する特定の手順グループの作業マップへの参照先を示します。

作業	説明	参照先
メールサービスを設定する	メールサービスの各コンポーネントを設定する手順。メールサーバー、メールクライアント、メールホスト、メールゲートウェイ、仮想ホストなどの設定方法と、sendmail での DNS の使用方法を学ぶ	320 ページの「メールサービスの設定 (作業マップ)」
sendmail 構成ファイルを構築する	sendmail.cf ファイルを変更する手順。例としてドメインマスカレードを有効にする方法を取り上げる	329 ページの「sendmail.cf 構成ファイルの構築 (手順) 」
代替構成を使ってメール配信を管理する	マスターデーモンが無効な場合に発生する可能性があるメール配信上の問題を防ぐための手順	331 ページの「代替構成を使用したメール配信の管理 (手順)」
メール別名ファイルを管理する	ネットワークで別名を提供するための手順。NIS+ テーブルのエントリの管理方法を説明する。また、NIS マップ、ローカルメール別名、キー付きマップファイル、およびポストマスター別名の設定方法も説明する	332 ページの「メール別名ファイルの管理 (作業マップ)」
メールキューを管理する	スムーズなキュー処理を提供するための手順。メールキューの表示と移動方法、メールキューの強制処理の方法、メールキューのサブセットの実行方法、古いメールキューの実行方法を説明する	344 ページの「キューディレクトリの管理 (作業マップ)」
.forward ファイルを管理する	.forward を無効にしたり、.forward ファイルの検索パスを変更したりする手順。/etc/shells を作成し生成することにより、.forward ファイルの使用をユーザーに許可する方法も説明する	348 ページの「.forward ファイルの管理 (作業マップ)」

作業	説明	参照先
メールサービスの障害追跡手順とヒント	メールサービスで発生した問題を解決するための手順とヒント。メール構成のテスト、メール別名の確認、`sendmail` ルールセットのテスト、他のシステムへの接続の確認、メッセージの記録などの方法について学ぶ。他のメール診断情報の情報源を紹介する	350 ページの「メールサービスの障害対処とヒント (作業マップ)」
エラーメッセージを解釈処理する	メール関連のエラーメッセージを解釈処理するための情報	355 ページの「エラーメッセージの解釈」

メールシステムの計画

以下に、メールシステムを計画するときに考慮すべき点を挙げます。

- 必要に応じてメール構成のタイプを決定します。この節では、メール構成の基本の 2 タイプについて説明し、各構成を設定するために必要なことがらについて簡単に説明します。新しいメールシステムを設定する必要がある場合、あるいは既存のメールシステムを拡張する場合は、この節の内容が役立つでしょう。318 ページの「ローカルメール専用」 では、1 番目の構成タイプについて、318 ページの「ローカルメールとリモート接続」 では 2 番目の構成タイプについて説明します。
- 必要に応じてメールサーバー、メールホスト、およびメールゲートウェイとして動作するシステムを選択します。
- サービスを提供するすべてのメールクライアントのリストを作成し、メールボックスの場所も含めます。このリストは、ユーザーのメール別名を作成するときに役立ちます。
- 別名の更新方法とメールメッセージの転送方法を決めます。ユーザーがメールの転送要求やデフォルトのメール別名の変更要求を送る場所として、`aliases` メールボックスを設定できます。 システムで NIS または NIS+ を使用する場合、メール転送の管理は、ユーザー自身ではなく、管理者が行うことができます。332 ページの「メール別名ファイルの管理 (作業マップ)」 に、別名に関連する作業の一覧があります。348 ページの「`.forward` ファイルの管理 (作業マップ)」 に、`.forward` ファイルの管理に関連する作業の一覧があります。

メールシステムの計画を立てたら、サイトにシステムを設定し、320 ページの「メールサービスの設定 (作業マップ)」 で説明する機能を実行します。他の作業については、316 ページの「メールサービス (作業マップ)」 を参照してください。

ローカルメール専用

図 25–1 に示すように、もっとも単純なメール構成は、1 台のメールホストに 2 台以上のワークステーションが接続されている場合です。すべてのクライアントがローカルのディスクにメールを格納し、メールサーバーとして機能します。メールアドレスは `/etc/mail/aliases` ファイルを使って構文解析されます。

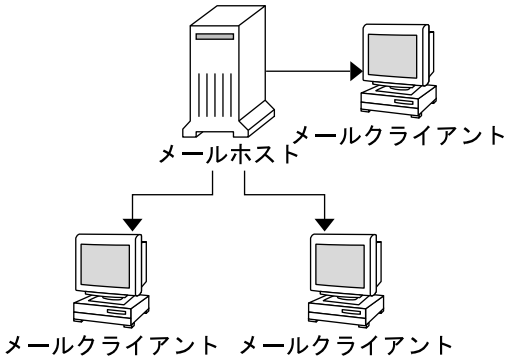


図 **25–1** ローカルメール構成

この種類のメール構成を設定するには、以下が必要です。

- 各メールクライアントシステムでのデフォルトの `/etc/mail/sendmail.cf` ファイル (編集は不要)
- メールホストとして指定されたサーバー。 メールホストを指定するには、メールホストの `/etc/hosts` ファイルに `mailhost.`*domain_name* を追加。また、NIS や NIS+ を実行していない場合は、すべてのメールクライアントの `/etc/hosts` ファイルにメールホスト IP アドレス行を追加
- ローカルメールボックスを持つ任意のシステム上にある同じ内容の `/etc/mail/aliases` ファイル (NIS や NIS+ を実行していない場合)
- 各メールクライアントシステムの `/var/mail` に、メールボックスを格納できるだけの十分な領域

メールサービスの設定の詳細については、320 ページの「メールサービスの設定 (作業)」 を参照してください。 メールサービスの設定に関する特定の手順については、320 ページの「メールサービスの設定 (作業マップ)」 を参照してください。

ローカルメールとリモート接続

小規模なネットワークにおけるもっとも一般的なメール構成を 図 25–2 に示します。1 つのシステムが、メールサーバー、メールホスト、および外部へのメールゲートウェイを兼ねます。メールは、メールゲートウェイ上の `/etc/mail/aliases` ファイルを使って配布されます。ネームサービスは必要ありません。

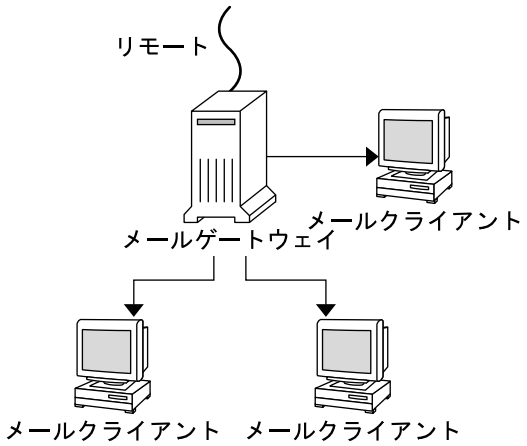


図 **25–2** UUCP 接続を使ったローカルメール構成

この構成では、メールクライアントがメールホスト上の `/var/mail` からメールファイルをマウントすると想定できます。この種類のメール構成を設定するには、以下が必要です。

- メールゲートウェイ上に `main.cf` ファイル (MX [メール交換] レコードを使用する場合、編集は不要)
- 各メールクライアントシステムに、デフォルトの `/etc/mail/sendmail.cf` ファイル (編集は不要)
- メールホストとして指定されたサーバー。 メールホストを指定するには、メールホストの `/etc/hosts` ファイルに `mailhost.`*domain_name* を追加。また、NIS や NIS+ を実行していない場合は、すべてのメールクライアントの `/etc/hosts` ファイルにメールホストの IP アドレス行を追加
- ローカルメールボックスを持つ任意のシステム上にある同じ内容の `/etc/mail/aliases` ファイル (NIS や NIS+ を実行していない場合)
- メールサーバーの `/var/mail` に、クライアントのメールボックスを格納できるだけの十分な領域

メールサービスの設定の詳細については、320 ページの「メールサービスの設定 (作業)」 を参照してください。 メールサービスの設定に関する特定の手順については、320 ページの「メールサービスの設定 (作業マップ)」 を参照してください。

メールサービスの設定 (作業マップ)

次の表では、メールサービスの設定の手順を説明します。

作業	説明	参照先
メールサーバーを設定する	サーバーがメールを経路指定できるようにする手順	321 ページの「メールサーバーを設定する方法」
メールクライアントを設定する	ユーザーがメールを受信できるようにする手順	323 ページの「メールクライアントを設定する方法」
メールホストを設定する	電子メールアドレスを解釈処理できるメールホストを確立する手順	325 ページの「メールホストを設定する方法」
メールゲートウェイを設定する	ドメイン外のネットワークとの通信を管理する手順	326 ページの「メールゲートウェイを設定する方法」
`sendmail` で DNS を使用する	DNS ホストルックアップ機能を有効にする手順	328 ページの「`sendmail` で DNS を使用する方法」
仮想ホストを設定する	ホストに複数の IP アドレスを割り当てる手順	329 ページの「仮想ホストを設定する方法」

メールサービスの設定 (作業)

サイトが企業外の電子メールサービスに接続していないか、あるいは企業が 1 つのドメイン内にある場合は、メールサービスを比較的容易に設定できます。

ローカルメール用に 2 つのタイプの構成が必要です。これらの構成については、318 ページの「ローカルメール専用」 の 図 25–1 を参照してください。ドメイン外のネットワークと通信するためには、さらに 2 つのタイプの構成が必要です。これらの構成については、313 ページの「ハードウェアコンポーネントの概要」 の 図 24–1 または 318 ページの「ローカルメールとリモート接続」 の 図 25–2 を参照してください。これらの構成は、同じシステムで組み合わせるか、あるいは別のシステムで提供できます。 たとえば、同じシステムにメールホストとメールサーバーの機能を持たせる場合は、この節の説明に従って、まずそのシステムをメールホストとして設定します。次に、この節の説明に従って、同じシステムをメールサーバーとして設定します。

注 – 次のメールサーバーとメールクライアントの設定の手順は、メールボックスが NFS でマウントされているときに適用されます。ただし、メールボックスは通常、ローカルにマウントされた /var/mail ディレクトリで維持されるので、次の手順は必要ありません。

▼ メールサーバーを設定する方法

メールサーバーはローカルユーザーにメールサービスを提供するだけなので、設定には特別な手順は必要ありません。ユーザーはパスワードファイルか名前空間にエントリが必要です。またユーザーはメール配信用のローカルのホームディレクトリが必要です (~/.forward ファイルを確認するため)。このため、ホームディレクトリサーバーがしばしばメールサーバーとして設定されます。メールサーバーについては、第 26 章 の 370 ページの「ハードウェアコンポーネント」 でさらに詳しく説明します。

メールサーバーは、メールクライアント宛てにメールを経路指定します。メールサーバーに必要な唯一のリソースは、クライアントメールボックスのための十分なスプール空間です。

注 – クライアントが自分のメールボックスにアクセスできるには、/var/mail ディレクトリがリモートマウントに使用可能であるか、あるいは POP (Post Office Protocol) または IMAP (Internet Message Access Protocol) のようなサービスがサーバーから使用可能でなければなりません。以下では、/var/mail ディレクトリを使ってメールサーバーを設定する方法を示します。このマニュアルでは、POP または IMAP の構成方法については説明しません。

次の作業のために、/var/mail ディレクトリがエクスポートされていることを /etc/dfs/dfstab ファイルで確認します。

1. メールサーバーのスーパーユーザーになるか、同等の役割になります。
   役割については、『*Solaris* のシステム管理 (セキュリティサービス)』の「特権付きアプリケーションの使用」を参照してください。
2. sendmail を停止します。

   ```
   # /etc/init.d/sendmail stop
   ```
3. /var/mail ディレクトリをリモートアクセスに使用できるかどうかを確認します。

   ```
   # share
   ```
   /var/mail ディレクトリが表示された場合は、手順 5 に進みます。
   /var/mail ディレクトリが表示されない場合、あるいはリストが表示されない場合は、該当する手順に進みます。

a. **(**省略可能**)** リストが表示されない場合は、**NFS** サービスを起動します。
151 ページの「ファイルシステム自動共有を設定する方法」 の手順に従って、/var/mail ディレクトリを使用して NFS サービスを起動します。

b. **(**省略可能**)** /var/mail ディレクトリがリストに含まれていない場合は、/var/mail ディレクトリを /etc/dfs/dfstab に追加します。
/etc/dfs/dfstab ファイルに次のコマンド行を追加します。

```
share -F nfs -o rw /var/mail
```

4. ファイルシステムをマウントできるようにします。

```
# shareall
```

5. ネームサービスが起動されていることを確認します。

a. **(**省略可能**) NIS** を実行している場合は、次のコマンドを使用します。

```
# ypwhich
```

詳細は、ypwhich(1) のマニュアルページを参照してください。

b. **(**省略可能**) NIS+** を実行している場合は、次のコマンドを使用します。

```
# nisls
```

詳細は、nisls(1) のマニュアルページを参照してください。

c. **(**省略可能**) DNS** を実行している場合は、次のコマンドを使用します。

```
# nslookup hostname
```

hostname	ホスト名を指定

詳細は、nslookup(1M) のマニュアルページを参照してください。

d. **(**省略可能**) LDAP** を実行している場合は、次のコマンドを使用します。

```
# ldaplist
```

詳細は、ldaplist(1) のマニュアルページを参照してください。

6. sendmail を再起動します。

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

注 – mail.local プログラムは、メッセージがはじめて配信されたときに /var/mail ディレクトリでメールボックスを自動的に作成します。メールクライアントの個々のメールボックスを作成する必要はありません。

▼ メールクライアントを設定する方法

メールクライアントは、メールボックスがメールサーバーにあり、/etc/mail/aliases ファイルのメール別名がメールボックスの位置を指しているメールサービスのユーザーです。第 26 章 の 370 ページの「ハードウェアコンポーネント」 に、メールクライアントについての簡単な説明があります。

注 – POP (Post Office Protocol) または IMAP (Internet Message Access Protocol) のようなサービスを使ってメールクライアントを設定することもできます。ただし、POP または IMAP の構成方法については、このマニュアルでは説明していません。

---

1. メールクライアントシステム上でスーパーユーザーになるか、同等の役割になります。

   役割については、『*Solaris* のシステム管理 (セキュリティサービス)』の「特権付きアプリケーションの使用」を参照してください。

2. sendmail を停止します。

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

3. メールクライアントのシステムで /var/mail マウントポイントがあることを確認します。

   マウントポイントは、インストール過程で作成されています。ls を使用すると、ファイルシステムが存在するかどうかを確認できます。次の例はファイルシステムが作成されていない場合に受け取る応答を示しています。

   ```
   # ls -l /var/mail
   /var/mail not found
   ```

4. /var/mail ディレクトリにファイルが何もないことを確認します。

   メールファイルがこのディレクトリにある場合は、それらのファイルを移動させ、サーバーから /var/mail ディレクトリがマウントされる時にその対象とならないようにします。

5. メールサーバーから /var/mail ディレクトリをマウントします。

   メールディレクトリは自動的にマウントすることも、ブート時にマウントすることもできます。

   a. **(省略可能)** /var/mail を自動的にマウントします。

      次のようなエントリを /etc/auto_direct ファイルに追加します。

      ```
      /var/mail -rw,hard,actimeo=0 server:/var/mail
      ```

      server　　割り当てられているサーバー名を指定

b. **(省略可能)** ブート時に /var/mail をマウントします。

/etc/vfstab ファイルに以下のエントリを追加します。このエントリにより、指定されたメールサーバー上の /var/mail ディレクトリがローカルの /var/mail ディレクトリをマウントできます。

```
server:/var/mail - /var/mail nfs - no rw,hard,actimeo=0
```

システムをリブートするたびに、クライアントのメールボックスが自動的にマウントされます。 システムをリブートしない場合は、次のコマンドを入力すれば、クライアントのメールボックスをマウントできます。

```
# mountall
```

注意 – メールボックスのロックとメールボックスへのアクセスが適切に動作するには、NFS サーバーからメールをマウントする時に actimeo=0 オプションを入れる必要があります。

6. /etc/hosts を更新します。

/etc/hosts ファイルを編集し、メールサーバーのエントリを追加します。ネームサービスを使用する場合、この手順は必要ありません。

```
# cat /etc/hosts
#
# Internet host table
#
..
IP_address       mailhost   mailhost   mailhost.example.com
```

IP_address	割り当てられている IP アドレスを指定
example.com	割り当てられているドメインを指定
mailhost	割り当てられているメールホストを指定

詳細は、hosts(4) のマニュアルページを参照してください。

7. 別名ファイルの **1** つにクライアントのエントリを追加します。

メール別名ファイルの管理に関する作業マップについては、332 ページの「メール別名ファイルの管理 (作業マップ)」 を参照してください。

注 – mail.local プログラムは、メッセージがはじめて配信されたときに /var/mail ディレクトリでメールボックスを自動的に作成します。メールクライアントの個々のメールボックスを作成する必要はありません。

8. sendmail を再起動します。

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

▼ メールホストを設定する方法

メールホストは、電子メールアドレスを解決し、ドメイン内でメールを再度ルーティングします。メールホストに適しているシステムは、ドメイン外または親ドメインにネットワークを接続するシステムです。 次に、メールホストを設定する手順を示します。

1. メールホストシステム上でスーパーユーザーになるか、同等の役割になります。
   役割については、『*Solaris* のシステム管理 *(*セキュリティサービス*)*』の「特権付きアプリケーションの使用」を参照してください。

2. sendmail を停止します。

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

3. ホスト名の構成を確認します。
   次のように check-hostname スクリプトを実行し、sendmail が、このサーバーの完全指定のホスト名を識別できるかどうかを確認します。

   ```
   % /usr/lib/mail/sh/check-hostname
   hostname phoenix OK: fully qualified as phoenix.example.com
   ```

   このスクリプトによる完全指定ホスト名の識別ができなかった場合は、完全指定ホスト名を /etc/hosts 内のホストの最初の別名として追加する必要があります。

4. /etc/hosts ファイルを更新します。
   以下から、適切な手順を選択します。

   a. **(**省略可能**) NIS** または **NIS+** を使用している場合は、新しいメールホストとして割り当てられたシステムの /etc/hosts ファイルを編集します。
      メールホストシステムの IP アドレスとシステム名の後に mailhost と mailhost.*domain* を追加します。

      IP_address mailhost mailhost mailhost.*domain* loghost

IP_address	割り当てられている IP アドレスを指定
mailhost	メールホストシステムのシステム名を指定
domain	拡張ドメイン名を指定

      これで、このシステムはメールホストとして指定されます。*domain* は、次のコマンドの出力にサブドメイン名として指定されている文字列と同じにする必要があります。

      ```
      % /usr/lib/sendmail -bt -d0 </dev/null
      Version 8.12.0+Sun
      ```

```
 Compiled with: LDAPMAP MAP_REGEX LOG MATCHGECOS MIME7TO8 MIME8TO7

 NAMED_BIND NDBM NETINET NETINET6 NETUNIX NEWDB NIS NISPLUS
                QUEUE SCANF SMTP USERDB XDEBUG

============ SYSTEM IDENTITY (after readcf) ============
      (short domain name) $w = phoenix
  (canonical domain name) $j = phoenix.example.com
         (subdomain name) $m = example.com
              (node name) $k = phoenix
========================================================
```

以上の変更を行なった後の hosts ファイルの例を以下に示します。

```
# cat /etc/hosts
#
# Internet host table
#
172.31.255.255   localhost
192.168.255.255  phoenix mailhost mailhost.example.com loghost
```

b. **(省略可能) NIS** または **NIS+** を使用しない場合は、ネットワーク内の各システムにある /etc/hosts ファイルを編集して、次のエントリを作成します。

IP_address mailhost mailhost *mailhost.domain* loghost

5. 正しい構成ファイルを選択し、コピーして名前を変更します。

次のコマンドは、/etc/mail/main.cf ファイルをコピーし名前を変更します。

```
# cp /etc/mail/main.cf /etc/mail/sendmail.cf
```

6. sendmail を再起動します。

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

7. メール構成をテストします。

手順については、351 ページの「メール構成をテストする方法」 を参照してください。

メールホストの詳細については、第 26 章 の 370 ページの「ハードウェアコンポーネント」 を参照してください。

▼ メールゲートウェイを設定する方法

メールゲートウェイは、ドメイン外のネットワークとの通信を管理します。送信側メールゲートウェイ上のメールプログラムは、受信側システムのメールプログラムと同じでなければなりません。

メールゲートウェイに適しているシステムは、Ethernet および電話回線に接続されているシステムか、あるいはインターネットへのルーターとして設定されているシステムです。メールホストをメールゲートウェイとして設定するか、あるいは別のシステ

ムをメールゲートウェイとして設定できます。複数のメールゲートウェイを自分のドメイン用として設定できます。UUCP (UNIX-to-UNIX Copy Program) 接続がある場合は、メールゲートウェイとして UUCP 接続を使ってシステムを構成します。

1. メールゲートウェイ上でスーパーユーザーになるか、同等の役割になります。
   役割については、『*Solaris* のシステム管理 *(*セキュリティサービス*)*』の「特権付きアプリケーションの使用」を参照してください。

2. sendmail を停止します。

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

3. 正しい構成ファイルを選択し、コピーして名前を変更します。
   次のコマンドは main.cf ファイルをコピーし名前を変更します。

   ```
   # cp /etc/mail/main.cf /etc/mail/sendmail.cf
   ```

4. ホスト名の構成を確認します。
   次のように check-hostname スクリプトを実行し、sendmail が、このサーバーの完全指定のホスト名を識別できるかどうかを確認します。

   ```
   # /usr/lib/mail/sh/check-hostname
   hostname phoenix OK: fully qualified as phoenix.example.com
   ```

   このスクリプトによる完全指定ホスト名の識別ができなかった場合は、完全指定ホスト名を /etc/hosts 内のホストの最初の別名として追加する必要があります。この手順の詳細については、325 ページの「メールホストを設定する方法」 の 手順 4を参照してください。

5. ネームサービスが起動されていることを確認します。

   a. **(**省略可能**) NIS** を実行している場合は、次のコマンドを使用します。

      ```
      # ypwhich
      ```

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

   b. **(**省略可能**) NIS+** を実行している場合は、次のコマンドを使用します。

      ```
      # nisls
      ```

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

   c. **(**省略可能**) DNS** を実行している場合は、次のコマンドを使用します。

      ```
      # nslookup hostname
      ```

hostname	ホスト名を指定

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

   d. **(**省略可能**) LDAP** を実行している場合は、次のコマンドを使用します。

```
# ldaplist
```

詳細は、ldaplist(1) のマニュアルページを参照してください。

6. sendmail を再起動します。

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

7. メール構成をテストします。

手順については、351 ページの「メール構成をテストする方法」 を参照してください。

メールゲートウェイの詳細については、第 26 章 の 370 ページの「ハードウェアコンポーネント」 を参照してください。

▼ sendmail で DNS を使用する方法

DNS ネームサービスは、個別の別名をサポートしません。このネームサービスは、MX (メール交換局) レコードおよび cname レコードを使用するホストまたはドメインの別名をサポートします。ホスト名とドメイン名は両方またはいずれか一方を DNS データベースで指定できます。sendmail と DNS の詳細については、第 26 章 の 393 ページの「sendmail とネームサービスの相互作用」、または『*Solaris* のシステム管理 *(*ネーミングとディレクトリサービス *: DNS*、*NIS*、*LDAP* 編*)*』を参照してください。

1. スーパーユーザー、またはそれと同等の役割になります。

役割については、『*Solaris* のシステム管理 *(*セキュリティサービス*)*』の「特権付きアプリケーションの使用」を参照してください。

2. **DNS** ホストルックアップ機能を有効にします **(NIS** のみ**)**。

/etc/nsswitch.conf ファイルを編集し、dns フラグを含む hosts の定義から # を削除します。DNS ホスト別名を使用するには、次の例に示すように、ホストエントリに dns フラグが含まれている必要があります。

```
# grep hosts /etc/nsswitch.conf
#hosts:     nisplus [NOTFOUND=return] files
hosts:      nisplus dns [NOTFOUND=return] files
```

3. mailhost と mailhost.*domain* エントリを確認します。

nslookup を使用して、mailhost と mailhost.*domain* のエントリが DNS データベースに存在することを確認します。 詳細は、nslookup(1M) のマニュアルページを参照してください。

▼ 仮想ホストを設定する方法

ホストに複数の IP アドレスを割り当てる必要がある場合は、次の Web サイトを参照してください。 http://www.sendmail.org/virtual-hosting.html。このサイトでは、`sendmail` を使って仮想ホストを設定する方法を詳しく説明しています。ただし、「Sendmail Configuration」の節では、次に示す手順 3b は実行しないでください。

```
# cd sendmail-VERSION/cf/cf
# ./Build mailserver.cf
# cp mailserver.cf /etc/mail/sendmail.cf
```

代わりに、Solaris オペレーティング環境では、次の手順を実行してください。

```
# cd /usr/lib/mail/cf
# /usr/ccs/bin/make mailserver.cf
# cp mailserver.cf /etc/mail/sendmail.cf
```

mailserver `.cf` ファイルの名前を指定

329 ページの「`sendmail.cf` 構成ファイルの構築 (手順) 」 では、構築手順の一部として、これと同じ 3 つの手順を説明しています。

`/etc/mail/sendmail.cf` ファイルを生成したら、仮想ユーザーテーブルを作成するなど、次の手順へ進むことができます。

`sendmail.cf` 構成ファイルの構築 (手順)

330 ページの「新しい `sendmail.cf` ファイルを構築する方法」 で、構成ファイルの構築方法について説明します。`sendmail.cf` ファイルの以前のバージョンも引き続き使用できますが、新しい形式を使用することをお勧めします。

詳細は、以下の情報を参照してください。

- `/usr/lib/mail/README`。構成手順の詳細な説明
- http://www.sendmail.org。`sendmail` 構成に関するオンライン情報
- 第 26 章 の 362 ページの「構成ファイルのバージョン」 と 384 ページの「`sendmail` 構成ファイル」。いくつかのガイダンスが記載されています

第 27 章の次の節では、新しい `m4` 構成機能について説明しています。

- 403 ページの「構成ファイルの新しい構成オプションと改訂された構成オプション、および関連トピック」
- 420 ページの「`sendmail` における新しい `m4` 構成マクロおよび改訂された `m4` 構成マクロ」

▼ 新しい sendmail.cf ファイルを構築する方法

次に、新しい構成ファイルを構築する手順を示します。

注 – /usr/lib/mail/cf/main-v7sun.mc は、 /usr/lib/mail/cf/main.mc になりました。

---

1. スーパーユーザー、またはそれと同等の役割になります。
   役割については、『*Solaris* のシステム管理 *(*セキュリティサービス*)*』の「特権付きアプリケーションの使用」を参照してください。
2. sendmail を停止します。

   ```
   # /etc/init.d/sendmail stop
   ```
3. 変更しようとする構成ファイルのコピーを作成します。

   ```
   # cd /usr/lib/mail/cf
   # cp main.mc myhost.mc
   ```

myhost	.mc ファイルの新しい名前を指定

4. 必要に応じて、新しい構成ファイル **(**たとえば、*myhost* .mc**)** を編集します。
   たとえば、ドメインマスカレードを有効にするには、次のコマンド行を追加します。

   ```
   # cat myhost.mc
   ..
   MASQUERADE_AS(`host.domain')
   ```

host.domain	目的のホスト名とドメイン名を指定

   この例では、MASQUERADE_AS は、送信されたメールに、$j ではなく *host.domain* から送信されたものとしてラベルを付けます。
5. m4 を使って構成ファイルを構築します。

   ```
   # /usr/ccs/bin/make myhost.cf
   ```
6. -C オプションを使用して、新しい構成ファイルをテストし、新しいファイルを指定します。

   ```
   # /usr/lib/sendmail -C myhost.cf -v testaddr </dev/null
   ```

   このコマンドは testaddr にメッセージを送信し、その一方で稼働時にメッセージを表示します。 システム上で sendmail サービスを再起動することなく送信メールだけをテストできます。まだメールを処理していないシステムでは、351 ページの

「メール構成をテストする方法」 で説明する完全なテスト手順を使用してください。

7. オリジナルのコピーを作成した後、新しい構成ファイルをインストールします。

```
# cp /etc/mail/sendmail.cf /etc/mail/sendmail.cf.save
# cp myhost.cf /etc/mail/sendmail.cf
```

8. sendmail サービスを再起動します。

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

代替構成を使用したメール配信の管理 (手順)

送受信されるメールの転送を容易にするため、sendmail の新しいデフォルトの構成は、デーモンとクライアントキューランナーを使用します。デーモンを無効にしている場合は、次の作業を行います。詳細は、400 ページの「新しい構成ファイル submit.cf」 を参照してください。

▼ sendmail.cf の代替構成を使ってメール配信を管理する方法

sendmail のデフォルトの構成では、クライアントキューランナーは、ローカルの SMTP ポートのデーモンにメールを送信できなければなりません。デーモンが SMTP ポート上で待機していない場合、メールはキューに留まります。この問題を避けるには、次の作業を行います。デーモンとクライアントキューランナーについての詳細、およびこの代替構成を使用する必要性を理解するには、400 ページの「新しい構成ファイル submit.cf」 を参照してください。

この手順を実行すると、デーモンは、ローカルホストからの接続を受け付けるためだけに動作するようになります。

1. スーパーユーザー、またはそれと同等の役割になります。
   役割については、『*Solaris* のシステム管理 *(*セキュリティサービス*)*』の「特権付きアプリケーションの使用」を参照してください。

2. sendmail を停止します。

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

3. 変更しようとする構成ファイル **(**必要に応じて subsidiary.mc と main.mc のどちらか**)** のコピーを作成しますこの例では、subsidiary.mc ファイルを使用します。

```
# cd /usr/lib/mail/cf
# cp subsidiary.mc myhost.mc
```

myhost .mc ファイルの新しい名前を指定

4. 新しい構成ファイル **(**たとえば、*myhost*.mc**)** を編集します。
   MAILER() 行の前に次の行を追加します。

```
# cat myhost.mc
..
DAEMON_OPTIONS('NAME=NoMTA4, Family=inet, Addr=127.0.0.1')dnl
DAEMON_OPTIONS('NAME=NoMTA6, Family=inet6, Addr=::1')dnl
```

5. m4 を使って構成ファイルを構築します。

```
# /usr/ccs/bin/make myhost.cf
```

6. オリジナルのコピーを作成した後、新しい構成ファイルをインストールします。

```
# cp /etc/mail/sendmail.cf /etc/mail/sendmail.cf.save
# cp myhost.cf /etc/mail/sendmail.cf
```

7. sendmail サービスを再起動します。

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

メール別名ファイルの管理 (作業マップ)

次の表では、メール別名ファイルの管理の手順を説明します。 このトピックの詳細は、第 26 章 の386 ページの「メール別名ファイル」 を参照してください。

作業	説明	参照先
NIS+ mail_aliases テーブルで別名のエントリを管理する	ネームサービスが NIS+ である場合に、mail_aliases テーブルの内容を管理する手順。エントリの表示、追加、編集、削除の方法を説明する	333 ページの「NIS+ mail_aliases テーブルの別名エントリを管理する方法」
NIS mail.aliases マップを設定する	ネームサービスが NIS の場合、mail.aliases マップを使って別名を設定する手順	338 ページの「NIS mail.aliases マップを設定する方法」

作業	説明	参照先
ローカルのメール別名ファイルを設定する	NIS や NIS+ などのネームサービスを使用していない場合に、/etc/mail/aliases ファイルを使って別名を設定する手順	340 ページの「ローカルメール別名ファイルを設定する方法」
キー付きマップファイルを作成する	キー付きマップファイルを使って別名を設定する手順	341 ページの「キー付きマップファイルの作成方法」
postmaster 別名を設定する	postmaster 別名を管理する手順。この別名は必須	342 ページの「postmaster 別名の管理」

メール別名ファイルの管理

メール別名はドメイン独自にする必要があります。 この節では、メール別名ファイルを管理する手順を説明します。また、Solaris 管理コンソールの「メーリングリスト」機能を使って別名データベース上でこれらの作業を実行することもできます。

その他に、makemap を使ってローカルメールホストにデータベースファイルを作成することもできます。makemap(1M) のマニュアルページを参照してください。ローカルのデータベースファイルを使用しても、NIS や NIS+ のようなネームサービスを使用するほどの利点は得られません。しかし、ネットワークのルックアップは必要ないため、ローカルのデータベースファイルからの方がより早くデータを取り出すことができます。詳細は、第 26 章 の393 ページの「sendmail とネームサービスの相互作用」 および386 ページの「メール別名ファイル」 を参照してください。

▼ NIS+ mail_aliases テーブルの別名エントリを管理する方法

NIS+ テーブルでエントリを管理するために、aliasadm コマンドを使用できます。aliasadm コマンドを使ってテーブルエントリを表示、追加、変更、または削除するには、次の手順に従います。

1. テーブルを所有する **NIS+** グループのメンバーになるか、メールサーバーのスーパーユーザーになるか、同等の役割になります。

   役割については、『*Solaris* のシステム管理 *(*セキュリティサービス*)*』の「特権付きアプリケーションの使用」を参照してください。

2. 必要に応じて、次に挙げる例のいずれかの手順に従って作業します。

   - 334 ページの「NIS+ mail_aliases テーブルを作成する例」
   - 334 ページの「NIS+ mail_aliases テーブルの全内容を表示する例」

- 335 ページの「NIS+ mail_aliases テーブルの個々のエントリを表示する例」
- 335 ページの「NIS+ mail_aliases テーブル内の部分一致エントリを表示する例」
- 335 ページの「例 — コマンド行から NIS+ mail_aliases テーブルへ別名を追加する」
- 336 ページの「例 — NIS+ mail_aliases テーブルを編集してエントリを追加する」
- 337 ページの「例 — NIS+ mail_aliases テーブル内のエントリを編集する」
- 338 ページの「例 — NIS+ mail_aliases テーブルからエントリを削除する」

場合によっては、最初にメールクライアント、クライアントのメールボックスの位置、およびメールサーバーシステム名のそれぞれのリストをコンパイルする必要があります。

▼ NIS+ mail_aliases テーブルを作成する例

テーブルを作成するには、次の手順に従います。

1. テーブルを所有する **NIS+** グループのメンバーになるか、メールサーバーのスーパーユーザーになるか、同等の役割になります。

   役割については、『*Solaris* のシステム管理 *(*セキュリティサービス*)*』の「特権付きアプリケーションの使用」を参照してください。

2. **NIS+** テーブルを作成します。

   ```
   # aliasadm -I
   ```

3. テーブルにエントリを追加します。

   - 2 つまたは 3 つの別名を追加する方法については、335 ページの「例 — コマンド行から NIS+ mail_aliases テーブルへ別名を追加する」を参照してください。
   - 多数の別名を追加する方法については、336 ページの「例 — NIS+ mail_aliases テーブルを編集してエントリを追加する」を参照してください。

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

▼ NIS+ mail_aliases テーブルの全内容を表示する例

テーブルの全内容を表示するには、次の手順に従います。

1. テーブルを所有する **NIS+** グループのメンバーになるか、メールサーバーのスーパーユーザーになるか、同等の役割になります。

   役割については、『*Solaris* のシステム管理 *(*セキュリティサービス*)*』の「特権付きアプリケーションの使用」を参照してください。

2. 別名のアルファベット順に全エントリを表示します。

   ```
   # aliasadm -l
   ```

詳細は、aliasadm(1M) のマニュアルページを参照してください。

▼ NIS+ mail_aliases テーブルの個々のエントリを表示する例

テーブルの個々のエントリを表示するには、次の手順に従います。

1. テーブルを所有する **NIS+** グループのメンバーになるか、メールサーバーのスーパーユーザーになるか、同等の役割になります。

   役割については、『*Solaris* のシステム管理 *(*セキュリティサービス*)*』の「特権付きアプリケーションの使用」を参照してください。

2. 個々のエントリを表示します。

   ```
   # aliasadm -m ignatz
   ignatz: ignatz@saturn # Alias for Iggy Ignatz
   ```

   このコマンドは、完全に一致する別名のみ表示し、部分的に一致するエントリは表示しません。aliasadm -m オプションでは、メタキャラクタ (*、? など) は使用できません。

詳細は、aliasadm(1M) のマニュアルページを参照してください。

▼ NIS+ mail_aliases テーブル内の部分一致エントリを表示する例

テーブルの部分一致エントリを表示するには、次の手順に従います。

1. テーブルを所有する **NIS+** グループのメンバーになるか、メールサーバーのスーパーユーザーになるか、同等の役割になります。

   役割については、『*Solaris* のシステム管理 *(*セキュリティサービス*)*』の「特権付きアプリケーションの使用」を参照してください。

2. テーブル内の部分一致エントリを表示します。

   ```
   # aliasadm -l | grep partial_string
   ```

partial_string	検索に使用する文字列を指定

詳細は、aliasadm(1M) のマニュアルページを参照してください。

▼ 例 — コマンド行から NIS+ mail_aliases テーブルへ別名を追加する

2 つまたは 3 つの別名をテーブルに追加するには、次の手順に従います。

1. メールクライアント、メールボックスの場所、およびメールサーバーシステムの名前の各リストをコンパイルします。
2. テーブルを所有する **NIS+** グループのメンバーになるか、メールサーバーのスーパーユーザーになるか、同等の役割になります。

   役割については、『*Solaris* のシステム管理 *(*セキュリティサービス*)*』の「特権付きアプリケーションの使用」を参照してください。
3. **(**省略可能**)** 必要な場合は、**NIS+** テーブルを作成します。

   まったく新しい NIS+ `mail_aliases` テーブルを作成する場合は、最初に NIS+ テーブルを初期設定しなければなりません。 テーブルの作成方法については、334 ページの「NIS+ `mail_aliases` テーブルを作成する例」 を参照してください。
4. テーブルに別名を追加します。

   次に、一般的なエントリの例を示します。

   ```
   # aliasadm -a iggy iggy.ignatz@saturn "Iggy Ignatz"
   ```

   上記の例の入力内容を次に説明します。

`-a`	別名を追加するためのオプション
`iggy`	簡略別名
`iggy.ignatz@saturn`	拡張別名
`"Iggy Ignatz"`	引用符で囲んだ別名

5. 作成したエントリを表示して間違いがないことを確認します。

   ```
   # aliasadm -m alias
   ```

alias	作成したエントリ

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

▼ 例 — NIS+ `mail_aliases` テーブルを編集してエントリを追加する

多数の別名をテーブルに追加するには、次の手順に従います。

1. メールクライアント、メールボックスの場所、およびメールサーバーシステムの名前の各リストをコンパイルします。
2. テーブルを所有する **NIS+** グループのメンバーになるか、メールサーバーのスーパーユーザーになるか、同等の役割になります。

役割については、『*Solaris* のシステム管理 *(*セキュリティサービス*)*』の「特権付きアプリケーションの使用」を参照してください。

3. 別名テーブルを表示して編集します。

```
# aliasadm -e
```

このコマンドは、テーブルを表示し、テーブルの編集を可能にします。使用するエディタは、`$EDITOR` 環境変数で設定されています。この変数が設定されていない場合、`vi` がデフォルトのエディタになります。

4. 次の形式で、**1** 行に **1** 別名ずつ入力します。

alias`:` *expanded_alias* `# ["`*option*`" # "`*comments*`"]`

alias	この列には、簡略別名を入力
expanded_alias	この列には、拡張別名を入力
option	この列は、将来の拡張のために予約されている
comments	この列は、別名など、個々の別名に関するコメントに使用

オプション列をブランクにする場合は、空の引用符 2 つ（`""`）を入力し、その後にコメントを追加します。

NIS+ `mail_aliases` テーブルでは、エントリの順序は重要ではありません。`aliasadm -l` コマンドがリストをソートし、エントリをアルファベット順に表示します。

詳細は、386 ページの「メール別名ファイル」 および`aliasadm(1M)` のマニュアルページを参照してください。

▼ 例 — NIS+ `mail_aliases` テーブル内のエントリを編集する

テーブル内のエントリを編集するには、次の手順に従います。

1. テーブルを所有する **NIS+** グループのメンバーになるか、メールサーバーのスーパーユーザーになるか、同等の役割になります。

役割については、『*Solaris* のシステム管理 *(*セキュリティサービス*)*』の「特権付きアプリケーションの使用」を参照してください。

2. 別名エントリを表示します。

```
# aliasadm -m alias
```

alias	割り当てられている別名を指定

3. 必要に応じて別名エントリを編集します。

```
# aliasadm -c alias expanded_alias [options comments]
```

alias	必要な場合は、別名を編集
expanded_alias	必要な場合は、拡張別名を編集
options	必要な場合は、オプションを編集
comments	必要な場合は、このエントリのコメントを編集

詳細は、`aliasadm(1M)` のマニュアルページおよび 386 ページの「メール別名ファイル」を参照してください。

4. 編集したエントリを表示し、エントリに間違いがないことを確認します。

```
# aliasadm -m alias
```

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

▼ 例 — NIS+ `mail_aliases` テーブルからエントリを削除する

テーブルからエントリを削除するには、次の手順に従います。

1. テーブルを所有する **NIS+** グループのメンバーになるか、メールサーバーのスーパーユーザーになるか、同等の役割になります。
   役割については、『*Solaris* のシステム管理 *(*セキュリティサービス*)*』の「特権付きアプリケーションの使用」を参照してください。

2. テーブルからエントリを削除します。

```
# aliasadm -d alias
```

alias	削除するエントリの別名を指定

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

▼ NIS `mail.aliases` マップを設定する方法

次の手順によって、NIS の `mail.aliases` マップを使って別名の設定を容易に行うことができます。

1. メールクライアント、メールボックスの場所、およびメールサーバーシステムの名前の各リストをコンパイルします。

2. **NIS** マスターサーバーのスーパーユーザーになるか、同等の役割になります。
   役割については、『*Solaris* のシステム管理 *(*セキュリティサービス*)*』の「特権付きアプリケーションの使用」を参照してください。

3. /etc/mail/aliases ファイルを編集し、次のようなエントリを作成します。

   a. メールクライアントごとにエントリを追加します。

      ```
      # cat /etc/mail/aliases
      ..
      alias:expanded_alias
      ```

alias	簡略別名を指定
expanded_alias	拡張別名 (user@host.domain.com) を指定

   b. Postmaster: root エントリがあることを確認します。

      ```
      # cat /etc/mail/aliases
      ..
      Postmaster: root
      ```

   c. root の別名を追加します。 ポストマスターとして指定された個人のメールアドレスを使用します。

      ```
      # cat /etc/mail/aliases
      ..
      root: user@host.domain.com
      ```

user@host.domain.com	指定されたポストマスターに割り当てられているアドレスを指定

4. **NIS** マスターサーバーがネームサービスを実行中で、各メールサーバーのホスト名を解釈処理できることを確認します。

5. /var/yp ディレクトリに移動します。

   ```
   # cd /var/yp
   ```

6. make コマンドを適用します。

   ```
   # make
   ```

   /etc/hosts および /etc/mail/aliases ファイルの変更は、NIS スレーブシステムに伝達され、遅くとも数分後には有効になります。

▼ ローカルメール別名ファイルを設定する方法

ローカルメール別名ファイルで別名を解釈処理するには、次の手順に従います。

1. ユーザーとメールボックスの場所の各リストをコンパイルします。
2. メールサーバーのスーパーユーザーになるか、あるいは同等の役割になります。
   役割については、『*Solaris* のシステム管理 *(*セキュリティサービス*)*』の「特権付きアプリケーションの使用」を参照してください。
3. `/etc/mail/aliases` ファイルを編集し、次のようなエントリを作成します。
   a. ユーザーごとにエントリを追加します。

      user1`:` *user2@host.domain*

user1	新しい別名を指定
user2@host.domain	新しい別名の実際のアドレスを指定

   b. `Postmaster: root` エントリがあることを確認します。

      ```
      # cat /etc/mail/aliases
      ..
      Postmaster: root
      ```

   c. `root` の別名を追加します。 ポストマスターとして指定された個人のメールアドレスを使用します。

      ```
      # cat /etc/mail/aliases
      ..
      root: user@host.domain.com
      ```

user@host.domain.com	指定されたポストマスターに割り当てられているアドレスを指定

4. 別名データベースを再構築します。

   ```
   # newaliases
   ```

   `/etc/mail/sendmail.cf` の `AliasFile` オプションの構成に応じて、このコマンドは、`/etc/mail/aliases.db` ファイルを 1 つ、または `/etc/mail/aliases.dir` と `/etc/mail/aliases.pag` の 1 組のファイルのどちらかをバイナリ形式で生成します。
5. 次の手順のどちらかを実行して、生成されたファイルをコピーします。
   a. **(**省略可能**)** `/etc/mail/aliases`、`/etc/mail/aliases.dir`、および `/etc/mail/aliases.pag` ファイルを他の各システムにコピーします。

rcp または rdist コマンドを使用して 3 つのファイルをコピーできます。 詳細は、rcp(1) のマニュアルページまたは rdist(1) のマニュアルページを参照してください。 また、この目的のためのスクリプトを作成することもできます。

これらのファイルをコピーしたら、newaliases コマンドを他の各システムで実行する必要はありません。ただし、メールクライアントを追加または削除するたびにすべての /etc/mail/aliases ファイルを更新する必要があるので注意してください。

b. **(省略可能)** /etc/mail/aliases.db ファイルを他の各システムにコピーします。

rcp または rdist コマンドを使用してファイルをコピーできます。 詳細は、rcp(1) のマニュアルページまたは rdist(1) のマニュアルページを参照してください。 また、この目的のためのスクリプトを作成することもできます。

このファイルをコピーしたら、newaliases コマンドを他の各システムで実行する必要はありません。ただし、メールクライアントを追加または削除するたびにすべての /etc/mail/aliases ファイルを更新する必要があるので注意してください。

▼ キー付きマップファイルの作成方法

キー付きマップファイルを作成するには、次の手順に従います。

1. メールサーバーのスーパーユーザーになるか、同等の役割になります。
   役割については、『*Solaris* のシステム管理 *(*セキュリティサービス*)*』の「特権付きアプリケーションの使用」を参照してください。
2. 入力ファイルを作成します。
   エントリには、次の構文を使用できます。

```
old_name@newdomain.com      new_name@newdomain.com
old_name@olddomain.com      error:nouser No such user here
@olddomain.com              %1@newdomain.com
```

old_name@newdomain.com	新たに割り当てたドメインでこれまで割り当てられていたユーザー名を指定
new_name@newdomain.com	新たに割り当てるアドレスを指定
old_name@olddomain.com	これまで割り当てられていたドメインでこれまで割り当てられていたユーザー名を指定
olddomain.com	これまで割り当てられていたドメインを指定
newdomain.com	新たに割り当てるドメインを指定

1 番目のエントリにより、メールは新しい別名に転送されます。2 番目のエントリにより、不適切な別名が使用されたときにメッセージが作成されます。最後のエントリ

により、すべての着信メールは olddomain から newdomain へ転送されます。

3. データベースファイルを作成します。

```
# /usr/sbin/makemap maptype newmap < newmap
```

maptype	dbm、btree、hash などのデータベースタイプを選択
newmap	入力ファイル名とデータベースファイル名の最初の部分を指定。dbm データベースタイプを選択すると、データベースファイルは接尾辞に .pag または .dir を使って作成される。他の 2 つのデータベースタイプの場合、ファイル名には .db が付く

postmaster 別名の管理

各システムは postmaster メールボックスにメールを送信できなければなりません。postmaster の NIS または NIS+ 別名を作成できます。あるいは、ローカルの /etc/mail/aliases ファイルそれぞれに別名を作成することもできます。次の手順を参照してください。

- 342 ページの「ローカルの各 /etc/mail/aliases ファイルに postmaster 別名を作成する方法」
- 343 ページの「postmaster 用に別のメールボックスを作成する方法」
- 343 ページの「postmaster メールボックスを /etc/mail/aliases ファイルの別名に追加する方法」

▼ ローカルの各 /etc/mail/aliases ファイルに postmaster 別名を作成する方法

postmaster 別名をローカルの各 /etc/mail/aliases ファイルに作成する場合は、次の手順に従います。

1. 各ローカルシステムのスーパーユーザーになるか、同等の役割になります。
   役割については、『*Solaris* のシステム管理 (セキュリティサービス)』の「特権付きアプリケーションの使用」を参照してください。

2. /etc/mail/aliases エントリを表示します。

```
# cat /etc/mail/aliases
# Following alias is required by the mail protocol, RFC 2821
# Set it to the address of a HUMAN who deals with this system's
# mail problems.
Postmaster: root
```

3. 各システムの /etc/mail/aliases ファイルを編集します。
   root をポストマスターに指定する個人のメールアドレスに変更します。

```
Postmaster: mail_address
```

mail_address　　ポストマスターとして指定された個人に割り当てられたアドレスを使用します。

4. **(省略可能)** ポストマスター用に別のメールボックスを作成します。
   ポストマスターがポストマスターメールと個人メールとを区別するために、別のメールボックスを作成できます。別のメールボックスを作成する場合は、`/etc/mail/aliases` ファイルを編集するときに、ポストマスターの個人メールアドレスではなくメールボックスアドレスを使用してください。詳細は、343 ページの「postmaster 用に別のメールボックスを作成する方法」を参照してください。

▼ postmaster 用に別のメールボックスを作成する方法

`postmaster` 用に別のメールボックスを作成する場合は、次の手順に従います。

1. メールサーバーのスーパーユーザーになるか、あるいは同等の役割になります。
   役割については、『*Solaris* のシステム管理 *(*セキュリティサービス*)*』の「特権付きアプリケーションの使用」を参照してください。

2. `postmaster` として指定された個人のアカウントを作成し、アスタリスク **(*)** をパスワードフィールドに入れます。
   ユーザーアカウントの追加の詳細については、『*Solaris* のシステム管理 *(*基本編*)*』の「ユーザーアカウントとグループの管理 (手順)」を参照してください。

3. メールが配信されたら、`mail` プログラムがメールボックス名に読み書きできるようにします。

   ```
   # mail -f postmaster
   ```

   postmaster　　割り当てられているアドレスを指定

▼ postmaster メールボックスを /etc/mail/aliases ファイルの別名に追加する方法

`postmaster` メールボックスを `/etc/mail/aliases` ファイル内の別名に追加する場合は、次の手順に従います。

1. 各システムのスーパーユーザーになるか、同等の役割になります。
   役割については、『*Solaris* のシステム管理 *(*セキュリティサービス*)*』の「特権付きアプリケーションの使用」を参照してください。

2. `root` の別名を追加します。 ポストマスターとして指定された個人のメールアドレスを使用します。

   ```
   # cat /etc/mail/aliases
   ..
   root: user@host.domain.com
   ```

user@host.domain.com	ポストマスターとして指定された個人に割り当てられたアドレスを使用します。

3. ポストマスターのローカルシステムで、別名の名前 (たとえば、sysadmin) を定義する /etc/mail/aliases ファイルにエントリを作成し、ローカルメールボックスのパスを入れます。

```
# cat /etc/mail/aliases
..
sysadmin: /usr/somewhere/somefile
```

sysadmin	新しい別名の名前を作成
/usr/somewhere/somefile	ローカルメールボックスのパスを指定

4. 別名データベースを再構築します。

```
# newaliases
```

キューディレクトリの管理 (作業マップ)

次の表では、メールキューの管理の手順を説明します。

作業	説明	参照先
メールキュー /var/spool/mqueue の内容を表示する	キューにあるメッセージの数とそれらのメッセージがキューから消去されるのにかかる時間を表示する手順	345 ページの「メールキュー /var/spool/mqueue の内容を表示する方法」
メールキュー /var/spool/mqueue を強制処理する	以前にメッセージを受信できなかったシステムへのメッセージを処理する手順	345 ページの「メールキュー /var/spool/mqueue でメールキューを強制処理する方法」
メールキュー /var/spool/mqueue のサブセットを実行する	ホスト名などアドレスの部分文字列を強制的に処理したり、キューから特定のメッセージを排除したりする手順	346 ページの「メールキュー /var/spool/mqueue のサブセットを実行する方法」
メールキュー /var/spool/mqueue を移動する	メールキューを移動する手順	346 ページの「メールキュー /var/spool/mqueue を移動する方法」

作業	説明	参照先
古いメールキュー /var/spool/omqueue を実行する	古いメールキューを実行する手順	347 ページの「古いメールキュー /var/spool/omqueue を実行する方法」

キューディレクトリの管理 (手順)

この節では、キューの管理に役立つ作業について説明します。クライアント専用のキューの詳細については、400 ページの「新しい構成ファイル submit.cf」 を参照してください。他の関連情報については、425 ページの「キューの新しい機能」 を参照してください。

▼ メールキュー /var/spool/mqueue の内容を表示する方法

キューにあるメッセージの数とそれらのメッセージがキューから消去されるのにかかる時間を表示するには、次の手順に従います。

次の情報を表示するには、以下に示すコマンドを使用します。

- キュー ID
- メッセージのサイズ
- メッセージがキューに入った日付
- メッセージの状態
- 送信者と受信者

```
# /usr/bin/mailq | more
```

このコマンドは、認証属性 solaris.admin.mail.mailq を確認します。確認が取れると、sendmail で -bp フラグを指定するのと同じ処理が実行されます。確認ができない場合は、エラーメッセージが表示されます。デフォルトでは、この認証属性はすべてのユーザーで使用できるようになっています。認証属性は、prof_attr 内のユーザーエントリを変更することにより無効にできます。詳細は、prof_attr(4) および mailq(1) のマニュアルページを参照してください。

▼ メールキュー /var/spool/mqueue でメールキューを強制処理する方法

たとえば、以前にメッセージを受信できなかったシステムへのメッセージを処理するには、次の手順に従います。

1. スーパーユーザーになるか、それと同等の役割になります。
   役割については、『*Solaris* のシステム管理 *(*セキュリティサービス*)*』の「特権付きアプリケーションの使用」を参照してください。
2. キューを強制処理し、キューが消去されるとジョブの進捗状況を表示します。

   # **/usr/lib/sendmail -q -v**

▼ メールキュー /var/spool/mqueue のサブセットを実行する方法

たとえば、ホスト名などアドレスの部分文字列を強制的に処理したり、キューから特定のメッセージを強制処理したりするには、次の手順に従います。

1. スーパーユーザーになるか、それと同等の役割になります。
   役割については、『*Solaris* のシステム管理 *(*セキュリティサービス*)*』の「特権付きアプリケーションの使用」を参照してください。
2. -qR*string* を使用して、いつでもメールキューのサブセットを実行できます。

   # **/usr/lib/sendmail -qR***string*

string	受信者の別名または *user@host.domain* の部分文字列 (ホスト名など) を指定

   代わりに、-qI*nnnnn* を使ってメールキューのサブセットを実行することもできます。

   # **/usr/lib/sendmail -qI***nnnnn*

nnnnn	キュー ID を指定

▼ メールキュー /var/spool/mqueue を移動する方法

メールキューを移動する場合は、次の手順に従います。

1. メールホストのスーパーユーザーになるか、同等の役割になります。
   役割については、『*Solaris* のシステム管理 *(*セキュリティサービス*)*』の「特権付きアプリケーションの使用」を参照してください。
2. sendmail デーモンを終了します。

   # **/etc/init.d/sendmail stop**

これで、sendmail はキューディレクトリを処理しなくなります。

3. /var/spool ディレクトリに移動します。

```
# cd /var/spool
```

4. mqueue ディレクトリとディレクトリ内のすべての内容を omqueue ディレクトリに移動します。次に、mqueue という名前の新しい空のディレクトリを作成します。

```
# mv mqueue omqueue; mkdir mqueue
```

5. ディレクトリのアクセス権を所有者は読み取り/書き込み/実行に、またグループは読み取り/実行に設定します。また、所有者とグループを daemon に設定します。

```
# chmod 750 mqueue; chown root:bin mqueue
```

6. sendmail を起動します。

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

▼ 古いメールキュー /var/spool/omqueue を実行する方法

古いメールキューを実行するには、次の手順に従います。

1. スーパーユーザーになるか、それと同等の役割になります。

   役割については、『*Solaris* のシステム管理 *(*セキュリティサービス*)*』の「特権付きアプリケーションの使用」を参照してください。

2. 古いメールキューを実行します。

```
# /usr/lib/sendmail -oQ/var/spool/omqueue -q
```

   -oQ フラグは代替キュー (待ち行列) ディレクトリを指定し、-q フラグはキューでの各ジョブを処理するように指示します。画面に冗長出力を表示している場合は、-v フラグを使用します。

3. 空のディレクトリを削除します。

```
# rmdir /var/spool/omqueue
```

.forward ファイルの管理 (作業マップ)

次の表では、.forward ファイルを管理するための手順を説明します。詳細は、第 26 章 の 389 ページの「.forward ファイル」 を参照してください。

作業	説明	参照先
.forward ファイルを無効にする	たとえば、自動転送を禁止する場合に実行する手順	348 ページの「.forward ファイルを無効にする方法」
.forward ファイルの検索パスを変更する	たとえば、すべての .forward ファイルを共通ディレクトリに移動させる場合に実行する手順	349 ページの「.forward ファイルの検索パスを変更する方法」
/etc/shells を作成し生成する	メールをプログラムまたはファイルに転送するために、ユーザーが .forward ファイルを使用できるようにする手順	350 ページの「/etc/shells の作成および生成方法」

.forward ファイルの管理 (手順)

この節では、.forward ファイルの管理に関する複数の手順を説明します。これらファイルはユーザーが編集できるので、ファイルが問題の原因になる場合があります。詳細は、第 26 章 の 389 ページの「.forward ファイル」 を参照してください。

▼ .forward ファイルを無効にする方法

この手順は、自動転送を禁止し、特定のホストの .forward ファイルを無効にします。

1. スーパーユーザーになるか、それと同等の役割になります。
   役割については、『*Solaris* のシステム管理 (セキュリティサービス)』の「特権付きアプリケーションの使用」を参照してください。
2. /usr/lib/mail/domain/solaris-generic.m4 またはサイト固有のドメイン m4 ファイルのコピーを作成します。

```
# cd /usr/lib/mail/domain
# cp solaris-generic.m4 mydomain.m4
```

mydomain 選択するファイル名を指定

3. 次の行を、作成したファイルに追加します。

   define(‘confFORWARD_PATH’,‘’)dnl

   m4 ファイルに confFORWARD_PATH の値がすでに存在する場合は、NULL 値で置き換えられます。

4. 新しい構成ファイルを構築してインストールします。

   この手順の詳細については、330 ページの「新しい sendmail.cf ファイルを構築する方法」を参照してください。

▼ .forward ファイルの検索パスを変更する方法

たとえば、すべての .forward ファイルを共通ディレクトリに入れる場合は、次の手順に従います。

1. スーパーユーザーになるか、それと同等の役割になります。

   役割については、『*Solaris* のシステム管理 *(*セキュリティサービス*)*』の「特権付きアプリケーションの使用」を参照してください。

2. /usr/lib/mail/domain/solaris-generic.m4 またはサイト固有のドメイン m4 ファイルのコピーを作成します。

   ```
   # cd /usr/lib/mail/domain
   # cp solaris-generic.m4 mydomain.m4
   ```

   mydomain 選択するファイル名を指定

3. 次の行を作成したファイルに追加します。

   define(‘confFORWARD_PATH’,‘$z/.forward:/var/forward/$u’)dnl

   m4 ファイルに confFORWARD_PATH の値がすでに存在する場合は、新しい値で置き換えられます。

4. 新しい構成ファイルを構築してインストールします。

   この手順の詳細については、330 ページの「新しい sendmail.cf ファイルを構築する方法」を参照してください。

▼ /etc/shells の作成および生成方法

このファイルは標準のリリースには含まれていないので、プログラムまたはファイルにメールを転送するためにユーザーが .forward ファイルを使用できるようにする場合には、追加する必要があります。grep を使用して、パスワードファイルにリストされたすべてのシェルを特定し、ファイルを手動で作成することができます。これにより、シェルをファイルに入力できます。しかし、ダウンロード可能なスクリプトを使用する次の手順の方がより簡単です。

1. スクリプトをダウンロードします。

   http://www.sendmail.org/vendor/sun/gen-etc-shells.html

2. スーパーユーザーになるか、それと同等の役割になります。

   役割については、『*Solaris* のシステム管理 *(*セキュリティサービス*)*』の「特権付きアプリケーションの使用」を参照してください。

3. シェルのリストを作成するために、gen-etc-shells を実行します。

   ```
   # ./gen-etc-shells.sh> /tmp/shells
   ```

   このスクリプトでは、getent コマンドを使用して、/etc/nsswitch.conf 内にリストされたパスワードファイルソースに組み込まれたシェルの名前を収集します。

4. /tmp/shells 内のシェルのリストを調べて編集します。

   選択したエディタを使用し、組み込まないシェルを削除します。

5. ファイルを /etc/shells に移動します。

   ```
   # mv /tmp/shells /etc/shells
   ```

メールサービスの障害対処とヒント (作業マップ)

次の表では、メールサービスの障害追跡手順とヒントを説明します。

作業	説明	参照先
メール構成をテストする	sendmail 構成ファイルの変更をテストする手順	351 ページの「メール構成をテストする方法」
メール別名を確認する	指定された受信者にメールを配信できるかどうかを確認する手順	352 ページの「メール別名を確認する方法」

作業	説明	参照先
ルールセットをテストする	sendmail ルールセットの入力と戻りを確認する手順	352 ページの「sendmail ルールセットをテストする方法」
他のシステムへの接続を確認する	他のシステムへの接続を確認するためのヒント	353 ページの「他のシステムへの接続を調べる方法」
syslogd プログラムを使ってメッセージを記録する	エラーメッセージ情報を収集するためのヒント	354 ページの「エラーメッセージの記録」
診断情報のその他の情報源を確認する	他の情報源から診断情報を取得するためのヒント	355 ページの「メール診断情報のその他の情報源」

メールサービスの障害回避とヒント (手順)

この節では、メールサービスの問題解決に使用できる手順とヒントをいくつか示します。

▼ メール構成をテストする方法

構成ファイルに対して行なった変更をテストするには、次の手順に従います。

1. 変更した構成ファイルがあるシステムで sendmail を再起動します。

   ```
   # pkill -HUP sendmail
   ```

2. 各システムからテストメッセージを送信します。

   ```
   # /usr/lib/sendmail -v names </dev/null
   ```

   names 受信者の電子メールアドレスを指定

   このコマンドは、指定された受信者に NULL メッセージを送信し、画面にメッセージの動作を表示します。

3. メッセージを通常のユーザー名に送ることによって、メールを自分自身またはローカルシステム上の他の人に送信します。

4. **(**省略可能**)** ネットワークに接続している場合は、別のシステムの個人宛に次の **3** 方向でメールを送信します。

   - メインシステムからクライアントシステムへ

- クライアントシステムからメインシステムへ
- クライアントシステムから別のクライアントシステムへ

5. **(**省略可能**)** メールゲートウェイがある場合、メールホストから別のドメインにメールを送信してメールリレープログラムおよびホストが適切に設定されていることを確認します。

6. **(**省略可能**)** 電話回線上に別のホストへの **UUCP** 接続を設定している場合は、そのホストの誰かにメールを送信し、その個人にメールを返信してもらうか、あるいはその個人がメッセージを受信した時に電話してもらいます。

7. **UUCP** 接続を介してメールを送信するように他の人に頼みます。

   `sendmail` プログラムは、メッセージが配信されたかどうかは検出しません。これは、メッセージが配信のために UUCP に渡されるためです。

8. 異なるシステムからメッセージを `postmaster` 宛てに送信し、ポストマスターのメールボックスに配信されることを確認します。

▼ メール別名を確認する方法

別名を確認し、メールが指定された受信者に配信されるかどうかを確認するには、次の手順に従います。

別名を表示し、最終アドレスが配信可能かどうかを特定します。

`%` **`/usr/lib/sendmail -v -bv`** *recipient*

recipient	受信者の別名を指定

以下に出力例を示します。

```
% /usr/lib/sendmail -v -bv sandy
sandy... aliased to ssmith
ssmith... aliased to sandy@phoenix
sandy@phoenix... deliverable: mailer esmtp, host phoenix, user sandy@phoenix.example.com
%
```

ローカルおよびドメインの両方で別名を使用する場合は、ループやデータベースの不整合が生じないようにしてください。ユーザーをあるシステムから別のシステムに移動する時は、別名のループを作成しないように特に注意してください。

▼ `sendmail` ルールセットをテストする方法

`sendmail` ルールセットの入力と戻りを確認するには、次の手順に従います。

1. アドレステストモードに変更します。

```
# /usr/lib/sendmail -bt
```

2. メールアドレスをテストします。
   最後のプロンプト (>) で次の数値とアドレスを入力します。

```
> 3,0 mail_address
```

mail_address	テストするメールアドレスを指定

3. セッションを終了します。
   Control-D キーを押します。

   以下に出力例を示します。

```
% /usr/lib/sendmail -bt
ADDRESS TEST MODE (ruleset 3 NOT automatically invoked)
Enter <ruleset> <address>
> 3,0 sandy@phoenix
canonify           input: sandy @ phoenix
Canonify2          input: sandy < @ phoenix>
Canonify2        returns: sandy < @ phoenix . example . com .>
canonify         returns: sandy < @ phoenix . example . com .>
parse              input: sandy < @ phoenix . example . com .>
Parse0             input: sandy < @ phoenix . example . com .>
Parse0           returns: sandy < @ phoenix . example . com .>
ParseLocal         input: sandy < @ phoenix . example . com .>
ParseLocal       returns: sandy < @ phoenix . example . com .>
Parse1             input: sandy < @ phoenix . example . com .>
MailerToTriple     input: < mailhost . phoenix . example . com>
     sandy < @ phoenix . example . com .>
MailerToTriple   returns: $# relay $@ mailhost . phoenix . example . com
     $: sandy < @ phoenix . example . com .>
Parse1           returns: $# relay $@ mailhost . phoenix . example . com
     $: sandy < @ phoenix . example . com .>
parse            returns: $# relay $@ mailhost . phoenix . example . com
     $: sandy < @ phoenix . example . com .>
```

▼ 他のシステムへの接続を調べる方法

mconnect プログラムは、指定したホスト上のメールサーバーへの接続を開き、接続をテストできるようにします。プログラムは対話式で実行されるので、さまざまな診断コマンドを実行できます。詳細は、mconnect(1) のマニュアルページを参照してください。次の例では、ユーザー名 sandy へのメールが配信可能かどうかを調べます。

```
% mconnect phoenix

connecting to host phoenix (172.31.255.255), port 25
connection open
```

```
220 phoenix.example.com ESMTP Sendmail 8.12.0+Sun/8.12.0; Sun, 4 Sep 2001 3:52:56 -0700
(PDT)
expn sandy
250 2.1.5 <sandy@phoenix.example.com>
quit
```

mconnect を使用して SMTP ポートに接続できない場合は、次の条件を確認してください。

- システム負荷が高すぎないか
- sendmail デーモンが動作しているか
- システムに適切な /etc/mail/sendmail.cf ファイルがあるか
- ポート 25 (sendmail が使用するポート) がアクティブであるか

エラーメッセージの記録

メールサービスは、syslogd プログラムを使って大部分のエラーメッセージを記録します。デフォルトでは、syslogd プログラムはこれらのメッセージを /etc/hosts ファイルで指定されている loghost というシステムに送信します。loghost が NIS ドメイン全体のすべてのログを保持するように定義できます。loghost を指定しなければ、syslogd からのエラーメッセージはレポートされません。

/etc/syslog.conf ファイルは、syslogd プログラムがメッセージをどこに転送するかを制御します。 /etc/syslog.conf ファイルを編集することにより、デフォルトの構成を変更できます。変更内容を有効にするには、syslog デーモンを再起動する必要があります。メールに関する情報を収集するために、ファイルに次の選択を追加できます。

- mail.alert – ここで訂正する必要のある状態メッセージ
- mail.crit – クリティカルメッセージ
- mail.warning – 警告メッセージ
- mail.notice – エラーではないが注意すべきメッセージ
- mail.info – 情報メッセージ
- mail.debug – デバッグメッセージ

/etc/syslog.conf ファイルの次のエントリは、すべての重大な情報を含むデバッグメッセージを /var/log/syslog に送信します。

```
mail.crit;mail.info;mail.debug              /var/log/syslog
```

システムログの各行には、タイムスタンプ、そのログ行を生成したシステム名、およびメッセージが入っています。syslog ファイルは、大量の情報を記録できます。

ログは、連続したレベルとして並べられます。最下位レベルでは、異常なイベントだけが記録されます。最上位レベルでは、もっとも必須なイベントと注目する必要のないイベントが記録されます。通常、10 以下のログレベルが「有用」とみなされます。10 を超えるログレベルは通常、デバッグに使用されます。loghost および syslogd プログラムの詳細については、『*Solaris* のシステム管理 (上級編)』の「システムのメッセージ記録のカスタマイズ」を参照してください。

メール診断情報のその他の情報源

その他の診断情報については、次の情報源を確認してください。

- メッセージのヘッダーの Received 行を調べます。 これらの行は、メッセージがリレーされる時にとった経路を追跡できます。 時間帯の違いを考慮するのを忘れないでください。
- MAILER-DAEMON からのメッセージを調べます。 これらのメッセージは通常、配信上の問題をレポートします。
- ワークステーショングループの配信上の問題を記録するシステムログを確認します。sendmail プログラムは常に、その処理内容をシステムログに記録します。crontab ファイルを修正してシェルスクリプトを夜間に実行できます。これは、ログから SYSERR メッセージのログを検索し、見つかったメッセージをすべてポストマスターに送信します。
- mailstats プログラムを使ってメールタイプをテストし、着信メッセージと発信メッセージの数を判定します。

エラーメッセージの解釈

この節では、Solaris 9 オペレーティング環境で発生する sendmail 関連のエラーメッセージを解釈し対処する方法について説明します。http://www.sendmail.org/faq/ も参照してください。

以下のエラーメッセージには、次の種類の情報が含まれます。

- 原因 –メッセージ発生の原因となった可能性があるもの
- 説明 –エラーメッセージが発生した時にユーザーが行なっていた操作
- 対処方法 –問題を解決するため、あるいは作業を続けるための操作
- 技術メモ –開発者などの専門家向けの背景情報
- 関連項目 –他の情報源

`451 timeout waiting for input during` *source*

原因 –タイムアウトの可能性があるソース (SMTP 接続など) から読み取るとき、sendmail は、読み込みを開始する前にさまざまな Timeout オプションの値をタイマーに設定します。タイマーが期限切れになる前に読み取りが完了しなかった場合、このメッセージが表示され、読み取りが停止します。通常、この状況は RCPT 時に発生します。メールメッセージは、後で配信するためにキューに入れられます。

対処方法 –このメッセージが頻繁に表示される場合は、/etc/mail/sendmail.cf ファイルの Timeout オプションの値を大きくします。タイマーがすでに大きな値に設定されている場合は、ネットワークの配線や接続などハードウェアの問題点を探します。

関連項目 –Timeout オプションの詳細については、416 ページの「Timeout オプションの変更点」を参照してください。 オンラインマニュアルを使用している場合は、検索語として「timeouts」を指定します。

550 *hostname*... Host unknown
原因 –この sendmail のメッセージは、単価記号 (@) の後のアドレス部分で指定されている受信先のホストマシンが、ドメインネームシステム (DNS) ルックアップ時に見つからなかったことを示します。

対処方法 –nslookup コマンドを使用して、受信先ホストが、そのドメインまたは他のドメインにあることを確認します。スペルが間違っている可能性があります。あるいは、受信者に連絡して正しいアドレスを確認します。

550 *username*... User unknown
原因 –この sendmail のメッセージは、単価記号 (@) の前のアドレス部分で指定されている受信者を受信先ホストマシンで検出できなかったことを示します。

対処方法 –電子メールアドレスを確認し、再度送信してみます。スペルが間違っている可能性があります。これで解決しない場合は、受信者に連絡して正しいアドレスを確認します。

554 *hostname*... Local configuration error
原因 –この sendmail メッセージは通常、ローカルホストがメールを自分宛に送信しようとしていることを示します。

対処方法 –/etc/mail/sendmail.cf ファイル内の $j マクロの値が完全指定ドメイン名になっていることを確認します。

技術メモ –送信側のシステムが SMTP の HELO コマンドで受信側のシステムに自身のホスト名を示すと、受信側のシステムはそのホスト名を送信者の名前と比較します。これらの名前が同じ場合、受信側のシステムはこのエラーメッセージを発行し、接続を閉じます。HELO コマンドで提供される名前は、$j マクロの値です。

関連項目 –追加情報については、 http://www.sendmail.org/faq/section4.html#4.5 を参照してください。

config error: mail loops back to myself.
原因 –MX レコードを設定し、ホスト *bar* をドメイン *foo* のメールエクスチェンジャーにする場合で、かつ、ホスト *bar* に自分がドメイン *foo* のメールエクスチェンジャーであることを認識させる設定をしていない場合、このエラーメッセージを受け取ります。

また、送信側システムと受信側システムの両方が同じドメインとして識別される場合にも、このメッセージを受け取ります。

対処方法 –手順については、http://www.sendmail.org/faq/section4.html#4.5 を参照してください。

host name configuration error
対処方法 –次のエラーメッセージの対処方法で説明されている手順に従います。
554 *hostname*... Local configuration error.

技術メモ – これは sendmail の古いメッセージで、「I refuse to talk to myself」というメッセージから置き換えられたもので現在は、「Local configuration error」メッセージに置き換えられています。

user unknown
説明 –メールをユーザー宛てに送信しようとすると、「Username... user unknown」のエラーが表示されます。ユーザーが同じシステム上にいます。

対処方法 –入力した電子メールアドレスに誤字がないか確認します。あるいは、ユーザーが、/etc/mail/aliases またはユーザーの .mailrc ファイルに存在しない電子メールアドレスに別名を割り当てられている可能性があります。 また、ユーザー名の大文字も確認してください。できれば、電子メールアドレスは大文字と小文字が区別されないようにします。

関連項目 –追加情報について
は、 http://www.sendmail.org/faq/section4.html#4.17 を参照してください。

第 26 章

メールサービス (リファレンス)

sendmail プログラムは、構成ファイルを使用して「別名」変換と転送、ネットワークゲートウェイへの自動ルーティング、柔軟な構成を提供するメール転送エージェントです。 Solaris オペレーティング環境では、ほとんどのサイトで使用できる標準構成ファイルが付属しています。第 24 章 では、メールサービスのコンポーネントと典型的なメールサービスの構成を紹介します。第 25 章 では、電子メールシステムをセットアップして管理する方法について説明します。この章では、以下のトピックについて説明します。

この Solaris 9 の sendmail バージョン 8.12 に含まれる新しい機能については、第 27 章 を参照してください。 mail.local、mailstats、makemap に関する変更、および新しいメンテナンスユーティリティである editmap についてもお読みください。以上の章で説明されていない詳細については、sendmail(1M)、mail.local(1M)、mailstats(1)、makemap(1M)、および editmap(1M) のマニュアルページを参照してください。

Solaris 版 の sendmail

ここでは、以下の項目について sendmail の Solaris 版と一般的な Berkeley バージョンを比較します。

sendmail のコンパイルに使用できるフラグと使用できないフラグ

次に、Solaris 9 に含まれている sendmail のバージョンをコンパイルするときに使用するフラグを示します。構成に他のフラグが必要な場合は、そのソースをダウンロードし、バイナリにコンパイルし直してください。このプロセスについては、http://www.sendmail.org を参照してください。

表 **26–1** 一般的な sendmail フラグ

フラグ	説明
SOLARIS=20900	Solaris 9 オペレーティング環境をサポートする
MILTER	メールフィルター API をサポートする
NETINET6	IPv6 をサポートする。このフラグは、conf.h から Makefile に移動

表 **26–2** マップとデータベースの種類

フラグ	説明
NDBM	ndbm データベースをサポートする
NEWDB	db データベースをサポートする
USERDB	User データベースをサポートする
NIS	nis データベースをサポートする
NISPLUS	nisplus データベースをサポートする
LDAPMAP	LDAP のマップをサポートする
MAP_REGEX	正規表現のマップをサポートする

表 **26–3** Solaris のフラグ

フラグ	説明
SUN_EXTENSIONS	sun_compat.o に含まれる Sun の拡張をサポートする
SUN_LOOKUP_MACRO	sendmail.cf の L と G 構成コマンドをサポートする。この 2 つのコマンドの使用は推奨されない
SUN_DEFAULT_VALUES	Solaris フラグ SUN_CONTENT_LENGTH のデフォルト値だけをサポートする

表 **26–3** Solaris のフラグ　(続き)

フラグ	説明
SUN_INIT_DOMAIN	下位互換性を確保するために、NIS ドメイン名をサポートしてローカルホスト名を完全指定する。詳細は、http://www.sendmail.org のベンダー固有の情報を参照
SUN_CONTENT_LENGTH	ファイルへのメッセージで Content-Length: ヘッダーをサポートする。 詳細は、http://www.sendmail.org のベンダー固有の情報を参照
SUN_SIMPLIFIED_LDAP	Sun 固有の簡略化された LDAP API をサポートする。詳細は、http://www.sendmail.org のベンダー固有の情報を参照
VENDOR_DEFAULT=VENDOR_SUN	Sun をデフォルトのベンダーに選択する

次の表に、Solaris 9 に添付される sendmail のバージョンのコンパイルに使用されない一般的なフラグを示します。

表 **26–4** sendmail の Solaris 版に使用されない一般的なフラグ

フラグ	説明
SASL	Simple Authentication and Security Layer (RFC 2554)
STARTTLS	Transaction Level Security (RFC 2487)

sendmail のコンパイルに使用するフラグの一覧を参照するには、次のコマンドを使用します。

```
% /usr/lib/sendmail -bt -d0.10 < /dev/null
```

注 – 上記のコマンドでは、Sun 固有のフラグは表示されません。

sendmail の代替コマンド

Solaris リリースには、Berkley による汎用リリースで提供されているコマンドの同義語がすべて組み込まれているわけではありません。この表には、コマンドの別名のリストとそれが Solaris リリースに組み込まれているかどうか、および sendmail を使って同じ動作を生成する方法を示しています。

表 **26–5** 代替 sendmail コマンド

代替名	**Solaris** への組み込み	sendmail を使用したオプション
hoststat	組み込まれていない	sendmail -bh
mailq	組み込まれている	sendmail -bp
newaliases	組み込まれている	sendmail -bi
purgestat	組み込まれていない	sendmail -bH
smtpd	組み込まれていない	sendmail -bd

構成ファイルのバージョン

Solaris 9 に含まれている sendmail の バージョンには、sendmail.cf ファイルのバージョンを定義するための構成オプションが含まれます。現在のバージョンの sendmail でも以前のバージョンの構成ファイルを使用できます。バージョンレベルには 0 から 10 の値を設定できます。また、ベンダーの定義もできます。Berkeley または Sun をベンダーとして選択できます。ベンダーを定義しないでバージョンレベルだけを設定した場合は、Sun がデフォルトとして使用されます。次の表に有効なオプションを示します。

表 **26–6** 構成ファイルのバージョン

フィールド	説明
V7/Sun	sendmail のバージョン 8.8 で使用された設定
V8/Sun	sendmail のバージョン 8.9 で使用された設定。この設定は、Solaris 8 に含まれていた
V9/Sun	sendmail のバージョン 8.10 と 8.11 で使用された設定
V10/Sun	sendmail のバージョン 8.12 で使用される設定。バージョン 8.12 は、Solaris 9 のデフォルト

注 – V1/Sun は使用しないでください。詳細は、 http://www.sendmail.org/vendor/sun/differences.html#4 を参照してください。

作業手順については、第 25 章 の 329 ページの「sendmail.cf 構成ファイルの構築 (手順) 」 を参照してください。

メールサービスのソフトウェアとハードウェアのコンポーネント

ここでは、メールシステムのソフトウェアとハードウェアの構成要素について説明します。

ソフトウェアのコンポーネント

各メールサービスには、少なくとも以下のどれかのソフトウェアコンポーネントが含まれます。

ここでは、以下のソフトウェアコンポーネントについても説明します。

メールユーザーエージェント

メールユーザーエージェントは、ユーザーとメール転送エージェント間のインタフェースとして機能するプログラムです。`sendmail` プログラムは、メール転送エージェントです。Solaris のオペレーティング環境は、以下のメールユーザーエージェントを提供します。

- `/usr/bin/mail`
- `/usr/bin/mailx`
- `$OPENWINHOME/bin/mailtool`
- `/usr/dt/bin/dtmail`

メール転送エージェント

「メール転送エージェント」は、メールメッセージのルーティングとメールアドレスの解釈を行います。このエージェントは、メールトランスポートエージェントとも呼ばれます。Solaris オペレーティング環境ソフトウェアの転送エージェントは `sendmail` です。 転送エージェントは次の機能を実行します。

- メールユーザーエージェントからメッセージを受信する
- 宛先アドレスを認識する
- 適切な配信エージェントを選択してメールを配信する
- 他のメール転送エージェントからのメールを受信する

ローカル配信エージェント

「ローカル配信エージェント」は、メールの配信プロトコルを実行するプログラムです。Solaris オペレーティング環境に搭載されているローカル配信エージェントについては以下に述べます。

- UUCP ローカル配信エージェントは `uux` を使ってメールを配信します。
- 標準の Solaris リリースでは `mail.local` であるローカル配信エージェントを配信します。

第 27 章 では、以下の関連項目について説明します。

- 424 ページの「配信エージェントの新しいフラグ」
- 424 ページの「配信エージェントの新しい等号 (=)」
- 429 ページの「`mail.local` の変更点」

メールプログラム

メールプログラムは、`sendmail` 固有の用語です。メールプログラムは `sendmail` によって使用され、カスタマイズしたローカル配信エージェントまたはカスタマイズされたメール転送エージェントの特定のインスタンスを指定します。`sendmail.cf` ファイルに少なくとも 1 つのメールプログラムを指定する必要があります。作業手順については、第 25 章 の 329 ページの「`sendmail.cf` 構成ファイルの構築 (手順) 」を参照してください。 ここでは、2 種類のメールプログラムについて説明します。

- 364 ページの「SMTP (Simple Mail Transfer Protocol) メールプログラム」
- 365 ページの「UUCP (UNIX-to-UNIX Copy Program) メールプログラム」

メールプログラムの詳細については、 http://www.sendmail.org/m4/readme.html または `/usr/lib/mail/README` を参照してください。

SMTP (Simple Mail Transfer Protocol) メールプログラム

SMTP はインターネットで使用される標準のメールプロトコルです。このプロトコルが、メールプログラムを定義します。

- `smtp` は、他のサーバーへの通常 (従来の形式) の SMTP 転送機能を提供します。
- `esmtp` は、他のサーバーへの拡張 SMTP 転送機能を提供します。
- `smtp8` は、8 ビットデータを MIME に変更することなく、他のサーバーに SMTP 転送機能を提供します。

- dsmtp は、F=% メールプログラムフラグを使ってオンデマンド配信機能を提供します。第 27 章 の 423 ページの「MAILER() の宣言についての変更点」 と 424 ページの「配信エージェントの新しいフラグ」 を参照してください。

UUCP (UNIX-to-UNIX Copy Program) メールプログラム

UUCP の使用は、できるだけ避けてください。説明については、http://www.sendmail.org/m4/uucp.html を参照するか、/usr/lib/mail/README で「UUCP メールプログラムの使用」という文字列を検索してください。

UUCP が、メールプログラムを定義します。

uucp-old　$=U クラスの名前が uucp-old に送られます。 uucp は、このメールプログラムの以前の名前です。 uucp-old メールプログラムはヘッダーでは感嘆符を用いるアドレスを使用します。

uucp-new　$=Y クラスの名前が uucp-new に送られます。 受信側の UUCP メールプログラムが単一の転送で複数の受信者を管理できる場合は、このメールプログラムを使用します。suucp は、このメールプログラムの以前の名前です。 uucp-new メールプログラムはヘッダーで感嘆符を用いるアドレスも使用します。

構成に MAILER(smtp) も指定されている場合は、さらに以下の 2 つのメールプログラムが定義されます。

uucp-dom　このメールプログラムは、ドメインスタイルアドレスを使用し、基本的に SMTP のリライトルールを適用します。

uucp-uudom　$=Z クラスの名前が uucp-uudom に送られます。 uucp-uudom と uucp-dom は、ドメインスタイルアドレスという同じヘッダーアドレスフォーマットを使用します。

注 – smtp メールプログラムは UUCP メールプログラムを変更するので、.mc ファイルの MAILER(uucp) の前に必ず MAILER(smtp) を記述します。

メールアドレス

「メールアドレス」には、受信者の名前と、メールメッセージが配信されるシステムが含まれます。ネームサービスを使用しない小さなメールシステムを管理する場合、メールのアドレス指定は簡単です。つまり、ログイン名がユーザーを一意に識別します。メールボックスを含む複数のシステムで構成されるメールシステム、または 1 つ以上のドメインで構成されるメールシステムを管理する場合は複雑になります。UUCP またはその他のメールシステムによって外部に接続する場合は、さらに複雑になります。以下の節で、メールアドレスの各部とその複雑さを説明しています。

- 366 ページの「ドメインとサブドメイン」

ドメインとサブドメイン

電子メールのアドレス指定には、ドメインが使用されます。「ドメイン」は、ネットワークアドレスの命名のためのディレクトリ構造です。ドメインは 1 つ以上のサブドメインを持つことができます。アドレスのドメインとサブドメインは、ファイルシステムの階層と比較できます。サブディレクトリが上位のディレクトリに含まれるように、メールアドレスの各サブドメインもその右のドメインに含まれると考えられます。

次の表に米国における最上位のドメインを示します。

表 **26–7** 最上位のドメイン

ドメイン	説明
`com`	企業
`edu`	教育機関用
`gov`	米国の政府機関
`mil`	米国の軍事機関
`net`	ネットワーク組織
`org`	その他の非営利組織

ドメインには大文字と小文字の区別がありません。アドレスのドメイン部分には、大文字、小文字、またはその両方を区別なく混合して使用できます。

ドメインの詳細は、『 *Solaris* のシステム管理 *(*ネーミングとディレクトリサービス *: DNS*、*NIS*、*LDAP* 編*)*』の「ドメインネームシステム (概要)」を参照してください。

ネームサービスドメイン名とメールドメイン名

ネームサービスドメイン名とメールドメイン名を操作するときは、以下のことに注意します。

- `sendmail` プログラムは、デフォルトで NIS または NIS+ ドメイン名から最初の構成要素を取り除き、メールドメイン名とします。たとえば、NIS+ ドメイン名が `bldg5.example.com` の場合、メールドメイン名は `example.com` になります。
- メールドメインアドレスは大文字と小文字の区別をしませんが、NIS または NIS+ ドメイン名は異なります。メールと NIS または NIS+ ドメイン名を設定するときは、小文字を使用するのが最善です。
- DNS ドメイン名とメールドメイン名は同じでなければなりません。

詳細は、393 ページの「sendmail とネームサービスの相互作用」 を参照してください。

メールアドレスの一般的な書式

一般に、メールアドレスは次のような書式になります。詳細は、367 ページの「経路に依存しないメールアドレス」 を参照してください。

user@subdomain. subdomain2.subdomain1.top-level-domain

アドレスの @ 記号より左の部分はローカルアドレスです。ローカルアドレスには、以下の内容を含めることができます。

- 別のメールトランスポートを使用するルーティングに関する情報 (たとえば、`bob::vmsvax@gateway` または `smallberries%mill.uucp@gateway`)
- 別名 (たとえば、`iggy.ignatz`)

注 – 受信側のメールプログラムでアドレスのローカル部分を解釈する必要があります。 メールプログラムの詳細は、364 ページの「メールプログラム」 を参照してください。

アドレスの @ 記号より右の部分は、ローカルアドレスが位置するドメインレベルを示します。各サブドメインはドットで区切られます。アドレスのドメイン部分は、組織、物理的な場所、または地域を表すことができます。 さらに、ドメイン情報の順序は階層的で、ローカルなサブドメインほど @ 記号に近くなります。

経路に依存しないメールアドレス

メールアドレスは、経路に依存しないアドレス指定ができます。経路に依存しないアドレス指定では、電子メールメッセージの発信者は、受信者の名前と最終の宛先を指定する必要があります。インターネットなどの高速ネットワークでは、経路に依存しないアドレスを使用します。経路に依存しないアドレスは次のような書式になります。

user@host . domain

UUCP 接続の経路に依存しないアドレスは次のような書式になります。

host . domain ! *user*

コンピュータのドメイン階層命名方式が普及したため、経路に依存しないアドレスがより一般的になってきました。実際、次に示すように、もっとも一般的な経路に依存しないアドレスはホスト名を省略し、電子メールメッセージの最終宛先の識別をドメインネームサービスに任せています。

user@domain

経路に依存しないアドレスでは、まず @ 記号を検索し、ドメイン階層を右 (最上位) から左 (@ 記号の右側にあるもっとも固有な部分) へと読み取ります。

メールボックスファイル

「メールボックス」は、電子メールメッセージの最終的な宛先となるファイルです。メールボックス名には、ユーザー名または postmaster などの特定の機能の名前を指定できます。メールボックスは、ユーザーのローカルシステムかリモートのメールサーバーのいずれかの `/var/mail/`*username* ファイルにあります。ただし、いずれの場合でも、メールボックスはメールが配信されるシステム上にあります。

ユーザーエージェントがメールスプールからメールを取り出し、ローカルメールボックスに容易に格納できるように、メールは常にローカルファイルシステムに配信される必要があります。ユーザーのメールボックスの宛先として、NFS でマウントされたファイルシステムを使用しないでください。特にリモートサーバーから `/var/mail` ファイルシステムをマウントしているメールクライアントには、直接メールを送信しないでください。この場合ユーザー宛てのメールは、クライアントのホスト名ではなく、メールサーバーにアドレス指定する必要があります。NFS でマウントされたファイルシステムは、メールの配信と処理に問題を起こすことがあります。

`/etc/mail/aliases` ファイルと NIS や NIS+ といったネームサービスは、電子メールのアドレスに別名を作成するメカニズムを持っているため、ユーザーは、ユーザーのメールボックスの正確なローカル名を知る必要はありません。

次の表に、特殊な目的のメールボックスに対する共通の命名規則をいくつか示します。

表 **26–8** メールボックス名の書式についての規則

書式	説明
username	多くの場合、ユーザー名はメールボックス名と同じ
Firstname .Lastname *Firstname _Lastname* *Firstinitial .Lastname* *Firstinitial_Lastname*	ユーザー名は、ドット (または下線) でファーストネームとラストネームに区切ったフルネームか、あるいはファーストネームがイニシャルで、ドット (または下線) でイニシャルとラストネームを区切ったもの
`postmaster`	ユーザーは、`postmaster` のメールボックスに質問を送ったり、問題点を報告したりできる。通常は各サイトとドメインに `postmaster` メールボックスがある
`MAILER-DAEMON`	`sendmail` は、`MAILER-DAEMON` 宛てのメールを自動的にポストマスターに送る
aliasname `-request`	`-request` で終わる名前は、配布リストの管理アドレス。このアドレスは、配布リストを管理する人にメールをリダイレクトする
`owner-` *aliasname*	`owner-` で始まる名前は、配布リストの管理アドレス。このアドレスは、メールエラーを処理する人にメールをリダイレクトする

表 **26–8** メールボックス名の書式についての規則　(続き)

書式	説明
`owner-owner`	この別名は、エラーを戻す先の `owner-`*aliasname* の別名がない場合に使用される。このアドレスは、メールエラーを処理する人にメールをリダイレクトし、大量の別名を管理する任意のシステムで定義される
local `%`*domain*	パーセント記号 (`%`) は、メッセージがその宛先に着くと展開されるローカルアドレスを示す。ほとんどのメールシステムは、`%` 記号つきのメールボックス名を全メールアドレスとして翻訳する。`%` は `@` と置き換えられ、メールはそれに応じてリダイレクトされる。多くの人が `%` を使用するが、これは正式な標準ではない。この規則は、電子メールの世界では「パーセントハック」と呼ばれている。この機能は、メールに問題が起こった場合にデバッグに使用されることが多い

`sendmail` バージョン 8 より、所有者の別名が存在する場合、グループの別名に送信されるメールの封筒の送信者は、所有者の別名から展開されるアドレスに変更されました。この変更によって、メールエラーは、送信者に返送されるのではなく、別名の所有者に送信されるようになりました。この変更によって、別名に送信されたメールは、別名の所有者から送信されたように見えます。次の別名の書式は、この変更に関連したいくつかの問題に対応します。

```
mygroup: :include:/pathname/mygroup.list
owner-mygroup: mygroup-request
mygroup-request: sandys, ignatz
```

この例では、`mygroup` の別名が、このグループの実際のメール別名です。`owner-mygroup` の別名は、エラーメッセージを受信します。`mygroup-request` の別名は、管理の要求に使用してください。この構造は、`mygroup` の別名に送信されたメールでは、封筒の送信者が `mygroup-request` に変更されることを意味します。

メール別名

別名 (alias) とは、もう 1 つの別の名前を指します。電子メールでは、メールボックスの場所を割り当てたり、メールリストを定義したりするために別名を使用できます。作業マップについては、第 25 章 の 332 ページの「メール別名ファイルの管理 (作業マップ)」 を参照してください。 この章の 386 ページの「メール別名ファイル」 も参照してください。

大きなサイトでは通常、メール別名は、メールボックスの場所を定義します。メール別名を提供することは、複数の部屋を占有する大きな会社の個人のアドレスに部屋番号を含めるようなものです。部屋番号を提供しない場合は、メールは中央アドレスに配信されます。部屋番号がなければ、ビルの内部のどこにメールを配信するかを特定するために余分な労力が必要になり、誤りが発生する可能性も増加します。たとえば、同じ建物に Kevin Smith という名前の人が 2 人いる場合、一方だけがメールを受け取ることになる可能性があります。この問題を解決するには、それぞれの Kevin Smith のアドレスに部屋番号を追加する必要があります。

メールリストを作成するときは、なるべくドメインの場所に依存しないアドレスを使用してください。別名ファイルの移植性と柔軟性を高めるため、別名エントリをできるかぎり一般的でシステムに依存しない形式にしてください。 たとえば、システム mars のドメイン example.com に ignatz というユーザー名がある場合、別名は ignatz@mars ではなく、ignatz@example としてください。ユーザー ignatz がシステム名を変更しても、example ドメインには存在し続ける場合、システム名の変更を反映するように別名ファイルを更新する必要はありません。

別名エントリを作成するときは、1 行ごとに 1 つの別名を入力します。ユーザーのシステム名を含むエントリは 1 つだけにしてください。たとえば、ユーザー ignatz には、次のエントリを作成できます。

```
ignatz: iggy.ignatz
iggyi: iggy.ignatz
iggy.ignatz: ignatz@mars
```

ローカル名やドメインに別名を作成できます。たとえば、システム mars にメールボックスがあり、ドメイン planets 内のユーザー fred の別名エントリでは、NIS+ 別名テーブルに次のエントリを作成できます。

```
fred: fred@planets
```

ドメイン外のユーザーを含むメールリストを作成するときは、ユーザー名とドメイン名を持つ別名を作成してください。たとえば、example.com ドメインの privet システムに smallberries というユーザーが存在する場合は、smallberries@example.com という別名を作成します。送信者の電子メールアドレスは、メールがユーザードメイン外に発信されるときは、完全指定ドメイン名に自動的に変換されます。

以下に、メール別名のファイルを作成して管理する方法を示します。

- NIS+ mail_aliases テーブル、NIS aliases マップ、または、ローカルの /etc/mail/aliases ファイルでグローバルに使用するメール別名を作成します。また、同じ別名ファイルを使用するメールリストを作成して管理することができます。
- メールサービスの構成に応じて、NIS または NIS+ ネームサービスを使って別名を管理し、グローバル aliases データベースを維持したり、ローカルの /etc/mail/aliases ファイルをすべて同時に更新することにより、別名を同一にできます。
- また、ユーザー自身が別名を作成して使用できます。ユーザーは、別名をユーザーだけが使用できるようにローカル ~/.mailrc ファイルで作成することも、誰でも使用できるようにローカル /etc/mail/aliases ファイルで作成することもできます。通常の場合、ユーザーは NIS や NIS+ 別名ファイルの作成および管理はできません。

ハードウェアコンポーネント

メールの構成に必要な 3 つの要素は、単一のシステムによって提供することも個別のシステムによって提供することもできます。

- 371 ページの「メールホスト」
- 371 ページの「メールサーバー」
- 372 ページの「メールクライアント」

ユーザーがドメイン外のネットワークと通信をするためには、4 番目の要素であるメールゲートウェイを追加する必要があります。詳細は、372 ページの「メールゲートウェイ」 を参照してください。次の節では各ハードウェアコンポーネントについて説明しています。

メールホスト

「メールホスト」は、ネットワークのメインのメールマシンに指定するマシンです。メールホストはサイトにおいて、他のシステムでは配信できないメールを転送するためのマシンになります。`hosts` データベースにシステムをメールホストとして指定するには、ローカルの `/etc/hosts` ファイルか、ネームサービスのホストファイルで、IP アドレスの右に `mailhost` を追加します。メールホストシステムでは、`main.cf` ファイルもメール構成ファイルとして使用する必要があります。作業手順については、第 25 章 の 325 ページの「メールホストを設定する方法」 を参照してください。

メールホストとして適切なのは、ローカルエリアネットワーク上のシステムで、電話回線に PPP または UUCP リンクを設定するためのモデムがあるものです。もう 1 つの候補は、ネットワークからグローバルなインターネットネットワークへのルーターとして構成されたシステムです。詳細は、第 29 章、第 38 章、および『*Solaris* のシステム管理 *(IP* サービス*)*』の「ルーターの構成」を参照してください。ローカルネットワークのどのシステムにもモデムがない場合は、その中の 1 つをメールホストに指定します。

サイトの中には、タイムシェアリング構成でネットワークに接続されていないスタンドアロンのマシンを使用するものがあります。つまり、スタンドアロンのマシンが、シリアルポートに接続された端末として機能する場合です。このような構成では、スタンドアロンのシステムを 1 つのシステムネットワークのメールホストに指定することで、電子メールを設定できます。第 24 章 の 313 ページの「ハードウェアコンポーネントの概要」に、典型的な電子メール構成を示す図があります。

メールサーバー

「メールボックス」は、特定のユーザーの電子メールを含む単一のファイルです。メールは、ローカルマシンまたはリモートサーバーのユーザーのメールボックスが存在するシステムに配信されます。「メールサーバー」は、`/var/mail` ディレクトリにユーザーのメールボックスを保持しているいずれかのシステムになります。作業手順については、第 25 章 の 321 ページの「メールサーバーを設定する方法」 を参照してください。

メールサーバーはクライアントからすべてのメールをルーティングします。クライアントがメールを送信するときに、メールサーバーは配信のためそのメールをキューに入れます。 メールがキューに入れられたら、ユーザーはこれらのメールメッセージを失わずに、クライアントをリブートしたり、電源を切ったりすることができます。 受

信者がクライアントからメールを受け取ると、メッセージの「From」行のパスには、メールサーバー名が含まれます。受信者が応答すると、その応答はユーザーのメールボックスに送られます。メールサーバーとして適しているのは、ユーザーにホームディレクトリを提供するシステムか、定期的にバックアップされるシステムです。

メールサーバーがユーザーのローカルシステムでない場合は、構成内で NFS ソフトウェアを使用するユーザーは、/etc/vfstab ファイル (root アクセスがある場合) を使用するか、オートマウンタを使用して、/var/mail ディレクトリをマウントできます。NFS サポートが利用できない場合、ユーザーはサーバーにログインしてメールを読み込めます。

ネットワーク上のユーザーが、オーディオファイル、DTP システムからのファイルなど他の形式のファイルを送信する場合は、メールボックスのメールサーバーには、さらに多くの領域を割り当てる必要があります。

全メールボックス用に 1 台のメールサーバーを設定する利点の 1 つは、バックアップが簡単になることです。メールが多くのシステムに分散しているとバックアップ作業が困難になる場合があります。1 台のサーバーに多くのメールボックスを保存する場合の短所は、サーバーに障害が発生した場合に多くのユーザーが影響を受けることです。ただし、十分なバックアップ機能を提供すれば、1 台のサーバーを採用する価値があります。

メールクライアント

「メールクライアント」は、メールサーバーでメールを受信し、ローカルの /var/mail のないシステムです。 このような構成は、リモートモードと呼ばれます。リモートモードは、デフォルトでは /etc/mail/subsidiary.cf で使用することができます。

メールクライアントには、/etc/vfstab ファイルに適切なエントリがあり、メールサーバーからメールボックスをマウントするマウント先があることを確認する必要があります。またクライアントの別名の宛先が、クライアント名ではなく、メールサーバーのホスト名になっていることを確認してください。作業手順については、第 25 章 の 323 ページの「メールクライアントを設定する方法」 を参照してください。

メールゲートウェイ

「メールゲートウェイ」は、異なる通信プロトコルを実行するネットワーク間の接続を処理したり、同じプロトコルを使用する異なるネットワーク間の通信を処理するマシンです。たとえば、メールゲートウェイでは、SNA (Systems Network Architecture) プロトコルセットを実行するネットワークに、TCP/IP ネットワークを接続する場合もあります。

設定のもっとも簡単なメールゲートウェイは、同じプロトコルかメールプログラムを使用する 2 つのネットワークを接続するものです。 このシステムでは、sendmail がドメインで受信者を見つけられないアドレスのあるメールを処理します。 メールゲートウェイがある場合、sendmail はこれを使用して、ドメイン外でメールの送受信を行います。

2 つのネットワーク間には、次の図に示すように内容の異なるメールプログラムを使ってメールゲートウェイを設定できます。この構成をサポートするには、メールゲートウェイシステムで sendmail.cf ファイルをカスタマイズする必要がありますが、これは困難で時間のかかる作業になる場合もあります。

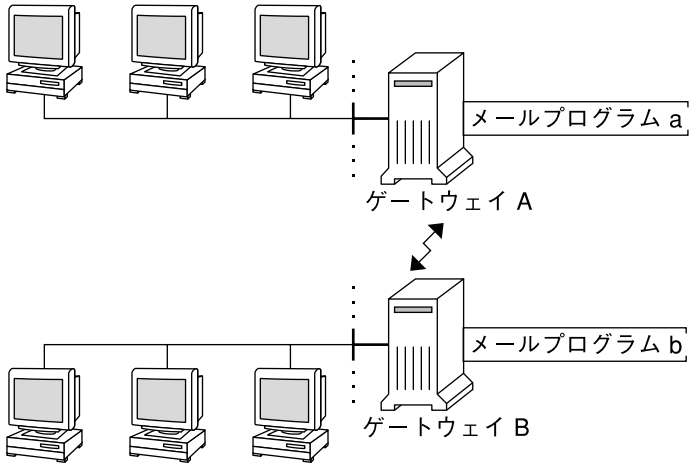


図 **26–1** 異なる通信プロトコル間のゲートウェイ

メールゲートウェイを設定する場合に、必要とするものにもっとも近いゲートウェイ構成ファイルを見つけ、状況に合わせて修正する必要があります。

インターネットに接続できるマシンがある場合は、そのマシンをメールゲートウェイとして構成できます。メールゲートウェイを構成するときは、まずサイトのセキュリティ要件を慎重に考慮する必要があります。社内ネットワークを外部と接続するには、ファイアウォールゲートウェイを構築し、それをメールゲートウェイとして設定する必要がある場合があります。作業手順については、第 25 章 の 326 ページの「メールゲートウェイを設定する方法」 を参照してください。

メールサービスのプログラムとファイル

メールサービスには、相互に対応する数多くのプログラムやデーモンが含まれています。ここでは、電子メールの管理に関連するファイル、プログラム、用語、および概念について説明します。

/usr/bin ディレクトリの内容

次の表にメールサービスに使用する /usr/bin ディレクトリの内容を示します。

名前	形式	説明
aliasadm	ファイル	NIS+ 別名マップを処理するプログラム
mail	ファイル	ユーザーエージェント
mailcompat	ファイル	メールを SunOS 4.1 メールボックスフォーマットに格納するフィルタ
mailq	リンク	/usr/lib/sendmail へのリンク。メールキューを表示するために使用
mailstats	ファイル	/etc/mail/sendmail.st ファイルに格納されたメール統計情報の読み込みに使用するプログラム (存在する場合のみ)
mailx	ファイル	ユーザーエージェント
mconnect	ファイル	アドレスの検証とデバッグのためメールプログラムに接続するプログラム
praliases	ファイル	別名データベースを表示するコマンド。praliases(1) のマニュアルページにあるコンパイルされていない情報を参照
rmail	リンク	/usr/bin/mail へのリンク。メール送信だけに使用されるコマンド
vacation	ファイル	メールへの自動応答を設定するコマンド

/etc/mail ディレクトリの内容

次の表に、/etc/mail ディレクトリの内容を示します。

名前	形式	説明
Mail.rc	ファイル	mailtool ユーザーエージェントのデフォルトの設定値
aliases	ファイル	メール転送情報
aliases.db	ファイル	メール転送情報のバイナリ形式 (newaliases の実行によって作成される)
aliases.dir	ファイル	メール転送情報のバイナリ形式 (newaliases の実行によって作成される)。まだ使用できるが、Solaris 9 ではデフォルトでは使用されない
aliases.pag	ファイル	メール転送情報のバイナリ形式 (newaliases の実行によって作成される)。まだ使用できるが、Solaris 9 ではデフォルトでは使用されない
mailx.rc	ファイル	mailx ユーザーエージェントのデフォルトの設定値
main.cf	ファイル	メインシステム用の構成ファイルの例
relay-domains	ファイル	リレーを許容するすべてのドメインのリスト。デフォルトでは、ローカルドメインだけが使用できる
sendmail.cf	ファイル	メールルーティング用の構成ファイル
submit.cf	ファイル	メール送信プログラム (MSP) のための新しい構成ファイル。詳細は、400 ページの「新しい構成ファイル submit.cf」 を参照
local-host-names	ファイル	メールホスト用の別名の数が多すぎるときに作成可能なオプションファイル
helpfile	ファイル	SMTP HELP コマンドで使用するヘルプファイル
sendmail.pid	ファイル	リスニングデーモンの PID をリストし、現在は /var/run にあるファイル
sendmail.st	ファイル	sendmail 統計ファイル。このファイルが存在すると、sendmail は各メールプログラムのトラフィック量をログに記録する
subsidiary.cf	ファイル	サブシステム用の構成ファイルの例

名前	形式	説明
trusted-users	ファイル	特定のメール操作を実行するための信頼を与えられたユーザーをリストするファイル (各行 1 ユーザー)。デフォルトでは、root だけがこのファイルに入っている。信頼されていないユーザーが特定のメール操作を実行すると、X-Authentication-Warning: header being added to a message という警告が生成される

/usr/lib ディレクトリの内容

表 26–9 にメールサービスに使用する /usr/lib ディレクトリの内容を示します。

表 **26–9** /usr/lib ディレクトリの内容

名前	形式	説明
mail.local	ファイル	メールボックスにメールを配信するメールプログラム
sendmail	ファイル	メール転送エージェントとしても知られるルーティングプログラム
smrsh	ファイル	sendmail の |program 構文を使用して /var/adm/sm.bin ディレクトリにあるプログラムに対して sendmail を実行できるプログラムを制限するシェルプログラム (sendmail に限定されたシェル)。/var/adm/sm.bin に含める内容については、smrsh (1M) のマニュアルページを参照。有効にするには、この m4 コマンドと FEATURE('smrsh') を mc ファイルに含める

/usr/lib/mail ディレクトリの内容

/usr/lib ディレクトリには、sendmail.cf ファイルを構築するために必要なすべてのファイルを含む mail というサブディレクトリがあります。表 26–10 に mail ディレクトリの内容を示します。

表 **26–10** メールサービスに利用する /usr/lib/mail ディレクトリの内容

名前	形式	説明
README	ファイル	構成ファイルの説明
cf	ディレクトリ	ホストのサイトに依存する、およびサイトに依存しない説明

表 **26–10** メールサービスに利用する /usr/lib/mail ディレクトリの内容　(続き)

名前	形式	説明
cf/main.mc	ファイル	以前は cf/main-v7sun.mc という名前のファイル 。メインの構成ファイル
cf/makefile	ファイル	新しい構成ファイルを作成する場合の規則を提供する
cf/submit.mc	ファイル	メッセージを送信するためのメール差し出しプログラム (MSP) のための構成ファイル
cf/subsidiary.mc	ファイル	以前は cf/subsidiary-v7sun.mc という名前のファイル。別のホストから /var/mail を NFS マウントするホストのための構成ファイル
domain	ディレクトリ	サイトに依存するサブドメインの説明
domain/generic.m4	ファイル	Berkeley からの汎用ドメインファイル
domain/solaris-antispam.m4	ファイル	sendmail 関数を以前の Solaris 版のようにする変更を伴うドメインファイル。ただし、リレーは完全に無効に設定されるので、ホスト名のない送信者アドレスは拒否され、解決されないドメインは拒否される
domain/solaris-generic.m4	ファイル	sendmail 関数を以前の Solaris 版のようにする変更を伴うデフォルトドメインファイル
feature	ディレクトリ	特定のホスト用の特別な機能の定義を含む (機能の詳細な説明は README を参照)
m4	ディレクトリ	サイトに依存しないインクルードファイルを含む
mailer	ディレクトリ	local、smtp、uucp などのメールプログラムの定義を含む
ostype	ディレクトリ	各種のオペレーティングシステム環境の説明
ostype/solaris2.m4	ファイル	デフォルトのローカルメールプログラムを mail.local に定義する
ostype/solaris2.ml.m4	ファイル	デフォルトのローカルメールプログラムを mail.local に定義する
ostype/solaris2.pre5.m4	ファイル	ローカルメールプログラムを mail に定義する

表 **26–10** メールサービスに利用する /usr/lib/mail ディレクトリの内容　(続き)

名前	形式	説明
ostype/solaris8.m4	ファイル	ローカルメールプログラムを LMTP モードで mail.local に定義し、IPv6 を有効にし、sendmail.pid ファイルのディレクトリとして /var/run を指定する
sh	ディレクトリ	m4 構築プロセスと移行補助に使用するシェルスクリプトを含む
sh/check-permissions	ファイル	include: 別名と .forward ファイルのアクセス権、および正確なアクセス権に必要なこれらの親ディレクトリのパスを確認する
sh/check-hostname	ファイル	sendmail が完全指定のホスト名を判別できることを確認する

メールサービスに使用するその他のファイル

メールサービスは、その他のいくつかのファイルおよびディレクトリを使用します。これらを 表 26–11 に示します。

表 **26–11** メールサービスに使用するその他のファイル

名前	形式	説明
sendmailvars.org_dir	テーブル	sendmailvars ファイルの NIS+ バージョン
/etc/default/sendmail	ファイル	sendmail の起動スクリプトの環境変数をリストする
/etc/shells	ファイル	有効なログインシェルをリストする
/usr/sbin/editmap	ファイル	sendmail のデータベースマップの単一のレコードに対してクエリーを実行して編集する
/usr/sbin/in.comsat	ファイル	メール通知デーモン
/usr/sbin/makemap	ファイル	入力されたマップのバイナリ形式を構築する
/usr/sbin/newaliases	リンク	/usr/lib/sendmail へのリンク 。別名データベースのバイナリ形式を作成するために使用する。以前は /usr/bin にあった

表 **26–11** メールサービスに使用するその他のファイル　(続き)

名前	形式	説明
/usr/sbin/syslogd	ファイル	sendmail が使用するエラーメッセージログをとるデーモン
/usr/sbin/etrn	ファイル	クライアント側リモートメールキューを起動するための Perl スクリプト
/usr/dt/bin/dtmail	ファイル	CDE メールユーザーエージェント
/var/mail/*mailbox1*、/var/mail/*mailbox2*	ファイル	配信されたメールのメールボックス
/var/spool/clientmqueue	ディレクトリ	クライアントデーモンによって配信されるメールの記憶領域
/var/spool/mqueue	ディレクトリ	マスターデーモンによって配信されるメールの記憶領域
$OPENWINHOME/bin/mailtool	ファイル	ウィンドウベースのメールユーザーエージェント
/var/run/sendmail.pid	ファイル	リスニングデーモンの PID を表示するファイル

メールプログラム間の相互作用

メールサービスは以下のプログラムで構成され、図 26–2 のように作用します。

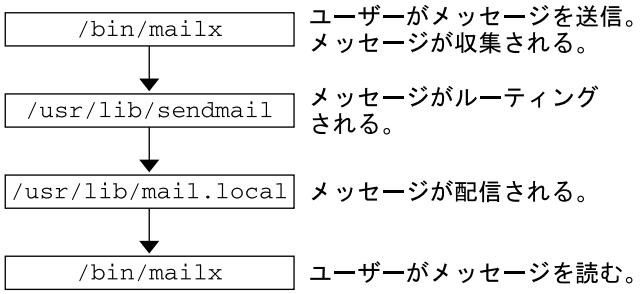


図 **26–2** メールプログラム間の相互作用

さらに詳しい図については、383 ページの「sendmail プログラムの機能」 を参照してください。

以下に、メールプログラムの相互作用について説明します。

1. ユーザーは、mailx 、mailtool などのプログラムを使ってメッセージを送信します。これらのプログラムについては、mailx(1) または mailtool(1) のマニュアルページを参照してください。

2. メッセージは、メッセージを生成したプログラムにより収集され、sendmail デーモンに渡されます。
3. sendmail デーモンがメッセージのアドレスを識別可能な各部に分割して解析します。sendmail デーモンは、/etc/mail/sendmail.cf という構成ファイルの情報を使ってネットワーク名の構文、別名、転送情報、およびネットワークトポロジを決定します。sendmail はこの情報を使用して、メッセージが受信者に到達する経路を決定します。
4. sendmail デーモンはメッセージを適切なシステムに渡します。
5. ローカルシステムの /usr/lib/mail.local プログラムは、メッセージの受信者の /var/mail/*username* ディレクトリのメールボックスにメールを配信します。
6. 受信者は、メールが届いたことが通知されるので、mail、mailx、mailtool などのプログラムを使用してこれを受け取ります。

sendmail プログラム

以下に、 sendmail プログラムの機能の一部を示します。

- sendmail は、TCP/IP や UUCP などの異なる通信プロトコルを使用できます。
- sendmail は、SMTP サーバー、メッセージキュー、メーリングリストを実装します。
- sendmail は、以下の命名規則に準拠したパターンマッチングシステムを使って名前の解釈を制御します。
  - ドメインベースの命名規則ドメインの手法は、物理的なネーミング対論理的なネーミングの問題を分離します。詳細は、365 ページの「メールアドレス」 を参照してください。
  - 他のネットワークのホストからローカルに見えるネットワーク名を提供するなどの即席のテクニック
  - 任意 (以前) の命名構文
  - 異種の命名スキーム

Solaris オペレーティング環境では、sendmail プログラムをメールルーターとして使用します。 以下に、機能の一部を示します。

- sendmail は、電子メールメッセージの受信と配信を担当します。
- sendmail は、mail、mailx、 mailtool などのメール読み出しプログラムと uucp などのメール転送プログラムとのインタフェースです。
- sendmail は、次の要領でユーザーが送信する電子メールメッセージを制御します。
  - 受信者のアドレスを確認します。
  - 適切な配信プログラムを選択します。
  - アドレスを配信エージェントが処理できるフォーマットに書き換えます。
  - 必要に応じて、メールヘッダーをフォーマットし直します。

- 最後に転送されたメッセージをメール配信プログラムに渡します。

sendmail の詳細は、以下のトピックを参照してください。

- 381 ページの「sendmail とその再ルーティングメカニズム」
- 383 ページの「sendmail プログラムの機能」
- 384 ページの「sendmail 構成ファイル」

sendmail とその再ルーティングメカニズム

sendmail プログラムでは、メールルーティングに必要な 3 つのメカニズムをサポートしています。 適切なメカニズムは、変更の種類によって決まります。

- サーバーの変更
- ドメイン全体の変更
- 単独のユーザーの変更

さらに、選択する再ルーティングメカニズムによって必要な管理レベルが異なります。次のオプションを考慮してください。

1. 別名再ルーティングメカニズム

   別名を使用すれば、使用するファイルの種類に基づいて、サーバー全体またはネームサービス全体をベースにしてアドレス名をマップできます。

   以下に、ネームサービスの別名の長所と短所を示します。

   - NIS や NIS+ などのネームサービス別名ファイルを使用すれば、メール再ルーティングの変更を単一のソースで管理できます。ただし、ネームサービスの別名指定では、再ルーティングの変更を伝達する際に遅延が起こります。
   - 通常、ネームサービスの管理は、特定のシステム管理者グループに制限されます。一般ユーザーは、このファイルを管理しません。

   以下に、サーバー別名ファイルを使用する際の長所と短所を示します。

   - サーバー別名ファイルを使用すれば、指定されたサーバーの root になることができる任意のユーザーが再ルーティングを管理できます。
   - サーバー別名指定は、再ルーティングの変更を伝達する際の遅延はほとんどありません。
   - 変更はローカルサーバーだけに影響します。ほとんどのメールが単一のサーバーに送信される場合は、影響が少なくなります。ただし、この変更を多くのメールサーバーに伝達する必要がある場合は、ネームサービスの別名指定を使用します。
   - 一般ユーザーは、この変更を管理しません。

   詳細は、この章の 386 ページの「メール別名ファイル」 を参照してください。 作業マップについては、第 25 章 の 332 ページの「メール別名ファイルの管理 (作業マップ)」 を参照してください。
2. 次のメカニズムは、転送です。

このメカニズムでは、ユーザーがメールの再ルーティングを管理できます。ローカルユーザーは、受信メールを以下の対象に再ルーティングできます。

- 別のメールボックス
- 別のメールプログラム
- 別のメールホスト

このメカニズムは、`.forward` ファイルによってサポートされます。`.forward` ファイルの詳細は、この章の 389 ページの「`.forward` ファイル」 を参照してください。作業マップについては、第 25 章 の 348 ページの「`.forward` ファイルの管理 (作業マップ)」 を参照してください。

3. 最後のメカニズムは、取り込みです。

   このメカニズムでは、`root` アクセス権を持たないユーザーも別名リストを保守できます。このメカニズムを提供するには、`root` ユーザーは、サーバー上の別名ファイル内に適切なエントリを作成する必要があります。このエントリが作成されると、ユーザーは必要に応じてメールをルーティングし直すことができるようになります。取り込みの詳細は、この章の 386 ページの「`/etc/mail/aliases` ファイル」 を参照してください。 作業マップについては、第 25 章 の 332 ページの「メール別名ファイルの管理 (作業マップ)」 を参照してください。

図 26–3 は、`sendmail` がユーザー別名をどのように使用するかを示します。`/usr/bin/mailx` のようなメールを読み取るプログラムは、プログラム自身の別名を持つことができ、それらはメッセージが `sendmail` に達する前に展開されます。`sendmail` の別名は、多くのネームサービスのソース (ローカルファイル、NIS、NIS+) からのものでもかまいません。検索順序は `nsswitch.conf` ファイルによって決定されます。`nsswitch.conf(4)` のマニュアルページを参照してください。

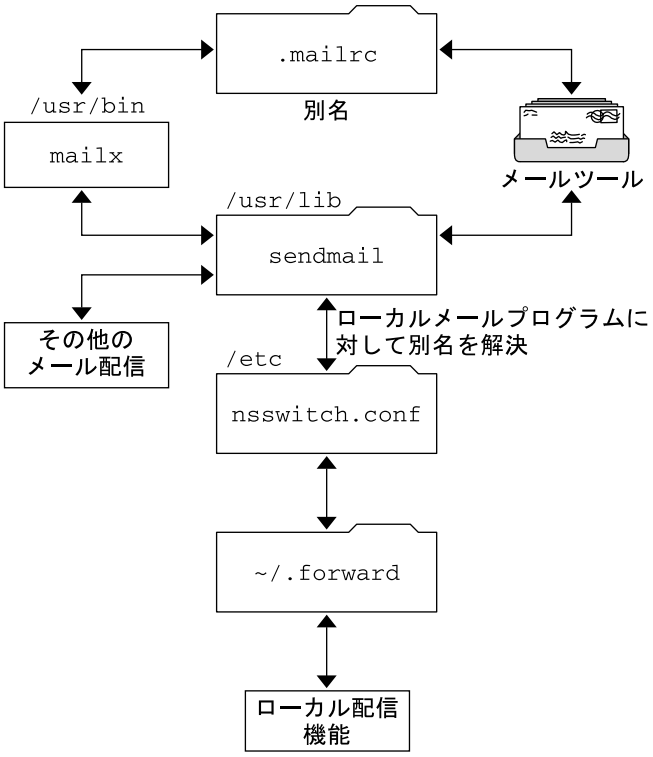


図 **26–3** sendmail が別名を使用する方法

sendmail プログラムの機能

sendmail プログラムには、次のような機能があります。

- sendmail は、信頼性の高いプログラムです。すべてのメッセージを正しく配信するように設計されています。どのようなメッセージも完全に失われることはありません。
- sendmail は、既存のソフトウェアを配信に随時使用します。
- sendmail は、1 つのネットワークタイプ (UUCP や Ethernet など) に複数の接続を行う場合なども含め、複雑な環境を処理するように構成できます。sendmail は、名前とその構文を確認し、どのメールプログラムを使用するかを判断します。
- sendmail は、構成情報をコードにコンパイルする代わりに、構成ファイルを使ってメール構成を制御します。
- ユーザーは独自のメーリングリストを管理できます。各ユーザーは、ドメイン全体で有効な別名ファイル (通常、NIS または NIS+ によって管理されるドメイン全体の別名の中にある) を修正することなく自分自身の転送メカニズムを指定できます。

- 各ユーザーは、受信メールを処理するカスタムメールプログラムを指定できるので、次のようなメッセージを返すこともできます。 私は休暇中です。詳細は、vacation(1) マニュアルページを参照してください。
- sendmail は、1 つのホストでアドレスを処理し、ネットワークトラフィックを削減します。

図 26–4 には、sendmail がメールシステムで他のプログラムと相互作用する方法を示します。

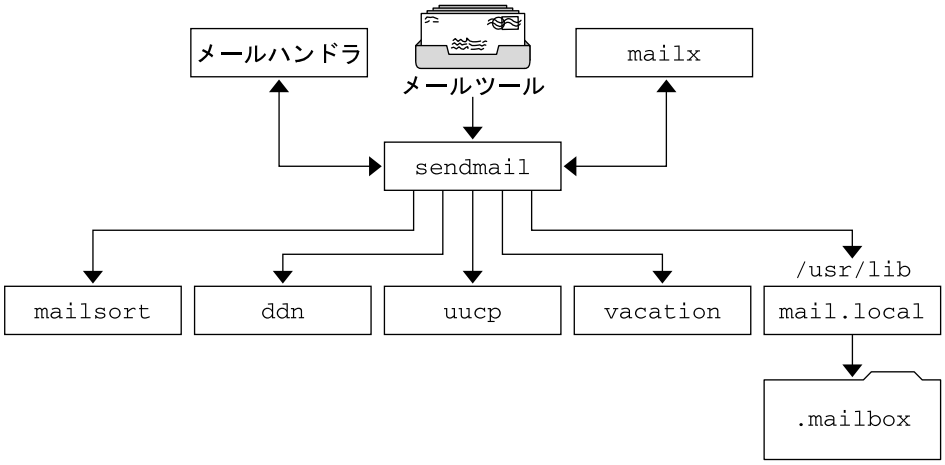


図 **26–4** sendmail と他のメールプログラムとの対話

図 26–4 に示すように、ユーザーはメール作成プログラムおよびメール送信プログラムと対話できます。メール送信が依頼されると、メール生成プログラムは sendmail を呼び出し、sendmail は適切なメールプログラムにメッセージを送信します。 発信者の一部はネットワークサーバーであったり、またメールプログラムの一部はネットワーククライアントであるため、sendmail は、インターネットメールゲートウェイとしても使用できます。このプロセスの詳細は、379 ページの「メールプログラム間の相互作用」 を参照してください。

sendmail 構成ファイル

構成ファイルは、sendmail がその機能を実行する方法を制御します。構成ファイルにより、配信エージェント、アドレスの変換の規則、およびメールヘッダーのフォーマットが選択されます。

sendmail プログラムは、/etc/mail/sendmail.cf ファイルの情報を使用して、その機能を実行します。各システムには、/etc/mail ディレクトリにインストールされたデフォルトの sendmail.cf ファイルがあります。メールサーバーまたはメールクライアントのためにデフォルト構成ファイルを編集または変更する必要はありません。カスタマイズされた構成ファイルを必要とするシステムは、メールホストとメールゲートウェイだけです。

Solaris オペレーティング環境には、以下に示すように、/etc/mail ディレクトリに 3 つのデフォルト構成ファイルがあります。

1. メールホストまたはメールゲートウェイとして使用する 1 つのシステム (または複数のシステム) を指定するための main.cf という名前の構成ファイル
2. デフォルトの sendmail.cf ファイルのコピーで subsidiary.cf という名前の構成ファイル
3. デーモンモードの代わりにメール送信プログラムモードで sendmail を実行するために使用する submit.cf という名前の構成ファイル詳細は、400 ページの「新しい構成ファイル submit.cf」 を参照してください。

システムで使用する構成ファイルは、メールサービスのシステムの役割によって異なります。

- メールクライアントまたはメールサーバーについては、デフォルト構成ファイルを設定または編集する必要はありません。
- メールホストやメールゲートウェイを設定するには、main.cf ファイルをコピーし、それを /etc/mail ディレクトリで sendmail.cf と名称変更します。次に、 sendmail の構成ファイルを再構成して、メールリレープログラムを設定して、メール設定に必要なホストパラメータをリレーします。 作業手順については、第 25 章 の 320 ページの「メールサービスの設定 (作業マップ)」 または 329 ページの「sendmail.cf 構成ファイルの構築 (手順) 」 を参照してください。

次に、サイトの要求に応じて変更が可能な構成パラメータをいくつか説明します。

- 以下の情報を指定する時間値
  - 読み取りのタイムアウト。416 ページの「Timeout オプションの変更点」を参照してください。
  - メッセージが送信者に返送されるまで、配信されずにキューに置かれる時間。425 ページの「キューの新しい機能」 を参照してください。作業マップについては、344 ページの「キューディレクトリの管理 (作業マップ)」 を参照してください。
- メール配信の速度を指定する配信 (delivery) モード
- 長いメッセージ、多くの受信者へのメッセージ、および長時間ダウンしているサイトへのメッセージを配信しないことにより、ビジー期間中の効率を高めるためのロード制限
- ログ出力する問題の種類を指定するログレベル

メール別名ファイル

別名を保守するには、以下のファイル、マップ、またはテーブルを使用します。

- 386 ページの「.mailrc の別名」
- 386 ページの「/etc/mail/aliases ファイル」
- 388 ページの「NIS aliases マップ」
- 388 ページの「NIS+ mail_aliases テーブル」

別名を保守する方法は、誰が使用し、誰が変更するかによって決まります。別名のタイプにはそれぞれ固有の形式要件があります。

関連する作業については、第 25 章 の 332 ページの「メール別名ファイルの管理 (作業マップ)」 を参照してください。

.mailrc の別名

.mailrc ファイルのリストに入っている別名には、ファイルを所有するユーザーしかアクセスできません。この制限により、ユーザーは自分で制御し、所有者だけが使用できる別名を作成できます。.mailrc ファイルの別名は、次のようになります。

alias *aliasname value value value* ...

aliasname は、ユーザーがメールの送信時に使用する名前であり、*value* は有効な電子メールアドレスです。

ユーザーが scott に個人的な別名を作成し、それがネームサービスの scott の電子メールアドレスと一致しない場合、そのユーザーが作成したメールにユーザーが返信しようとするときに、メールが間違ったユーザーに転送されることになります。これを回避するには、別の別名命名方式を使用する以外にありません。

/etc/mail/aliases ファイル

/etc/mail/aliases ファイルで作成したいずれの別名も、その別名の名前とファイルを含んでいるシステムのホスト名を知っているユーザーなら誰でも使用できます。ローカルの /etc/mail/aliases ファイルの配布リストは、以下のようになります。

aliasname: *value,value,value* ...

aliasname は、ユーザーがこの別名にメールを送信するときに使用する名前で、*value* は有効な電子メールアドレスになります。

ご使用のネットワークがネームサービスを実行していない場合は、各システムの /etc/mail/aliases ファイルにすべてのメールクライアントのエントリを入れておく必要があります。各システムのファイルを編集するか、1 つのシステムのファイルを編集してからそのファイルを他のシステムに個々にコピーします。

/etc/mail/aliases ファイルの別名は、テキスト形式で保存されます。/etc/mail/aliases ファイルを編集するときに、newaliases プログラムを実行してデータベースをコンパイルし直し、sendmail プログラムでその別名がバイナリ形式で使用できるようにします。作業手順については、第 25 章 の 340 ページの「ローカルメール別名ファイルを設定する方法」 を参照してください。 それ以外の場合、Solaris 管理コンソールの「メーリングリスト」機能を使ってローカルの /etc ファイルに保存されているメール別名を管理できます。

別名を作成できるのは、ローカル名、つまり現在のホスト名に対してだけ、またはホスト名は指定できません。 たとえば、システム saturn 上にメールボックスを持っているユーザー ignatz に対する別名エントリは、以下のエントリを /etc/mail/aliases ファイル内に持っています。

```
ignatz: ignatz@saturn
```

各メールサーバーに管理アカウントを作成する必要があります。管理アカウントを作成するには、メールサーバーのメールボックスを root に割り当て、root のエントリを /etc/mail/aliases ファイルに追加します。たとえば、システム saturn がメールボックスサーバーの場合は、エントリ root: sysadmin@saturn を /etc/mail/aliases ファイルに追加します。

通常は、root ユーザーだけがこのファイルを編集できます。ただし、Administration を使用する場合は、sysadmin グループであるグループ 14 のすべてのユーザーが、ローカルファイルを変更できます。または、以下のエントリを作成します。

aliasname: :include:*/path/aliasfile*

aliasname は、ユーザーがメールを送信するときに使用する名前であり、/path/aliasfile は別名リストを含むファイルへの完全パスになります。別名ファイルには、各行に 1 つの電子メールエントリを入れ、その他の表記は付けないでください。

user1@host1
user2@host2

/etc/mail/aliases に追加のメールファイルを定義して、ログやバックアップコピーの管理もできます。以下のエントリでは、*filename* の *aliasname* に送信されるすべてのメールを格納します。

aliasname: /home/backup/*filename*

また、メールを他のプロセスにルーティングすることもできます。次の例のように入力すると、メールメッセージのコピーが *filename* 内に格納され、コピーが出力されます。

aliasname: "|tee -a /home/backup/*filename* |lp"

作業マップについては、第 25 章 の 332 ページの「メール別名ファイルの管理 (作業マップ)」 を参照してください。

NIS aliases マップ

sendmail プログラムはローカルの /etc/mail/aliases ファイルの代わりに NIS aliases マップを使って送信アドレスを決定できるので、ローカルドメインのすべてのユーザーは NIS aliases マップのエントリを使用できます。詳細は、nsswitch.conf(4) のマニュアルページを参照してください。

NIS aliases マップの別名は、以下のようになります。

aliasname: *value,value,value* ...

aliasname は、ユーザーがメールを送信するときに使用する名前であり、*value* は有効な電子メールアドレスです。

NIS aliases マップには、すべてのメールクライアント用のエントリを含めてください。一般にこれらのエントリを変更できるのは、NIS マスターの root ユーザーだけです。この種の別名は頻繁に変更される場合には適していませんが、次の構文例のように、他の別名ファイルをポイントする場合には役立ちます。

aliasname: *aliasname@host*

aliasname はユーザーがメールを送信するときに使用する名前であり、*host* は /etc/mail/alias ファイルを含むサーバー用のホスト名です。

作業手順については、第 25 章 の 338 ページの「NIS mail.aliases マップを設定する方法」 を参照してください。

NIS+ mail_aliases テーブル

NIS+ mail_aliases テーブルには、名前が含まれており、それによってローカルドメインにおけるシステムや個人が登録されています。 sendmail プログラムは、ローカルの /etc/mail/aliases ファイルの代わりに NIS+ mail_aliases テーブルを使用して、メールアドレスを決定できます。詳細は、aliasadm(1M) と nsswitch.conf(4) のマニュアルページを参照してください。

NIS+ mail_aliases テーブルの別名は次のようになります。

alias: *expansion* # ["*options* " # "*comments*"]

表 26–12 に、NIS+ mail_aliases テーブルの 4 つの列を示します。

表 **26–12** NIS+ mail_aliases テーブルの列

列	説明
alias	別名の名前
expansion	sendmail の /etc/mail/aliases ファイルに現れる別名の値または別名のリスト

表 **26–12** NIS+ mail_aliases テーブルの列　(続き)

列	説明
options	今後の使用のために予約された列
comments	個々の別名のコメントのための列

NIS+ mail_aliases テーブルには、すべてのメールクライアントのエントリを含めてください。NIS+ aliases テーブルでは、aliasadm コマンドで、エントリの表示、作成、変更、および削除ができます。aliasadm コマンドを使用するには、aliases テーブルを所有する NIS+ グループのメンバーでなければなりません。作業手順については、第 25 章 の 333 ページの「NIS+ mail_aliases テーブルの別名エントリを管理する方法」 を参照してください。 Solaris 管理コンソールの「メーリングリスト」機能を使用して NIS+ メール別名を管理することもできます。

注 – 新規の NIS+ aliases テーブルを作成する場合は、エントリを作成する前にテーブルを初期設定する必要があります。テーブルが存在するときは、初期設定は不要です。

.forward ファイル

ホームディレクトリに .forward ファイルを作成すれば、sendmail およびその他のプログラムは、メールのリダイレクトや送信にこのファイルを使用できます。以下の節を参照してください。

- 389 ページの「回避すべき状況」
- 390 ページの「.forward ファイルの内容」
- 390 ページの「.forward.*hostname* ファイル」
- 390 ページの「.forward+*detail* ファイル」

作業マップについては、第 25 章 の 348 ページの「.forward ファイルの管理 (作業マップ)」 を参照してください。

回避すべき状況

以下に、容易に回避または修復できる状況を示します。

- メールが宛先のアドレスに配信されない場合は、ユーザーの .forward ファイルをチェックします。ユーザーが host1 のホームディレクトリに .forward ファイルを置いている場合があります。この場合、メールは user@host2 に転送されます。host2 にメールが着信すると、sendmail は NIS または NIS+ 別名に user があるかどうかを確認し、メッセージを user@host1 に返送します。これによってループが発生し、メールのバウンスが増加します。
- セキュリティの問題を予防するために、.forward ファイルは決して root または bin アカウントに入れないでください。必要な場合は、代わりに aliases ファイルを使ってメールを転送してください。

.forward ファイルの内容

メール配信で .forward ファイルを有効に使用するために、アクセス権などの以下の設定が正しく適用されていることを確認します。

- .forward ファイルへの書き込みは、ファイルの所有者に制限されます。この制限によって、他のユーザーに対するセキュリティを確保できます。
- ホームディレクトリのパスは root だけが所有し、root だけが書き込めるようにする必要があります。たとえば、.forward ファイルが /export/home/terry にある場合、 /export および /export/home は root が所有し、root だけが書き込めるようにします。
- また実際のホームディレクトリに書き込めるのは、そのユーザーだけでなければなりません。
- .forward ファイルをシンボリックリンクにすることはできません。また、複数のハードリンクを持つこともできません。

.forward.*hostname* ファイル

.forward.*hostname* ファイルを作成すれば、特定のホストに送信されるメールをリダイレクトできます。 たとえば、ユーザーの別名が sandy@phoenix.example.com から sandy@example.com に変更された場合は、sandy のホームディレクトリに .forward.phoenix ファイルを置きます。

```
% cat .forward.phoenix
sandy@example.com
"|/usr/bin/vacation sandy"
% cat .vacation.msg
From: sandy@example.com (via the vacation program)
Subject: my alias has changed

My alias has changed to sandy@example.com.
Please use this alias in the future.
The mail that I just received from you
has been forwarded to my new address.

Sandy
```

この例では、メールが正しい宛先に転送され、送信者には別名の変更が通知されます。vacation プログラムではメッセージファイルは 1 つしか使用できないため、この場合 1 回につき 1 つのメッセージしか実行できません。ただし、メッセージが特定のホストに限定されない場合、.forward ファイルで複数のホストに同じ休暇メッセージファイルを使用できます。

.forward+*detail* ファイル

転送メカニズムの拡張機能にはこの他に、.forward+*detail* ファイルがあります。*detail* 文字列には、演算子文字を除く任意の文字を使用できます。演算子文字とは、.:%&!^[]+ です。この種のファイルを使用すれば、他のユーザーが電子メールアドレスを無断で使用しているかどうかを確認できます。たとえば、あるユーザーが、誰

かに電子メールアドレス sandy+test1@example.com を使用するように指示した場合、ユーザーは、この別名に配信されるメールを、アドレスに送信されるメールの中から識別できます。デフォルトにより、sandy+test1@example.com の別名に送信されたメールはすべて、この別名と .forward+*detail* ファイルと突き合わせて検査されます。ここで一致しない場合は、そのメールは最終的に sandy@example.com に配信されますが、ユーザーは、これらのメールの To: ヘッダー内の変更箇所を調べることができます。

/etc/default/sendmail ファイル

このファイルは sendmail のための初期設定用オプションを保存し、ホストをアップグレードしたときに除去されないようにするために使用します。次の変数を使用することができます。

CLIENTOPTIONS="*string*"
: クライアントデーモンで使用する追加オプションを選択します。クライアントデーモンは、クライアントだけのキュー (/var/spool/clientmqueue) の内容を確認し、クライアントキューランナーとして動作します。構文の確認は行われないため、この変数を変更するときは間違えないように注意してください。

CLIENTQUEUEINTERVAL=#
: CLIENTQUEUEINTERVAL には、QUEUEINTERVAL オプションと同様に、メールキューの実行間隔を設定します。ただし、CLIENTQUEUEINTERVAL オプションは、マスターデーモンではなくクライアントデーモンを制御します。一般に、マスターデーモンはすべてのメッセージを SMTP ポートに配信できます。ただし、メッセージ負荷が高すぎる場合、またはマスターデーモンが実行されていない場合、メッセージはクライアントだけのキューである /var/spool/clientmqueue に入ります。次に、クライアントだけのキューをチェックするクライアントデーモンがクライアントキューを処理します。

ETRN_HOSTS="*string*"
: SMTP クライアントとサーバーが、定期的なキューの実行を待たずに即座に対話を実行できるようにします。サーバーは、指定されたホストに送信されるキューを即座に配信できます。詳細は、etrn(1M) のマニュアルページを参照してください。

MODE=-bd
: sendmail を起動するためのモードを選択します。-bd オプションを使用するか、未定義のままにしておきます。

OPTIONS=*string*
: マスターデーモンで使用される追加オプションを選択します。構文の確認は行われないため、この変数を変更するときは間違えないように注意してください。

QUEUEINTERVAL=#
: マスターデーモンのメールキューの実行間隔を設定します。# は正の整数とし、その後に秒の場合は s、分の場合は m、時の場合は h、日の場合は d、週の場合は w を付けます。この構文は sendmail の起動前に確認されます。この間隔が負の場合、またはエントリの最後の文字が不適当な場合、この間隔は無視され、sendmail は 15 分のキュー間隔で起動します。

QUEUEOPTIONS=p
キューを実行するたびに新しいキューランナーを作成する代わりに、各実行の間に休止する単一の永続的なキューランナーを使用できるようにします。このオプションに設定可能な値は p だけです。p 以外に設定すると、このオプションは無効になります。

メールアドレスとメールルーティング

配信時にメールメッセージが辿る経路は、クライアントシステムの設定とメールドメインのトポロジによって異なります。メールホストやメールドメインの各追加レベルでは、別名のもう 1 つの解釈を追加できますが、ルーティングプロセスは基本的にほとんどのホストで同じになります。

クライアントシステムは、メールをローカルに受信できるようにセットアップできます。メールをローカルで受信することは、ローカルモードでの `sendmail` の実行として知られています。すべてのメールサーバーと一部のクライアントでは、ローカルモードがデフォルトです。ローカルモードのメールサーバーまたはクライアントでは、メッセージは以下の要領でルーティングされます。

注 – 次の例では、`sendmail.cf` ファイルに設定されたデフォルトの規則を使用することを前提にしています。

---

1. 可能な場合はメール別名を展開し、ローカルのルーティングプロセスを再起動します。

   ネームサービスでメール別名を確認し、見つかった場合に新しい値と置換することで、メールアドレスが展開されます。次にこの新しい別名が再度確認されます。

2. メールがローカルの場合、`/usr/lib/mail.local` に配信されます。

   メールはローカルのメールボックスに配信されます。

3. メールアドレスがこのメールドメインにホストを含んでいると、そのホストにメールを配信します。

4. アドレスがこのドメインにホストを含んでいない場合、メールホストにメールを転送します。

   メールホストはメールサーバーと同じルーティングプロセスを使用しますが、メールホストはホスト名に加えて、ドメイン名が宛先になっているメールも受信できます。

sendmail とネームサービスの相互作用

ここでは、sendmail とネームサービスに適用されるドメイン名、ネームサービスを有効に利用するための規則、および sendmail とネームサービスの対話について説明します。詳細は、以下のトピックを参照してください。

関連する作業については、第 25 章 の 328 ページの「sendmail で DNS を使用する方法」 または 332 ページの「メール別名ファイルの管理 (作業マップ)」 を参照してください。

sendmail.cf とメールドメイン

標準の sendmail.cf ファイルは、メールドメインを使ってメールを直接配信するか、あるいはメールホストを経由して配信するかを決定します。ドメイン内メールは直接 SMTP 接続経由で配信され、ドメイン間メールはメールホストに送られます。

セキュリティの高いネットワークでは、ほんの少数の選ばれたホストだけが、外部宛てのパケットを生成する権限を与えられています。ホストがメールドメインの外部のリモートホストの IP アドレスを持っている場合も、SMTP 接続の確立は保証されません。標準の sendmail.cf では次のことを仮定しています。

- 現在のホストは、パケットを直接メールドメイン外のホストに送信する権限がない
- メールホストは、パケットを外部ホストに直接送信できる認可されたホストにメールを転送できます。実際には、メールホスト自体が認可されたホストになることがあります。

このように仮定すると、ドメイン間メールの配信または転送はメールホスト側の責任です。

sendmail とネームサービス

sendmail は各種の要件をネームサービスに課します。これらの必要条件の理解を深めるために、ここでは、まずメールドメインからネームサービスドメインへの関係について説明し、次に個々の要件について説明します。 以下を参照してください。

- 394 ページの「ホストネームサービスデータ」
- in.named(1M)、nis+(1)、nisaddent(1M)、および nsswitch.conf(4) のマニュアルページ

メールドメインとネームサービスドメイン

メールドメイン名はネームサービスドメイン名の接尾辞の 1 つでなければなりません。たとえば、ネームサービスのドメイン名が「A.B.C.D」ならば、メールドメイン名は次のうちのいずれかです。

- A.B.C.D
- B.C.D
- C.D
- D

メールドメイン名は、最初に設定されたときには、多くの場合ネームサービスドメインと同じになります。ネットワークが大きくなれば、ネームサービスドメインを小さく分割してネームサービスを管理しやすくすることができます。ただし、メールドメインは、一貫した別名を提供するために分割されないまま残ることがあります。

ホストネームサービスデータ

ここでは、sendmail がネームサービスに必要とする要件について説明します。

ネームサービスにおけるホストテーブルまたはマップは、次の 3 種類の gethostbyname() による問い合わせをサポートするように設定しなければなりません。

- mailhost – いくつかのネームサービスの構成では、自動的にこの要件を満たします。
- 完全なホスト名 (たとえば、smith.admin.acme.com) – 多くのネームサービスの構成がこの要件を満たします。
- 短いホスト名 (たとえば、smith) – sendmail は、外部メールを転送するためにメールホストに接続する必要があります。メールアドレスが現在のメールドメイン内であるかどうかを判定するために、gethostbyname() が完全なホスト名で呼び出されます。エントリが見つかると、アドレスは内部にあるとみなされます。

  NIS、NIS+、および DNS は、短いホスト名を引数にする gethostbyname() をサポートします。したがって、この要件は自動的に満たされます。

ネームサービス内に有効な sendmail サービスを確立するために、ホストネームサービスに追加された以下の 2 つの規則に従う必要があります。

- 完全なホスト名と短いホスト名の引数を持った gethostbyname() は、同一の結果を生成する必要があります。たとえば、両関数がメールドメイン admin.acme.com から呼び出されるかぎり、gethostbyname (smith.admin.acme.com) と gethostbyname (smith) は同じ結果になるよ

うにします。

- 共通のメールドメイン下のすべてのネームサービスドメインに対しては、短いホスト名による gethostbyname() で同じ結果を生じるようにします。たとえば、ebb.admin.acme.com ドメインおよび esg.admin.acme.com ドメインから smith.admin.acme.com メールドメインを呼び出した場合、どちらの場合も gethostbyname(smith) は同じ結果を返す必要があります。主なメールドメイン名は通常ネームサービスドメインより短く、このために各種ネームサービスにとって特別な意味のあるものになっています。

gethostbyname() 関数については、gethostbyname(3NSL) のマニュアルページを参照してください。

sendmail と NIS との相互作用

以下に、sendmail と NIS との相互作用について説明し、ガイドラインを示します。

- メールドメイン名 – NIS をプライマリネームサービスとして設定している場合に、sendmail は、自動的に NIS ドメイン名の最初の構成要素を取り除いた結果をメールドメイン名として使用します。 たとえば、ebs.admin.acme.com は、admin.acme.com となります。
- メールホスト名 – NIS のホストマップには、mailhost エントリが必要になります。
- 完全なホスト名 – 通常の NIS の設定では、完全なホスト名は認識されません。NIS に完全なホスト名を認識させようとするよりは、sendmail.cf ファイルを編集し %l を %y で置き換えて、sendmail 側からこの要件をなくしてください。この変更によって、sendmail のドメイン間のメール検出機能をオフにできます。ターゲットとするホストの IP アドレスを取得できれば、SMTP による直接配信が試みられます。NIS のホストマップに現在のメールドメインの外部のホストのエントリが含まれていないことを確認してください。もし、そのエントリがあれば、さらに sendmail.cf ファイルをカスタマイズする必要があります。
- ホストの完全名および短縮名のマッチング – 前述の手順を参考にして、完全なホスト名による gethostbyname() をオフにしてください。
- 1 つのメールドメイン内の複数の NIS ドメイン – 共通のメールドメインの NIS のホストマップ中のホストのエントリは同じでなければなりません。たとえば、ebs.admin.acme.com ドメインのホストマップは、esg.admin.acme.com のホストマップと同じものにします。異なる場合には、ある NIS ドメインで有効なアドレスが他の NIS ドメインでは無効になることがあります。

作業手順については、第 25 章 の 332 ページの「メール別名ファイルの管理 (作業マップ)」 を参照してください。

sendmail と NIS および DNS との相互作用

以下に、sendmail と NIS および DNS との相互作用について説明し、ガイドラインを示します。

- メールドメイン名 – NIS をプライマリネームサービスとして設定している場合に、sendmail は、自動的に NIS ドメイン名の最初の構成要素を取り除いた結果をメールドメイン名として使用します。 たとえば、ebs.admin.acme.com は、admin.acme.com となります。
- メールホスト名 – DNS の転送機能がオンになっていれば、NIS で解決できない照会は DNS に転送されるため、NIS ホストマップに mailhost エントリは必要ありません。
- 完全なホスト名 – NIS が完全なホスト名を認識できなくても、DNS が認識します。NIS と DNS の通常の設定手順を踏んでいる場合には、完全なホスト名の要件は満たされます。
- ホストの完全名および短縮名のマッチング – NIS のホストテーブルにおけるすべてのホストエントリに対して、DNS にも対応するホストエントリが必要です。
- 1 つのメールドメイン内の複数の NIS ドメイン – 共通のメールドメインの NIS のホストマップ中のホストのエントリは同じでなければなりません。たとえば、ebs.admin.acme.com ドメインのホストマップは、esg.admin.acme.com のホストマップと同じものにします。異なる場合には、ある NIS ドメインで有効なアドレスが他の NIS ドメインでは無効になることがあります。

作業手順については、第 25 章 の 328 ページの「sendmail で DNS を使用する方法」 と 332 ページの「メール別名ファイルの管理 (作業マップ)」 を参照してください。

sendmail と NIS+ との相互作用

以下に、sendmail と NIS+ との相互作用について説明し、ガイドラインを示します。

- メールドメイン名 – プライマリネームサービスとして、NIS+ を設定していれば、sendmail は、NIS+ の sendmailvars テーブル (キーと値から構成される 2 列の NIS+ テーブル) からメールドメインを確認します。メールドメインを設定するには、 1 つのエントリをこのテーブルに追加する必要があります。このエントリは、キーの列に文字列 maildomain が、値の列には自分のドメイン名 (たとえば、admin.acme.com) が設定されている必要があります。NIS+ では、sendmailvars テーブルにどのような文字列でも設定できますが、メールシステムが正常に機能するように接尾辞の規則が適用されます。nistbladm を使用して、maildomain エントリを sendmailvars テーブルに追加できます。以下の例では、メールドメインが NIS+ ドメインの接尾辞になっています。

```
nistbladm -A key="maildomain" value=<mail domain> sendmailvars.org_dir.<NIS+ domain>
```

- メールホスト名 – NIS+ ホスト名には、mailhost エントリが必要です。
- 完全なホスト名 – NIS+ は、完全なホスト名を認識することができます。 通常の NIS+ の設定手順を行えば、この完全なホスト名の要件は満たされます。
- ホストの完全名および短縮名のマッチング – この要件を満たすには、すべてのホストテーブルでエントリをコピーするか、ユーザーネームサービスのドメイン中の全ホストのエントリをメールドメインレベルのマスターホストテーブルに入力する必

要があります。

- 1 つのメールドメイン内の複数の NIS ドメイン – この項目を満たすには、すべてのホストテーブルのエントリをコピーするか、ユーザーネームサービスのドメイン中の全ホストのエントリをメールドメインレベルのマスターホストテーブルに入力する必要があります。実際には、論理的または物理的に複数のホストテーブルを 1 つのホストテーブルにマージしています。したがって、メールドメインを共有する複数のネームサービスドメインで同じホスト名を使用することはできません。

作業手順については、第 25 章 の 332 ページの「メール別名ファイルの管理 (作業マップ)」 を参照してください。

sendmail と NIS+ および DNS との相互作用

以下に、sendmail と NIS+ および DNS との相互作用について説明し、ガイドラインを示します。

- メールドメイン名 – プライマリネームサービスとして、NIS+ を設定していれば、sendmail は、NIS+ の sendmailvars テーブル (キーと値から構成される 2 列の NIS+ テーブル) からメールドメインを確認します。メールドメインを設定するには、 1 つのエントリをこのテーブルに追加する必要があります。このエントリは、キーの列に文字列 maildomain が、値の列に自分のドメイン名 (たとえば、admin.acme.com) が設定されている必要があります。NIS+ では、sendmailvars テーブルにどのような文字列でも設定できますが、メールシステムが正常に機能するように接尾辞の規則が適用されます。nistbladm を使用して、maildomail エントリを sendmailvars テーブルに追加できます。以下の例では、メールドメインが NIS+ ドメインの接尾辞になっています。

```
nistbladm -A key="maildomain" value=<mail domain> sendmailvars.org_dir.<NIS+ domain>
```

- メールホスト名 – ネットワークがホストデータベースのソースとして NIS+ と DNS の両方を使用しているときは、mailhost エントリを NIS+ あるいは DNS ホストテーブルのいずれかに置くことができます。NIS+ と DNS をホストデータベースのソースとして /etc/nsswitch.conf ファイルに含めるようにしてください。
- 完全なホスト名 – NIS+ も DNS も完全なホスト名を認識します。通常の NIS+ と DNS の設定手順を踏めば、この項目の要件は満たされます。
- ホストの完全名および短縮名のマッチング – NIS+ ホストテーブルの全ホストエントリに対して、対応するエントリが DNS に必要です。
- 1 つのメールドメイン内の複数の NIS ドメイン – この要件を満たすには、全ホストテーブルエントリをコピーするか、ネームサービスのドメイン中の全ホストのエントリをメールドメインレベルのマスターホストテーブルに入力する必要があります。

作業手順については、第 25 章 の 332 ページの「メール別名ファイルの管理 (作業マップ)」 と 328 ページの「sendmail で DNS を使用する方法」 を参照してください。

第 27 章

メールサービスの新機能 (リファレンス)

第 24 章では、メールサービスのコンポーネントの概要および一般的なメール構成について説明しています。第 25 章では、標準の構成ファイルを使用して、電子メールシステムを設定および管理する方法について説明しています。第 26 章では、メールサービスのコンポーネントについて、詳しく説明しています。 また、メールサービスのプログラムとファイル、メールルーティング処理、ネームサービスを使った sendmail の対話式操作についても説明しています。この章では、今回の Solaris 9 リリースに付属している sendmail バージョン 8.12 の新機能について説明します。mail.local、mailstats、および makemap の変更点についての説明もあります。また、この章では、新しい保守ユーティティ editmap についても説明します。 特定のトピックについては、下記のページを参照してください。

この章で扱っていないトピックについては、sendmail(1M)、mail.local(1M)、mailstats(1)、makemap(1M)、および editmap(1M) のマニュアルページを参照してください。

sendmail の変更点

この章では、次のトピックについて説明します。

- 417 ページの「sendmail に新しく定義されたマクロ」
- 420 ページの「sendmail における新しい m4 構成マクロおよび改訂された m4 構成マクロ」
- 424 ページの「配信エージェントの新しいフラグ」
- 424 ページの「配信エージェントの新しい等号 (=)」
- 425 ページの「キューの新しい機能」
- 426 ページの「sendmail における LDAP のための変更」
- 427 ページの「メールプログラムに新しく組み込まれた機能」
- 428 ページの「新しいルールセット」
- 429 ページの「ファイルへの変更」
- 429 ページの「構成内の IPv6 アドレス」

新しい構成ファイル submit.cf

sendmail バージョン 8.12 には、新しい構成ファイル /etc/mail/submit.cf が含まれています。この新しいファイル submit.cf を使用して、sendmail をデーモンモードではなく、メール差し出しプログラムモードで実行できます。 デーモンモードとは異なり、メール差し出しプログラムモードでは root 権限は必要ありません。そのため、この新しいパラダイムを使用すると、セキュリティが向上します。

submit.cf の機能については、次のリストを参照してください。

- sendmail は、MSP (メール差し出しプログラム) モードでは submit.cf を使って実行します。submit.cf は、電子メールを送信したり、ユーザー以外の mailx のようなプログラムによって呼び出したりすることができます。 -Ac オプションおよび -Am オプションについては、402 ページの「コマンド行の新しいオプションまたは推奨されないオプション」 の説明を参照してください。
- submit.cf は、次の操作モードで使用します。

  -bm デフォルトの操作モード
  -bs 標準入力を使用して SMTP を実行する
  -bt アドレスの解決に使用されるテストモード

- submit.cf を使用している場合には、sendmail は SMTP デーモンとして動作しません。
- submit.cf を使用している場合には、sendmail はクライアント専用のメールキューである /var/spool/clientmqueue を使用します。このキューにより、sendmail デーモンに配信されなかったメッセージが保持されます。 クライアント専用キューにあるメッセージは、クライアントの「デーモン」によって配信されます。実際には、このデーモンが、クライアントキューを実行します。
- デフォルトでは、sendmail は submit.cf を使用して、定期的に MSP キュー (クライアント専用キュー) である /var/spool/clientmqueue を実行します。

```
/usr/lib/sendmail -Ac -q15m
```

次の事項に注意してください。

- Solaris 9 オペレーティング環境をインストールまたはそれにアップグレードすると、submit.cf は自動的にインストールされます。

- Solaris 9 オペレーティング環境をインストールする前に、submit.cf について計画および準備をする必要はありません。
- 構成ファイルを指定しないかぎり、sendmail は、必要に応じて、submit.cf を自動的に使用します。 基本的に、sendmail は各タスクについて、submit.cf と sendmail.cf のどちらを使用するのが適切かを判断できます。
- submit.cf を変更することはできません。

sendmail.cf と submit.cf の機能の相違点

構成ファイル sendmail.cf は、デーモンモードで使用します。このファイルを使用すると、sendmail は、メール転送エージェント (MTA) として動作します。sendmail は、root によって起動されます。

```
/usr/lib/sendmail -L sm-mta -bd -q1h
```

sendmail.cf 特有の他の機能については、次のリストを参照してください。

- デフォルトでは、sendmail.cf は、ポート 25 および 587 で SMTP 接続を受け入れます。
- デフォルトでは、sendmail.cf がメインキュー /var/spool/mqueue を実行します。

sendmail の機能の変更

submit.cf が追加されたため、次の機能が変更されました。

- sendmail バージョン 8.12 では、root だけがメインキューを実行できます。 この変更の詳細については、mailq(1) のマニュアルページを参照してください。新しい作業手順については、344 ページの「キューディレクトリの管理 (作業マップ)」 を参照してください。
- メール通信プログラムモードは、root 権限なしに実行されるため、sendmail が .forward などの特定のファイルにアクセスできないことがあります。この場合、sendmail に -bv オプションを追加すると、誤った項目が出力されることがあります。回避策はありません。
- 8.12 より前のバージョンの sendmail では、sendmail デーモンを実行しない場合、つまりデーモンモードで実行しない場合は、受信メールの配信を防止することしかできませんでした。バージョン 8.12 では、デフォルトの構成で、sendmail デーモンを実行しない場合でも、送信メールの配信を防止することができます。クライアントキューランナー (メール通信プログラム) を設定して、ローカル SMTP ポートのデーモンにメールを送信できるようにする必要があります。クライアントキューランナーが SMTP のセッションをローカルホストで開こうとした場合で、デーモンが SMTP ポートで待機していないときには、メールはキューにとどまります。デフォルトの構成では、デーモンが実行されます。そのため、デフォルト構成を使用する場合には、この問題は発生しません。ただし、デーモンを無効にした場合の解決方法については、331 ページの「代替構成を使用したメール配信の管理

(手順)」 を参照してください。

コマンド行の新しいオプションまたは推奨されないオプション

次の表では、sendmail の新しいコマンド行オプションについて説明します。 コマンド行の他のオプションについては、sendmail(1M) のマニュアルページを参照してください。

表 **27–1** sendmail のコマンド行の新しいオプション

オプション	説明
-Ac	オペレーションモードが新しいメールの差し出し依頼を示していない場合でも、構成ファイル submit.cf を使用する。 submit.cf の詳細については、400 ページの「新しい構成ファイル submit.cf」 を参照
-Am	オペレーションモードが新しいメールの差し出し依頼を示している場合でも、構成ファイル sendmail.cf を使用する。 詳細は、400 ページの「新しい構成ファイル submit.cf」 を参照
-bP	各キューのエントリ数を出力する
-G	コマンド行から送信したメッセージが、新たな差し出しを依頼するものではなく、リレーするものであることを示す。 アドレスが絶対パスではない場合は、メッセージは拒否される。正規化は実行されない。 ftp://ftp.sendmail.org で sendmail とともに配布しているリリースノートで説明しているように、将来のリリースでは、不適切な形式のメッセージを拒否される可能性がある
-L *tag*	指定された syslog メッセージに使用する識別子を *タグ (tag)* に設定する
-q[!]I *substring*	受信者にこの部分文字列 *(substring)* を含むジョブだけを処理する。 オプションに ! を追加すると、受信者にこの部分文字列 *(substring)* を含まないジョブだけを処理する
-q[!]R *substring*	キュー ID にこの部分文字列 *(substring)* を含むジョブだけを処理する。 オプションに ! を追加すると、送信者にこの部分文字列 *(substring)* を含まないジョブだけを処理する
-q[!]S *substring*	送信者にこの部分文字列 *(substring)* を含むジョブだけを処理する。 オプションに ! を追加すると、送信者にこの部分文字列 *(substring)* を含まないジョブだけを処理する
-qf	キューにあるメッセージをシステムコール fork を使用しないで一度処理し、フォアグラウンドで処理を実行する。 fork(2) のマニュアルページを参照
-qG*name*	キューグループ「*name (名前)*」にあるメッセージだけを処理する
-qp*time*	各キュー用にフォークされた子を使用して、キューに保存されているメッセージを指定した間隔で処理する。 次にキューが実行されるまでの間、この子は動作しないこの新しいオプションは -q*time* に似ています。 -q*time* は、定期的に子をフォークしてキューを処理する

表 **27–1** sendmail のコマンド行の新しいオプション　(続き)

オプション	説明
-U	ftp://ftp.sendmail.org で sendmail とともに配布しているリリースノートで説明しているように、このオプションは、バージョン 8.12 では使用できない。メールユーザーエージェントでは、引数 -G を使用する必要がある

構成ファイルの新しい構成オプションと改訂された構成オプション、および関連トピック

この節では、構成ファイルの新しいオプションと改訂されたオプションについて、また次の関連トピックについて説明しています。

- 411 ページの「sendmail の構成ファイルにおける推奨されないオプションまたはサポートされていないオプション」
- 412 ページの「新しい ClientPortOptions オプション」
- 413 ページの「DaemonPortOptions オプションの変更点」
- 415 ページの「PidFile オプションおよび ProcessTitlePrefix オプションのその他の引数」
- 415 ページの「PrivacyOptions オプションの変更点」
- 416 ページの「Timeout オプションの変更点」

これらのオプションを宣言する場合は、次の構文のどれかを使用します。

```
O OptionName=argument              # 構成ファイル
-OOptionName=argument              # コマンド行
define(`m4Name',argument)          # m4 を使った構成記述
```

新しい sendmail.cf ファイルを構築する必要がある場合は、第 25 章 の 329 ページの「sendmail.cf 構成ファイルの構築 (手順) 」 を参照してください。

次の表では、sendmail の新しいオプションおよび改訂されたオプションについて説明しています。

表 **27–2** sendmail の新しいオプションおよび改訂されたオプション

オプション	説明
BadRcptThrottle	m4 名: confBAD_RCPT_THROTTLE 引数 : 数値 この新しいオプションを使用して、受信者のしきい値が拒否された後、SMTP エンベロープ内の受信者を承認する率を制限する
ClientPortOption	詳細については、412 ページの「新しい ClientPortOptions オプション」 を参照

表 **27–2** sendmail の新しいオプションおよび改訂されたオプション　*(続き)*

オプション	説明
ConnectionRateThrottle	m4 名: confCONNECTION_RATE_THROTTLE 引数 : 数値 ConnectionRateThrottle オプションを使用して、各デーモンへの合計接続数ではなく、1 秒あたりの接続数を制限する
ControlSocketName	m4 名: confCONTROL_SOCKET_NAME 引数 : ファイル名推奨ソケット名は /var/spool/mqueue/.smcontrol。セキュリティを向上させるために、この UNIX ドメインソケットを、root だけがアクセスできるディレクトリに置く このオプションを設定すると、デーモン管理用の制御ソケットが作成される。このオプションを使用して、指定した名前のソケットを使って実行した sendmail デーモンの状態を、外部のプログラムから制御したり照会したりすることができる。このソケットは、INN ニュースサーバーに対する ctlinnd インタフェースに似ている。このオプションを設定しないと、制御ソケットは使用できない
DaemonPortOptions	詳細については、413 ページの「DaemonPortOptions オプションの変更点」 を参照
DataFileBufferSize	m4 名: confDF_BUFFER_SIZE 引数 : 数値 この新しいオプションを指定すると、ディスクベースのファイルを使用する前にメモリーに蓄積できるデータ (df) ファイルの最大サイズを、バイトで制御できる。 デフォルトは 4096 バイト。Solaris オペレーティング環境のデフォルトを変更する必要はない
DeadLetterDrop	m4 名: confDEAD_LETTER_DROP 引数 : ファイル名 この新しいオプションを使用して、システム全体のファイル dead.letter の場所を定義する。前のバージョンでは、このファイルは /usr/tmp/dead.letter に固定されていた。このオプションを設定する必要はない
DelayLA	m4 名: confDELAY_LA 引数 : 数値 この新しいオプションに 0 より大きな値を設定すると、次のように動作する 平均負荷率が指定値を超えると、接続を 1 秒ずつ遅らせる ほとんどの SMTP コマンドを 1 秒ずつ遅れて実行する デフォルト値が 0 であるため、このオプションを設定しないと sendmail の動作は変更されない

表 **27–2** sendmail の新しいオプションおよび改訂されたオプション　(続き)

オプション	説明
DeliverByMin	m4 名: confDELIVER_BY_MIN 引数 : 時間 この新しいオプションを使用して、クライアントは、FC 2852 の「Deliver By SMTP Service Extension」で指定されているように、電子メールメッセージを配信する際の最短時間を指定できる time を 0 に設定すると、時間は表示されない time を 0 より小さくすると、拡張は利用できない time を 0 より大きくすると、その拡張した時間が EHLO のキーワードである DELIVERBY の最短時間として表示される
DirectSubmissionModifiers	m4 名: confDIRECT_SUBMISSION_MODIFIERS 引数 : 修飾子 この新しいオプションにより、直接の実行依頼 (コマンド行) の ${daemon_flags} を定義する。このオプションを設定しない場合、${daemon_flags} の値はオプション -G を使用すると CC f に、使用しないと c u になる
DontBlameSendmail	新たに、次の引数を使用できる 引数 NonRootSafeAddr が追加された。sendmail に .forward プログラムを実行したり、所有者としてそのファイルに配信する権限がない場合には、アドレスは「安全ではない」とマークされる。 また、RunAsUser を設定すると、プログラムを使用したり、.forward プログラムのファイルに配信したりすることができない。 これらの問題を解決するには、NonRootSafeAddr を使用する
DoubleBounceAddress	m4 名: confDOUBLE_BOUNCE_ADDRESS 引数 : アドレス。デフォルトは、postmaster sendmail がエラーメッセージを送信する際にエラーが発生した場合には、sendmail は、このオプションの引数で指定したアドレスに、「double-bounced」エラーメッセージを送信する
FallBackMXhost	m4 名: confFALLBACK_MX 引数 : 完全指定ドメイン名 このオプションを使用して、MX レコードを参照することができる。MX レコードを参照しない前のバージョンのオプションを使用するには、名前を角括弧で括って指定する

表 **27–2** sendmail の新しいオプションおよび改訂されたオプション　*(続き)*

オプション	説明
FastSplit	m4 名: confFAST_SPLIT
	引数 : 数値デフォルト値は 1
	この新しいオプションを指定すると、次の動作を実行する
	このオプションを 0 より大きな値に設定する場合、アドレスをソートする場合、最初の MX 参照を行わない。そのため、エンベロープをより高速に分割できる
	メールをコマンド行から送信する場合は、この値により、エンベロープを配信するのに使用するプロセスの数を制限できる
	さらに多くのエンベロープが作成されると、それらはいったんキューに置かれ、キューが実行されると解釈処理される
LDAPDefaultSpec	m4 名: confLDAP_DEFAULT_SPEC
	引数 : 適切に定義されたクラス指定。たとえば、-h*host*、-p*port*、-d*bind DN* など
	この新しいオプションを使用して、LDAP マップのデフォルトのマップ仕様を指定できる。 K コマンドを使って個別のマップ仕様を作成しないかぎり、ここで行われたデフォルトの設定が、すべての LDAP マップに使用される。このオプションを設定してから、LDAP マップを定義する
MailboxDatabase	m4 名: confMAILBOX_DATABASE
	引数 : デフォルト値は pw。これは、getpwnam() を使用する。他の値を使用することはできない
	この新しいオプションにより、ローカルな受信者を確認するのに使用されるメールボックスデータベースのタイプを指定できる
MaxHeadersLength	m4 名: confMAX_HEADERS_LENGTH
	引数 : 数値
	このオプションにより、全ヘッダーの合計した最大の長さを指定できる。また、このオプションを使用して、サービス拒否害攻撃を防止できる。 デフォルト値は 32768。 16384 より小さい値を使用すると、警告が発行される。 Solaris オペレーティング環境のデフォルト値を変更する必要はない

表 **27–2** sendmail の新しいオプションおよび改訂されたオプション　(続き)

オプション	説明
MaxMimeHeaderLength	m4 名: confMAX_MIME_HEADER_LENGTH 引数 : 数値 このオプションにより、特定の MIME ヘッダ ーフィールド値の最大の長さを、文字数 *x* に設定できる。また、ヘッダー内のパラメータについては、*y* の最大の長さを指定できる。値を結合すると、「*x/y*」のようになる。*y* を指定しない場合には、*x* の半分の値が使用される。 デフォルト値は 0 であるため、値を指定しないと、確認が実行されない。 このオプションは、メールユーザーエージェントをバッファーオーバーフロー攻撃から保護する目的で提供されている。推奨値の範囲は、256/128 から 1024/256。128/40 より小さい値を使用すると、警告が発行される。
MaxQueueChildren	m4 名: confMAX_QUEUE_CHILDREN 引数 : 数値 この新しいコマンドにより、同時にアクティブになるキューランナー処理の数を引数で指定した値に制限できる。このオプションを使用すると、キューの処理時に使用されるシステム資源を制限できる。複数のキューグループにおけるキューランナーの合計数が定義した引数を超えると、残りのキューグループは、後で実行される
MaxRecipientsPerMessage	m4 名: confMAX_RCPTS_PER_MESSAGE 引数 : 数値 このオプションを設定すると、SMTP エンベロープ内の受信者が指定した数を超えないようにする。最小の引数は 100。このオプションは、コマンド行からも、構成ファイルからも宣言できる。ただし、通常のユーザーは、コマンド行からこのオプションを設定し、sendmail -bs を使って送信したメッセージの上書きを有効にすることができる。この場合でも、sendmail は、その root 権限を放棄しない
MaxRunnersPerQueue	m4 名: confMAX_RUNNERS_PER_QUEUE 引数 : 数値デフォルト値は 1。リソースについてよく考慮し、この値を高く設定しないように注意する この新しいオプションにより、1 キューグループあたりのキューランナーの最大数を指定できる。複数のキューランナーは、キューグループのメッセージを並行処理する。この機能は、前のメッセージの処理が原因で次のメッセージの処理が遅れる可能性がある場合に便利である
NiceQueueRun	m4 名: confNICE_QUEUE_RUN 引数 : 数値 この新しいオプションにより、キューランナーの優先順位を設定できる。nice(1) のマニュアルページを参照

表 **27–2** sendmail の新しいオプションおよび改訂されたオプション　*(続き)*

オプション	説明
PidFile	m4 名: confPID_file 引数 : 415 ページの「PidFile オプションおよび ProcessTitlePrefix オプションのその他の引数」 を参照 この新しいオプションにより、pid ファイルの場所を定義できる。 ファイルを開く前に、そのファイル名がマクロで展開される。デフォルトの位置は /var/run/sendmail.pid
PrivacyOptions	詳細については、415 ページの「PrivacyOptions オプションの変更点」 を参照
ProcessTitlePrefix	m4 名: confPROCESS_TITLE_PREFIX 引数 : 415 ページの「PidFile オプションおよび ProcessTitlePrefix オプションのその他の引数」 を参照 この新しいオプションにより、/usr/ucb/ps auxww にリストされるプロセスのタイトルについて、接頭辞の列を指定できる。この文字列はマクロで処理される。Solaris オペレーティング環境のデフォルト値を変更する必要はない
QueueFileMode	m4 名: confQUEUE_FILE_MODE 引数 : 数値 この新しいオプションを使用すると、キューファイルのデフォルトアクセス権を 8 進数で指定できる。このオプションを設定しないと、sendmail は 0600 を使用する。ただし、オプションの実ユーザー ID と実行ユーザー ID が異なる場合には、sendmail は 0644 を使用する
QueueLA	m4 名: confQUEUE_LA 引数 : 数値 デフォルト値は、8 からシステム起動時にオンラインであるプロセッサ数の 8 倍に変更された。 単一プロセッサマシンでは、このデフォルト値の変更による影響はない。 この値を変更するとデフォルト値が無効になり、プロセッサ数を考慮しなくなる。 そのため、値を変更することによる影響について、よく理解する必要がある

表 **27–2** sendmail の新しいオプションおよび改訂されたオプション　(続き)

オプション	説明
QueueSortOrder	m4 名: confQUEUE_SORT_ORDER このオプションにより、キューのソートに使用するアルゴリズムを設定する。デフォルト値は priority であり、キューをメッセージの優先順位でソートする。 次の変更に注意してください。 host の引数は、ホスト名を逆にしてからソートを実行する。つまり、ドメインをグループ化して、キューを同時に実行する。このように改良されたため、接続キャッシュがある場合には、それをより有効に使用できる 新しい引数 filename は、キューをファイル名でソートする。この機能により、キューを実行する準備をする際に、各キューにあるファイルを開いたり読み込んだりすることを避けることができる 新しい引数 modification は、キューを変更日時でソートし、qf ファイルのエントリを古い順に実行する 新しい引数 random は、キューを無作為にソートする。こうすると、複数のキューランナーを手動で開始する際に、回線争奪を避けることができる 詳細については、sendmail(1M) マニュアルページの「QueueSortOrder」を参照
RefuseLA	m4 名: confREFUSE_LA 引数 : 数値 デフォルト値は、12 からシステム起動時にオンラインであるプロセッサ数の 12 倍に変更された。 単一プロセッサマシンでは、このデフォルト値の変更による影響はない。 この値を変更するとデフォルト値が無効になり、プロセッサ数を考慮しなくなる。 そのため、値を変更することによる影響について、よく理解する必要がある
ResolverOptions	このオプションについては、2 つの点が変更された。 ホスト名を正規化しようとする時、不具合が発生したネームサーバーが IPv6 T_AAAA 参照について一時障害メッセージ SERVFAIL を返すことがある。新しい引数 WorkAroundBrokenAAAA を使用して、このような動作を避けることができる また、引数 RES_USE_INET6 は、新しいフラグ use_inet6 を使って制御できる。詳細は、resolver(3RESOLV) のマニュアルページを参照
RrtImpliesDsn	m4 名: confRRT_IMPLIES_DSN 引数 : true または false この新しいオプションを設定すると、「Return-Receipt-To:」ヘッダーにより、DSN (delivery status notification) が要求される。この要求は、ヘッダーで指定されているアドレスではなく、エンベロープの送信側に送信される

表 **27–2** sendmail の新しいオプションおよび改訂されたオプション　(続き)

オプション	説明
SendMimeErrors	m4 名: confMIME_FORMAT_ERRORS 引数 : true または false デフォルトは true
SharedMemoryKey	m4 名: confSHARED_MEMORY_KEY 引数 : 数値 この新しいオプションを使用すると、共有メモリーがある場合に、それを使ってキューのファイルシステム用の容量を保存できる。このオプションにより、空き容量を確認するシステムコールの数を最小限にすることができる
SuperSafe	m4 名: confSAFE_QUEUE 引数 : true、false、または interactiveデフォルト値および推奨値は true。false は使用しないこと このオプションを true に設定すると、キューのファイルをすぐに配信する場合でも、それらは常にインスタンス化される。 interactive と DeliveryMode=i を同時に使用して、このモード用に、コード実行パスで重複している同期コールをスキップすることができる
Timeout	詳細については、416 ページの「Timeout オプションの変更点」 を参照
TrustedUser	m4 名: confTRUSTED_USER 引数 : ユーザー名またはユーザー *ID* の数値 この新しいオプションを使用して、root の代わりに重要なファイルを所有するユーザー名を指定することができる。このオプションを設定すると、そのユーザーは、生成された別名データベースと、設定した場合には制御ソケットを自動的に所有する。このオプションには、HASFCHOWN を設定する必要がある。HASFCHOWN については、360 ページの「sendmail のコンパイルに使用できるフラグと使用できないフラグ」を参照 TrustedUser、root、およびクラス t ($=t) のユーザーだけが、別名マップを構築できる
UseMSP	m4 名: confUSE_MSP 引数 : true または false デフォルトは、false この新しいオプションにより、グループが sendmail バイナリのグループ ID セットグループと同じ場合には、キューファイルをそのグループについて書き込み可能にすることが許可される。 submit.cf では、このオプションを true に設定する必要がある

表 **27–2** sendmail の新しいオプションおよび改訂されたオプション　(続き)

オプション	説明
XscriptFileBufferSize	m4 名: confXF_BUFFER_SIZE 引数 : 数値 この新しいオプションを指定すると、ディスクベースのファイルを使用する前にメモリーに蓄積できるトランスクリプト (xf) ファイルの最大サイズを、バイトで制御できる。 デフォルトは 4096 バイト。Solaris オペレーティング環境のデフォルトを変更する必要はない

sendmail の構成ファイルにおける推奨されないオプションまたはサポートされていないオプション

構成ファイルで推奨されないオプションについては、次の表を参照してください。この表には AutoRebuildAliases オプションが記載されていますが、このオプションは、sendmail バージョン 8.12 には含まれていません。

表 **27–3** sendmail の構成ファイルにおける推奨されないオプションまたはサポートされていないオプション

オプション	説明
AutoRebuildAliases	このオプションを設定するとサービス妨害攻撃が実行されることがあるため、このオプションは sendmail バージョン 8.12 には含まれていない。ftp://ftp.sendmail.org で sendmail とともに配布しているリリースノートを参照。 別名ファイルを構築中に、sendmail の処理を停止して、そのファイルを矛盾した状態のままにすることができる さらに、AutoRebuildAliases を使用できないため、/etc/mail/aliases に加えた変更を適用するには、newaliases を手動で実行する必要がある。また、このバージョンでは、sendmail は setuid root ではないため、root だけが newaliases を実行できる。
MeToo	このオプションのデフォルトは True になっており、その使用を推奨されていない ftp://ftp.sendmail.org で sendmail とともに配布しているリリースノートを参照
UnsafeGroupWrites	このオプションは推奨されていない。 必要に応じて、GroupWritableForwardFileSafe および GroupWritableIncludeFileSafe の引数を DontBlameSendmail オプションに使用する必要がある
UseErrorsTo	このオプションは推奨されていない。 また、このオプションは RFC 1123 に違反するため、使用しないこと

新しい ClientPortOptions オプション

新しく追加された ClientPortOptions オプションは発信接続に使用します。このオプションは、DaemonPortOptions オプションに似ています。このオプションにより、クライアントの SMTP オプションが設定されます。クライアントの SMTP オプションは、一連の *key=value* ペアです。 このオプションを宣言するには、次の構文のどれかを使用します。 フォーマットのために、これらの例には 2 組のペアが含まれています。 ただし、1 組以上のペアを適用できます。

```
O ClientPortOptions=pair,pair                # 構成ファイル
-OClientPortOptions=pair,pair                # コマンド行
define('confCLIENT_OPTIONS','pair,pair')     # m4 を使った構成記述
```

新しい sendmail.cf ファイルを構築する必要がある場合は、第 25 章 の 329 ページの「sendmail.cf 構成ファイルの構築 (手順) 」 を参照してください。

次の表では、このオプションの新しいキーについて説明しています。

表 **27–4** ClientPortOptions の新しいキー

キー	説明
Addr	アドレスマスクを指定する。 この値は、ドット表記した数値のアドレスにすることも、ネットワーク名にすることもできる。このペアが省略されると、デフォルトは INADDR_ANY となり、どのネットワークからの接続も受け入れる
Family	アドレスファミリーを指定する。 AF_INET のキーのデフォルトは inet。 他の値は、AF_INET6 には inet6、AF_ISO には iso、AF_NS には ns、AF_CCITT には x.25 である
Listen	待機キューのサイズを指定する。 キーのデフォルトは 10。 Solaris オペレーティング環境のデフォルトを変更する必要はない
Port	待機ポートの名前および番号を指定する。 キーのデフォルトは smtp
RcvBufSize	TCP/IP 送信バッファーのサイズを指定する。キーにはデフォルト値がないため、サイズが自動的に設定されることはない。このオプションを 0 より大きな値に設定すると、その値が使用される。Solaris オペレーティング環境では、このバッファーのサイズを制限する必要はない

表 27–4 ClientPortOptions の新しいキー　(続き)

キー	説明
Modifier	次のような sendmail のフラグを指定する h フラグは、HELO または EHLO コマンドに、送信インタフェースアドレスに対応する名前を使用する。これは、その名前が接続パラメータで選択されたものであっても、デフォルトのものであっても同様である A フラグは、AUTH を無効にする。このフラグは、DaemonPortOptions の Modifier キーに使用できる。 413 ページの「DaemonPortOptions オプションの変更点」 を参照 S フラグは、電子メールの配信中または受信中に、STARTTLS を使用できないようにしたり、それを提供したりしないようにする

DaemonPortOptions オプションの変更点

次の表では、新しい機能について説明しています。

- 表 27–5
- 表 27–6

このオプションを宣言するには、次の構文のどれかを使用します。 この例では、*pair* は *key=value* を示します。フォーマットのために、これらの例には 2 組のペアが含まれています。 ただし、1 組以上のペアを適用できます。

```
O DaemonPortOptions=pair,pair                  # 構成ファイル
-ODaemonPortOptions=pair,pair                  # コマンド行
define(‘confDAEMON_OPTIONS’,‘pair,pair’)       # m4 を使った構成記述
```

注 – セキュリティのリスクを少なくするために、このオプションをコマンド行から設定すると、sendmail はスーパーユーザーアクセス権を放棄します。

新しい sendmail.cf ファイルを構築する必要がある場合は、第 25 章 の 329 ページの「sendmail.cf 構成ファイルの構築 (手順) 」 を参照してください。

次の表では、DaemonPortOptions オプションの新しいキーおよび改訂されたキーについて説明しています。

表 27–5 DaemonPortOptions の新しいキーおよび改訂されたキー

キー	説明
Name	この新しいキーは、ユーザーが定義可能な sendmail の名前を指定する。このキーは、エラーメッセージおよびログに使用する。デフォルトは、MTA

表 **27–5** DaemonPortOptions の新しいキーおよび改訂されたキー　*(続き)*

キー	説明
Modifier	この新しいキーは、sendmail の値を指定する。この値は、区切り記号なしで、順番にリストすることができる。値のリストについては、表 27–6 を参照
Family	DaemonPortOptions オプションで、Family を指定しないかぎり、inet だけがデフォルトとなる。また、IPv6 ユーザーが、IPv6 インタフェースに対しても待機するには、Family=inet6 設定を DaemonPortOptions オプションに追加して、追加ソケットを sendmail.cf に設定する

次の表では、新しい Modifier キーの値について説明しています。

表 **27–6** 新しい Modifier キーの値

値	説明
A	Modifier 値を a にして、AUTH を無効にする
	ClientPortOptions の Modifier キーに使用できる。 412 ページの「新しい ClientPortOptions オプション」 を参照
C	ホスト名の正規化を実行しない
E	ETRN コマンドを不許可にする
O	障害が発生したら、ソケットを無視する
S	電子メールの配信中または受信中に、STARTTLS を使用できないようにしたり、それを提供したりしないようにする
	ClientPortOptions の Modifier キーに使用できる。
a	認証を要求する
b	メールを受信するインタフェースに結合する
c	ホスト名の正規化を実行する。この値は、構成ファイルの宣言でのみ使用する
f	完全指定ホスト名を要求する。この値は、構成ファイルの宣言でのみ使用する
h	送信 HELO コマンドに、インタフェース名を使用する
u	修飾されていないアドレスを使用する。この値は、構成ファイルの宣言でのみ使用する

PidFile オプションおよび ProcessTitlePrefix オプションのその他の引数

次の表では、PidFile オプションおよび ProcessTitlePrefix オプションにおけるマクロ処理の引数について説明します。これらのオプションについては、表 27–2 を参照してください。

表 27–7 PidFile オプションおよび ProcessTitlePrefix オプションの引数

マクロ	説明
${daemon_addr}	0.0.0.0 などのデーモンアドレスを提供する
${daemon_family}	inet や inet6 などのデーモンファミリーを提供する
${daemon_info}	SMTP+queueing@00:30:00 などのデーモン情報を提供する
${daemon_name}	MSA などのデーモン名を提供する
${daemon_port}	25 などのデーモンポートを提供する
${queue_interval}	キューを実行する間隔を提供する (00:30:00 など)

PrivacyOptions オプションの変更点

次の表では、PrivacyOptions (popt) の新しい引数および改訂された引数を説明しています。このオプションは、sendmail がその root 権限を放棄することなく、コマンド行から宣言できます。この sendmail オプションを宣言するには、次の構文のどれかを使用します。

```
O PrivacyOptions=argument                    # 構成ファイル
-OPrivacyOptions=argument                    # コマンド行
define(`confPRIVACY_FLAGS',`argument')       # m4 を使った構成記述
```

新しい sendmail.cf ファイルを構築する必要がある場合は、第 25 章 の 329 ページの「sendmail.cf 構成ファイルの構築 (手順)」 を参照してください。

次の表では、PrivacyOptions オプションの新しい引数および改訂された引数について説明しています。

表 27–8 PrivacyOptions の新しい引数および改訂された引数

引数	説明
goaway	この引数には、フラグ noetrn、restrictmailq、restrictqrun、restrictexpand 、nobodyreturn、および noreceipts は使用できない
nobodyreturn	この引数は、元のメッセージの本文を DNS (Delivery Status Notifications) に含めないように、sendmail に指示する

表 **27–8** PrivacyOptions の新しい引数および改訂された引数　*(続き)*

引数	説明
noreceipts	この引数を設定すると、DSN (Delivery Status Notifications) が通知されない
restrictexpand	この引数は、root でも TrustedUser でもないユーザーが -bv オプションを指定した場合に、権限を解除するように sendmail に指示する。ユーザーは、.forward ファイルまたは :include: ファイルなどの非公開の別名を読み込むことができない。 また、この引数は、コマンド行オプションの -v を無効にする

Timeout オプションの変更点

次の表では、Timeout オプションの変更点について説明しています。具体的に言うと、この sendmail オプションには、ident における新しいキーワードおよび新しい値があります。Solaris オペレーティング環境では、この表に表示されているキーワードのデフォルト値を変更する必要はありません。ただし、変更する場合には、*keyword=value* の構文を使用してください。この *value* は、時間の間隔です。次の例を参照してください。

```
O Timeout.keyword=value    # 構成ファイル
-OTimeout.keyword=value    # コマンド行
define('m4_name', value) # m4 を使った構成記述
```

新しい sendmail.cf ファイルを構築する必要がある場合は、第 25 章 の 329 ページの「sendmail.cf 構成ファイルの構築 (手順) 」 を参照してください。

注 – セキュリティのリスクを少なくするために、このオプションをコマンド行から設定すると、sendmail はスーパーユーザーアクセス権を放棄します。

表 **27–9** Timeout の新しい設定および改訂された設定

キーワード	デフォルト値	説明
aconnect	0	m4 名: confTO_ACONNECT 1 回の配信について、すべての接続が成功するまでの合計時間を制限する。最大値は指定されていない
control	2m	m4 名: confTO_CONTROL 制御ソケットの要求を完了するまでにかかる合計時間を制限する

表 **27–9** Timeout の新しい設定および改訂された設定　(続き)

キーワード	デフォルト値	説明
ident	5s	m4 名: confTO_IDENT デフォルトでは、30 秒ではなく 5 秒。IDENT パケットを欠落させるサイトへのメール送信が原因で発生する通常の遅延を防止する。最大値は指定されていない
lhlo	2m	m4 名: confTO_LHLO LMTP LHLO コマンドからの応答を待つ時間を制限する。最大値は指定されていない
queuereturn	5d	m4 名: confTO_QUEUERETURN 値 now を含める。こうすると、キューにあるエントリを配信しないで、すぐに戻すことができる
resolver.retrans	状況により異なる	m4 名: confTO_RESOLVER_RETRANS リゾルバによる再伝送の間隔を秒で指定する。この間隔は、resolver.retrans.first および resolver.retrans.normal に適用される
resolver.retrans.first	状況により異なる	m4 名: confTO_RESOLVER_RETRANS_FIRST リゾルバが、メッセージをはじめて配信する際の再送の間隔を秒で設定する
resolver.retrans.normal	状況により異なる	m4 名: confTO_RESOLVER_RETRANS_NORMAL リゾルバが、最初のメッセージ配信を除く、すべての参照を実行する際の再伝送の間隔を指定する
resolver.retry	状況により異なる	m4 名: confTO_RESOLVER_RETRY リゾルバクエリーを再送する回数を設定する。この回数は、Timeout.resolver.retry.first および Timeout.resolver.retry.normal に適用される
resolver.retry.first	状況により異なる	m4 名: confTO_RESOLVER_RETRY_FIRST メッセージをはじめて配信する際にリゾルバクエリーを再送する回数を設定する
resolver.retry.normal	状況により異なる	m4 名: confTO_RESOLVER_RETRY_NORMAL 最初のメッセージ配信を除く、すべてのリゾルバ参照を実行する際に、リゾルバクエリーを再送する回数を設定する

sendmail に新しく定義されたマクロ

次の表では、sendmail プログラムで使用するための新しいマクロについて説明しています。マクロの値は、内部で割り当てられています。詳細は、sendmail(1M) のマニュアルページを参照してください。

表 **27–10** sendmail に定義されたマクロ

マクロ	説明
${addr_type}	現在のアドレスを、エンベロープの送信側または受信者アドレスと認定する
${client_resolve}	${client_name} の解釈処理コールの結果、つまり OK、FAIL、FORGED、または TEMP を保持する
${deliveryMode}	DeliveryMode オプションの値ではなく、sendmail が使用している現在のデリバリモードを指定する
${dsn_notify}、${dsn_envid}、${dsn_ret}	対応する DSN パラメータ値を保持する
${if_addr}	インタフェースがループバックネット上にない場合に、受信接続用インタフェースのアドレスを提供する。このマクロは、特に仮想ホスティングに便利である
${if_addr_out}、${if_name_out}、${if_family_out}	${if_addr} の再利用を避ける。次の値を、それぞれ保持する
	送信接続用インタフェースのアドレス
	送信接続用インタフェースのホスト名
	送信接続用インタフェースのファミリ
${if_name}	受信接続用のインタフェースのホスト名を提供する。これは、特に仮想ホスティングに便利である
${load_avg}	実行キューにあるジョブの現在の平均数を確認して報告する
${msg_size}	ESMTP ダイアログにあるメッセージサイズの値 (SIZE=parameter) を保持してから、メッセージを収集する。その後、sendmail によって計算されたメッセージサイズを保持したマクロを check_compat で使用する。check_compat については、表 27–14 を参照
${nrcpts}	妥当性検査を行った受信者の数を保持する
${ntries}	配信を試みた回数を保持する
${rcpt_mailer}、 ${rcpt_host}、${rcpt_addr}、${mail_mailer}、${mail_host}、および${mail_addr}	引数 RCPT および MAIL を構文解析した結果を保持する。つまり、メール配信エージェント ($# *mailer*)、ホスト ($@*host*)、およびユーザー ($: *addr*) から解釈処理された RHS (Right-Hand Side) トリプレットを保持する

sendmail 構成ファイルを構築するのに使用する新しいマクロ

この節では、以下について説明します。

- 表 27–11
- 419 ページの「新しい MAX マクロ」

表 **27–11** sendmail 構成ファイルを構築するのに使用する新しいマクロ

マクロ	説明
LOCAL_MAILER_EOL	ローカルメールプログラムの行末を示すデフォルト文字列を置きかえる
LOCAL_MAILER_FLAGS	デフォルトでは、Return-Path: ヘッダーを追加する
MAIL_SETTINGS_DIR	メール設定ディレクトリのパスを格納する (末尾のスラッシュを含む)
MODIFY_MAILER_FLAGS	*_MAILER_FLAGS を拡張する。このマクロは、フラグを設定、追加、または削除する
RELAY_MAILER_FLAGS	リレーメールプログラムの追加フラグを定義する

新しい MAX マクロ

次の新しいマクロを使用して、受け入れ可能なコマンドを最大数設定し、sendmail による配信の遅れを防止することができます。これらの MAX マクロは、コンパイル時に設定できます。次の表にある最大値は、現在のデフォルト値でもあります。

表 **27–12** 新しい MAX マクロ

マクロ	最大値	各マクロが確認するコマンド
MAXBADCOMMANDS	25	未知なコマンド
MAXNOOPCOMMANDS	20	NOOP、VERB、ONEX 、XUSR
MAXHELOCOMMANDS	3	HELO、EHLO
MAXVRFYCOMMANDS	6	VRFY、EXPN
MAXETRNCOMMANDS	8	ETRN

注 – マクロによる確認を無効にするには、マクロの値を 0 に設定します。

sendmail における新しい m4 構成マクロおよび改訂された m4 構成マクロ

この節では、sendmail における新しい m4 構成マクロおよび改訂された m4 構成マクロを、表を使って説明します。これらのマクロを宣言するには、次の構文を使用します。

symbolic_name(‘*value*’)

新しい sendmail.cf ファイルを構築する必要がある場合は、第 25 章 の 329 ページの「sendmail.cf 構成ファイルの構築 (手順) 」 を参照してください。

表 27–13 sendmail における新しい m4 構成マクロおよび改訂された m4 構成マクロ

m4 マクロ	説明
FEATURE()	詳細は、420 ページの「FEATURE() の宣言についての変更点」を参照してください。
LOCAL_DOMAIN()	このマクロは、クラス w ($=w) にエントリを追加する
MASQUERADE_EXCEPTION()	マスカレードできないホストやサブドメインを定義する新しいマクロ
SMART_HOST()	このマクロは user@[*host*] のように、括弧で囲まれたアドレスに使用できる
VIRTUSER_DOMAIN() または VIRTUSER_DOMAIN_FILE()	これらのマクロを使用する場合は、$=R に $={VirtHost} を含める。 $=R は一連のホスト名で、これらを使ってリレーすることができる

FEATURE() の宣言についての変更点

FEATURE() の宣言についての変更点については、次の表を参照してください。

- 表 27–14
- 表 27–15

FEATURE の新しい名前および改訂された名前を使用するには、次の構文を使用します。

FEATURE(‘*name*’, ‘*argument*’)

新しい sendmail.cf ファイルを構築する必要がある場合は、第 25 章 の 329 ページの「sendmail.cf 構成ファイルの構築 (手順) 」 を参照してください。

表 **27–14** 新規および改訂された FEATURE() の宣言

FEATURE() の名前	説明
compat_check	引数 : 次の段落の例を参照 この新しい FEATURE() を使用して、送信側のアドレスおよび受信者のアドレスを含むアクセスマップからキーを検索できる。その際には、<@> で区切る。 例 - *sender@sdomain*<@>*recipient@rdomain*
delay_checks	引数 : friend にすると、スパムメールの friend テストを実行できる。また、hater にすると、スパムメールの hater テストを実行できる。 すべての確認作業を遅らせる新しい FEATURE()。FEATURE('delay_checks') を使用すると、クライアントが接続する場合、またはクライアントが MAIL コマンドを発行する場合に、ルールセット check_mail および check_relay は呼び出されない。代わりに、これらのルールセットはルールセット check_rcpt によって呼び出される。詳細については、/usr/lib/mail/README ファイルを参照
dnsbl	引数 : この FEATURE() は、最大次の 2 つの引数を受け入れる。 ■ DNS サーバー名 ■ リジェクトメッセージ DNS 参照の戻り値を確認する回数を複数にできる新しい FEATURE()。この FEATURE() を使用して、参照が一時的に失敗した場合の動作を指定できる
enhdnsbl	引数 : ドメイン名 dnsbl の強化バージョン。この FEATURE() を使用して、DNS 参照の戻り値を確認できる。詳細は、/usr/lib/mail/README を参照
generics_entire_domain	引数 : なし genericstable を $=G のサブドメインに適用するのに使用する新しい FEATURE()
ldap_routing	引数 : 詳細については、http://www.sendmail.org の「リリースノート」を参照 LDAP アドレスルーティングを実装する新しい FEATURE()
local_lmtp	引数 : LMTP (Local Mail Transfer Protocol) を使用できるメールプログラムのパス名。デフォルトは mail.local であり、今回の Solaris リリースでは LMTP を使用できる ローカルメールプログラムの DSN (delivery status notification) 診断コードのタイプを SMTP の正しい値に設定する FEATURE()
local_no_masquerade	引数 : なし ローカルメールプログラムをマスカレードしないようにするために使用する新しい FEATURE()

表 **27–14** 新規および改訂された FEATURE() の宣言　*(続き)*

FEATURE() の名前	説明
lookupdotdomain	引数 : なし アクセスマップの *.domain* を参照するのに使用する新しい FEATURE()
nocanonify	引数 : canonify_hosts またはなし FEATURE() には次の機能が含まれている CANONIFY_DOMAIN または CANONIFY_DOMAIN_FILE で指定した、ドメインのリストを演算子 $[および $] に渡して正規化することができる canonify_hosts がそのパラメータとして指定されている場合には、ホスト名だけを持つアドレス (<user@host> など) を正規化できる 複数のコンポーネントを持つアドレスの末尾にドットを追加できる
no_default_msa	引数 : なし sendmail のデフォルト設定を m4 構成ファイルでオフにする新しい FEATURE()。このファイルは、複数の異なるポート上で待機するために生成されたもので、RFC 2476 に実装されている
nouucp	引数 : reject にすると、! トークンを使用できない。nospecial にすると、! トークンを使用できる ! トークンをアドレスのローカルの部分に使用するかどうかを決定する FEATURE()
nullclient	引数 : なし 通常の構成ですべてのルールセットを提供する FEATURE()。スパムメール対策チェックを実行する
preserve_local_plus_detail	引数 : なし sendmail がアドレスをローカル配信エージェントに渡す際に、アドレスの +detail の部分を保存できる新しい FEATURE()
preserve_luser_host	引数 : なし LUSER_RELAY を使用している場合に、受信者のホスト名を保存できる新しい FEATURE()
queuegroup	引数 : なし 電子メールのアドレス全体または受信者のドメインに基づいたキューグループを選択できる新しい FEATURE()
relay_mail_from	引数 : ドメインは、任意の引数 メールの送信側がアクセスマップに RELAY として指定されており、それをヘッダー行 From: で呼び出せる場合に、リレーを許可する新しい FEATURE()。 任意の引数ドメインを指定すると、メール送信側のドメインの部分が確認される

表 **27–14** 新規および改訂された FEATURE() の宣言　*(続き)*

FEATURE() の名前	説明
virtuser_entire_domain	引数 : なし $={VirtHost} を適用するのに使用する FEATURE()。$={VirtHost} は、VIRTUSER_DOMAIN または VIRTUSER_DOMAIN_FILE を使って生成できる virtusertable エントリを一致させるための新しいクラス また、FEATURE('virtuser_entire_domain') を使用して、クラス $={VirtHost} をサブドメイン全体に適用することもできる

次の FEATURE() は、宣言できません。

表 **27–15** 宣言がサポートされていない FEATURE()

FEATURE() の名前	代わりの FEATURE()
rbl	削除されたこの FEATURE() の代わりに、FEATURE('dnsbl') および FEATURE('enhdnsbl') を使用できる
remote_mode	/usr/lib/mail/cf/subsidiary.mc では、FEATURE('remote_mode') の代わりに MASQUERADE_AS('$S') を使用できる。$S は、sendmail.cf における SMART_HOST の値
sun_reverse_alias_files	FEATURE('genericstable').
sun_reverse_alias_nis	FEATURE('genericstable').
sun_reverse_alias_nisplus	FEATURE('genericstable').

MAILER() の宣言についての変更点

MAILER() を宣言すると、配信エージェントのサポートを指定できます。 配信エージェントを宣言するには、次の構文を使用します。

MAILER('*symbolic_name*')

次の変更に注意してください。

- この新しいバージョンの sendmail では、MAILER('smtp') を宣言すると、メールプログラム dsmtp が追加されます。dsmtp により、メールプログラムのフラグ F=% を使用して、オンデマンドに配信することができます。dsmtp メールプログラムを定義する際には、新しい DSMTP_MAILER_ARGS を使用します。DSMTP_MAILER_ARGS のデフォルトは IPC $h です。
- MAILER によって使用されるルールセットの番号は削除されました。 MAILER ('uucp') を除いて、MAILER の表示順を自由に設定できます。uucp-dom および uucp-uudom を使用する場合には、MAILER('uucp') の後に MAILER('smtp')

を配置する必要があります。

メールプログラムの詳細については、364 ページの「メールプログラム」 を参照してください。新しい sendmail.cf ファイルを構築する必要がある場合は、第 25 章 の 329 ページの「sendmail.cf 構成ファイルの構築 (手順)」 を参照してください。

配信エージェントの新しいフラグ

次の表では、配信エージェントの新しいフラグについて説明しています。デフォルトでは、これらのフラグは設定されていません。これらの 1 文字のフラグはブール型です。このフラグを設定したりその設定を解除したりするには、次の例のように、フラグを構文ファイルの F= 文に含めるか除外します。

```
Mlocal,     P=/usr/lib/mail.local, F=lsDFMAw5:/|@qSXfmnz9, S=10/30, R=20/40,
Mprog,      P=/bin/sh, F=lsDFMoqeu9, S=10/30, R=20/40, D=$z:/,
Msmtp,      P=[IPC], F=mDFMuX, S=11/31, R=21, E=\r\n, L=990,
Mesmtp,     P=[IPC], F=mDFMuXa, S=11/31, R=21, E=\r\n, L=990,
Msmtp8,     P=[IPC], F=mDFMuX8, S=11/31, R=21, E=\r\n, L=990,
Mrelay,     P=[IPC], F=mDFMuXa8, S=11/31, R=61, E=\r\n, L=2040,
```

表 27–16 メールプログラムの新しいフラグ

フラグ	説明
%	このフラグを使用するメールプログラムは、ETRN 要求やキューオプション -qI、-qR、または -qS のいずれかを使ってキューにあるメッセージを選択しないかぎり、最初の受信者宛にメールを配信したり、キューを実行したりしない
1	このフラグは、\0 などのヌル文字を送信するメールプログラムの機能を無効にする
2	このフラグは、ESMTP の使用を無効にし、代わりに SMTP を使用するように要求する
6	このフラグを指定すると、メールプログラムでヘッダーを 7 ビットにすることができる

配信エージェントの新しい等号 (=)

次の表では、配信エージェントを定義するコマンド M とともに使用できる新しい等号 (=) について説明しています。次の構文は、等号 (=) を新たに付加する方法、および構成ファイルの既存の等号に新しい引数を付加する方法を示しています。

M*agent_name*, *equate*, *equate*, ...

次の例には、新しい等号 (=) W= が含まれています。この W= は、データが送信された後でメールプログラムが戻るまでの最長待ち時間を示します。

```
Msmtp, P=[IPC], F=mDFMuX, S=11/31, R=21, E=\r\n, L=990, W=2m
```

m4 の構成値の定義を変更するには、次の例のような構文を使用します。

```
define(‘SMTP_MAILER_MAXMSGS’, ‘1000’)
```

この例では、smtp メールプログラムで 1 回の接続で配信されるメッセージ数を 1000 に制限しています。

新しい sendmail.cf ファイルを構築する必要がある場合は、第 25 章 の 329 ページの「sendmail.cf 構成ファイルの構築 (手順)」 を参照してください。

注 – 通常、mailer ディレクトリで、この等号の定義を変更するのは、微調整が必要な場合だけです。

表 27–17 配信エージェントの新しい等号 (=)

等号	説明
/=	引数 : ディレクトリのパス メールプログラムのプログラムを実行する前に chroot() を適用するディレクトリを指定する
m=	引数 : define() ルーチンを使って事前に定義した次の m4 の値 smtp メールプログラムには SMTP_MAILER_MAXMSGS local メールプログラムには LOCAL_MAILER_MAXMSGS relay メールプログラムには RELAY_MAILER_MAXMSGS smtp、local、または relay の各メールプログラムで、1 回の接続で配信するメッセージの数を制限する
W=	引数 : 増分時間 すべてのデータの送信後、メールプログラムが戻るまでの最長待ち時間を指定する

キューの新しい機能

次に、キューの新しい機能について詳しく説明します。

- 本リリースでは、複数のキューディレクトリをサポートしています。複数のキューを使用するには、次の例のように、アスタリスク (*) で終わっている QueueDirectory オプション値を構成ファイルに追加します。

  ```
  O QueueDirectory=/var/spool/mqueue/q*
  ```

  このオプション値 /var/spool/mqueue/q* は、「q」で始まっているすべてのディレクトリ (またはディレクトリへのシンボリックリンク) をキューのディレクトリとして使用します。sendmail の実行中には、キューのディレクトリ構造を変更しないでください。キューを実行すると、デーモン以外のキューの実行時に冗

長フラグ (-v) を使用しないかぎり、各キューについての実行プロセスが作成されます。この新しい項目が、無作為にキューに割り当てられます。

- この新しいキューのファイルの名前付けシステムで使用する名前は、60 年間一意であることが保証されます。 このシステムでは、キュー ID が複雑なファイルシステムのロックを使用しないで割り当てられるため、キューにある項目を簡単に他のキューに移動することができます。
- `sendmail` バージョン 8.12 では、`root` だけがメインキューを実行できます。 この変更の詳細については、`mailq(1)` のマニュアルページを参照してください。新しい作業手順については、344 ページの「キューディレクトリの管理 (作業マップ)」 を参照してください。
- エンベロープの分割に対応するために、キューファイルの名前は 14 文字ではなく、15 文字にします。 14 文字までの名前を持つファイルシステムは、サポートされません。

関連する作業については、344 ページの「キューディレクトリの管理 (作業マップ)」を参照してください。

`sendmail` における LDAP のための変更

次に、LDAP (Lightweight Directory Access Protocol) を `sendmail` で使用する際の変更点について説明します。

- `LDAPROUTE_EQUIVALENT()` および `LDAPROUTE_EQUIVALENT_FILE()` を使用して、同じホスト名を指定することができます。これらのホスト名は、LDAP ルーティング参照のマスカレードドメイン名と置き換えられます。詳細は、`/usr/lib/mail/README` を参照してください。
- ftp://ftp.sendmail.org で `sendmail` とともに配布しているリリースノートで説明しているように、LDAPX マップの名前は LDAP に変更されました。 LDAP には、次の構文を使用します。

  ```
  Kldap ldap options
  ```

- 本リリースでは、一度の LDAP 参照に複数の値を返すことができます。次の例のように、返す値を `-v` オプションを付加したコンマ区切りの文字列に配置します。

  ```
  Kldap ldap -v"mail,more_mail"
  ```

- LDAP マップの宣言で LDAP 属性が指定されていない場合は、一致した属性がすべて返されます。
- このバージョンの `sendmail` は、LDAP 別名ファイルに指定された引用符などで囲まれたキーや値の文字列内のコンマによって、1 つのエントリが複数のエントリに分割されるのを防止します。
- このバージョンの `sendmail` には、LDAP マップ用の新しいオプションがあります。この -V*separator* オプションを使用して、区切り文字を指定できます。そのため、参照を行うと、該当する *separator* によって区切られた属性と値の両方が返されることがあります。

- `%s` トークンを使用して LDAP フィルタ指定を構文解析する代わりに、新しいトークンである `%0` を使用して、キーバッファーを符号化することができます。`%0` トークンは、LDAP の特殊文字に対して、文字どおりの意味を適用します。

  次の例では、`*` にこれらのトークンを使って参照する場合の相違点について説明します。

  表 **27–18** トークンの比較

LDAP のマップ指定	同等の指定	結果
`-k"uid=%s"`	`-k"uid=*"`	レコードとユーザー属性を照合する
`-k"uid=%0"`	`-k"uid=\2A"`	ユーザーと名前「*」を照合する

次の表では、LDAP マップの新しいフラグについて説明しています。

表 **27–19** LDAP マップの新しいフラグ

フラグ	説明
`-1`	一致したレコードが 1 つだけだった場合、そのレコードを返す。複数のレコードが一致して返される場合には、結果として、レコードが検出されなかったことと同じとなる
`-r` *never*|*always*|*search*|*find*	LDAP 別名の参照を解除するオプションを設定する
`-Z` *size*	一致したもののうち、返すレコード数を制限する

メールプログラムに新しく組み込まれた機能

前のバージョンに組み込まれていたメールプログラム `[TCP]` は使用できません。代わりに、新しく組み込まれたメールプログラム `P=[IPC]` を使用してください。新しく組み込まれたプロセス間通信メールプログラム（`[IPC]`）を使用して、それをサポートしているシステム上の UNIX ドメインソケット宛てにメールを配信できます。このメールプログラムは、指定したソケットで待機している `LMTP` 配信エージェントとともに使用できます。次に、メールプログラムの例を示します。

```
Mexecmail, P=[IPC], F=lsDFMmnqSXzA5@/:|, E=\r\n,
S=10, R=20/40, T=DNS/RFC822/X-Unix, A=FILE /var/run/lmtpd
```

`[IPC]` メールプログラムの最初の引数が、正当な値であるかどうかが確認されます。次の表では、最初のメールプログラム引数に設定可能な値について説明しています。

表 **27–20** 最初のメールプログラム引数に設定可能な値

値	説明
A=FILE	UNIX ドメインソケットによる配信に使用する
A=TCP	TCP/IP 接続に使用する
A=IPC	最初のメールプログラム引数としては使用できない

新しいルールセット

次の表では、新しいルールセットとその動作について説明しています。

表 **27–21** 新しいルールセット

ルールセット	説明
check_eoh	ヘッダーから収集した情報を相関させ、欠けているヘッダーを確認する。 このルールセットは、マクロストレージマップとともに使用し、すべてのヘッダーが収集された後、呼び出される
check_etrn	check_rcpt が RCPT を使用するように、ETRN コマンドを使用する
check_expn	check_rcpt が RCPT を使用するように、EXPN コマンドを使用する
check_vrfy	check_rcpt が RCPT を使用するように、 VRFY コマンドを使用する

次に、ルールセットの新しい機能について説明します。

- 番号が付けられたルールセットには、名前も付けられました。ただし、これらのルールセットに、番号でアクセスすることもできます。
- H ヘッダー構成ファイルコマンドを使用して、デフォルトのルールセット指定し、ヘッダーを確認することができます。各ヘッダーに、独自のルールセットが割り当てられていない場合にだけ、このルールセットが呼び出されます。
- ルールセット内のコメント、つまり括弧内のテキストは、構成ファイルのバージョンが 9 かそれ以上である場合には削除されません。たとえば、次のルールは、入力 token (1) を照合します。ただし、入力 token は照合しません。

```
R$+ (1)         $@ 1
```

- TCP ラッパーまたは check_relay ルールセットが原因でコマンドを拒否する場合でも、sendmail は SMTP RSET コマンドを受け入れます。
- OperatorChars オプションを何度も設定すると、警告が送信されます。また、ルールセットを定義した後で OperatorChars を設定しないでください。
- 無効なルールセットを宣言すると、行だけでなく、そのルールセットの名前も無視されます。そのルールセットの行は S0 に追加されません。

ファイルへの変更

次の変更に注意してください。

- helpfile は、/etc/mail/helpfile にあります。 古い名前 (/etc/mail/sendmail.hf) には、新しい名前へのシンボリックリンクがあります。
- trusted-users ファイルは、/etc/mail/trusted-users にあります。 アップグレード中に、新しい名前は検出されず、古い名前である /etc/mail/sendmail.ct が検出されると、古い名前から新しい名前へのハードリンクが作成されます。 それ以外の場合には、変更されません。 デフォルトの内容は、root です。
- local-host-names ファイルは、/etc/mail/local-host-names にあります。 アップグレード中に、新しい名前は検出されず、古い名前である /etc/mail/sendmail.cw が検出されると、古い名前から新しい名前へのハードリンクが作成されます。 それ以外の場合には、変更されません。 デフォルトの内容は、ゼロ長です。
- /usr/lib/mail/cf/main-v7sun.mc の新しい名前は /usr/lib/mail/cf/main.mc です。
- /usr/lib/mail/cf/subsidiary-v7sun.mc の新しい名前は /usr/lib/mail/cf/subsidiary.mc です。

構成内の IPv6 アドレス

sendmail バージョン 8.12 では、アドレスを正しく識別するために、構成に使用する IPv6 アドレスの前に IPv6: タグを付ける必要があります。 IPv6 アドレスを識別しない場合は、タグを前に付けません。

mail.local の変更点

次の表では、mail.local プログラムにおけるコマンド行の新しいオプションについて説明しています。sendmail は、このプログラムをローカルメールの配信エージェントとして使用します。

表 **27–22** mail.local におけるコマンド行の新しいオプション

オプション	説明
-7	LMTP (Local Mail Transfer Protocol) モードで、LHLO 応答時に、8BITMIME のサポートが通知されるのを防止する

表 **27–22** mail.local におけるコマンド行の新しいオプション　*(続き)*

オプション	説明
-b	メールボックスがその制限を超えた場合に、一時エラーではなく、永続エラーを発生させる

LMTP モードのデフォルトは mail.local です。ただし、本リリースでは、LMTP モード以外で mail.local をローカル配信エージェントとして使用するには、次のどれかの操作を実行して S フラグを設定する必要があります。

構成ファイルには、次の構文を使用します。

```
MODIFY_MAILER_FLAGS('LOCAL', '+S')       # 構成ファイル
```

または、m4 構成に対して次の 2 つの手順を実行します。

```
define('MODIFY_MAILER_FLAGS', 'S')dnl    # 第 1 段階
MAILER(local)dnl                         # 第 2 段階
```

注 – MODIFY_MAILER_FLAGS は、構成ファイルを構築するのに使用する新しいマクロです。詳細は、419 ページの「sendmail 構成ファイルを構築するのに使用する新しいマクロ」を参照してください。

詳細については、mail.local(1M) のマニュアルページを参照してください。

mailstats の変更点

sendmail プログラムには、メールプログラムの使用状況を統計する機能を持つ mailstats プログラムが付属しています。次の表では、mailstats の新しいオプションについて説明します。

表 **27–23** mailstats の新しいオプション

オプション	説明
-C *filename*	sendmail 構成ファイルを指定する
-p	プログラムが読み取り可能なモードで、統計を明確に示す
-P	プログラムが読み取り可能なモードで、統計を明確に示す。ただし、このオプションを指定すると、統計ファイルは切り捨てられない

詳細は、mailstats(1) のマニュアルページを参照してください。

makemap の変更点

makemap コマンドを実行すると、sendmail 用にキー付きのデータベースファイルが作成されます。次の表では、makemap の新しいオプションについて説明しています。 オプションを宣言する場合には、次の構文を使用します。

makemap *options class filename*

この構文を使用するときには、次のことに注意してください。

- -dN のように、*option* の前にダッシュを付けます。
- *class* は、btree、dbm、または hash のようなデータベースのタイプです。
- *filename* は、データベースファイルへの完全パスまたは相対名です。

表 **27–24** makemap の新しいオプション

オプション	説明
-C	TrustedUser オプションの検出に、指定した sendmail 構成ファイルを使用する
-c	指定した hash および btree のキャッシュサイズを使用する
-e	RHS (right-hand side) から空の値を使用することを許可する
-l	サポートされているマップのタイプをリストする
-t	空白ではなく、別の区切り記号を指定する
-u	データベースの内容を標準出力にダンプ (マップ形式を元に戻す) する

注 – makemap を root として実行すると、生成されたマップの所有権は、構成ファイル sendmail で指定したように、自動的に TrustedUser に変更されます。TrustedUser オプションの詳細については、表 27–2 を参照してください。

詳細は、makemap(1M) のマニュアルページを参照してください。

新しいコマンド editmap

新しい保守コマンド editmap を使用して、sendmail のキー付きデータベースマップのレコードを照会したり編集したりすることができます。コマンド行から、次の構文を使用します。

`editmap` *options maptype mapname key* "*value*"

- `-Nf` のように、*option* の前にダッシュを付けます。 使用できるオプションとその機能については、マニュアルページで説明しています。
- *maptype* は、データベースのタイプです。`editmap` では、`btree`、`dbm`、および `hash` を使用できます。
- *mapname* は、データベースファイルへの完全パスまたは相対名です。
- *key* は、検索に使用する単一の文字列または複数トークン文字列です。
- 「*value*」は、キー付きのデータベースファイル内で、キーの右側に表示される文字列です。次の例では、キーは `man` で、`man@example.com` がそのキーに割り当てられている値です。

```
man    man@host.com
```

詳細および使用可能なオプションについては、`editmap(1M)` のマニュアルページを参照してください。

他の変更点および機能

次に、他の変更点および機能について説明します。

- RFC 2476 で説明しているように、`sendmail` は、ポート 587 上で実行依頼に備えて待機します
- ftp://ftp.sendmail.org で `sendmail` とともに配布しているリリースノートで説明しているように、`XUSR` SMTP コマンドは推奨されていません。 メールユーザーエージェントは、最初のユーザーメッセージの実行依頼時に、RFC 2476 `Message Submission` の使用を開始する必要があります。
- バージョンに関係なく、Sun の構成ファイルを使ってプログラムにパイプされるメッセージには、`Content-Length:` ヘッダーは提供されません。 ただし、このヘッダーは、追加されたメッセージには提供されます。また、バージョンに関係なく、Sun の構成ファイルを使用する一般のメールボックスの配信には提供されます。
- ディスクの容量が少ない場合でも、`sendmail` は接続を受け入れます。ただし、このような場合は、`sendmail` で使用できるコマンドは `ETRN` だけです。
- 別名ファイルのエントリを続けて入力するには、改行文字のすぐ前にバックスラッシュを置きます。
- SMTP 経由でメッセージを送信する際のタイムアウトが変更され、5 分ごとに配信の進捗状況を確認するようになりました。 この変更により、送信できないメッセージを検出できるようになりました。そのため、情報をより迅速に送信したり、タイムアウトになるまで待機するプロセスの数を減らしたりすることができます。
- クラスの内容を他のクラスにコピーするには、次の例のような構文を使用します。

```
C{Dest} $={Source}
```

この例では、クラス $={*Source*} にある全項目が、クラス $={*Dest*} にコピーされます。

- デフォルトでは、マップを省略することはできません。 また、マップに問題が発生すると、エラーメッセージが送信されます。
- クラス P ($=P) のホストまたはドメインは正規化されません。
- オプションに値が関連付けられていない場合には、等号 (=) は、そのオプションの展開に含まれません。
- アドレスの経路は指定できません。たとえば、<@a,@b,@c:user@d> は <user@d> に変換されます。

第 28 章

モデム関連ネットワークサービス (トピック)

第 29 章	PPP の概要情報
第 30 章	PPP の計画情報
第 31 章	ダイアルアップ PPP リンクのセットアップの手順説明
第 32 章	専用回線 PPP リンクのセットアップの手順説明
第 33 章	PPP リンクに対する認証のセットアップの手順説明
第 34 章	DSL 機器を介した PPP リンクをサポートする PPPoE トンネル作成の手順説明
第 35 章	PPP リンクの保守および問題を解決する手順の説明
第 36 章	PPP の使用方法についての詳細なリファレンス情報
第 37 章	前のバージョンの Solaris PPP (asppp) から Solaris PPP 4.0 に移行する際の手順説明
第 38 章	UUCP の背景情報
第 39 章	UUCP のセットアップと障害追跡の手順説明
第 40 章	UUCP データベースファイル、UUCP 構成ファイル、UUCP シェルスクリプト、UUCP 障害追跡情報の参考資料

第 29 章

Solaris PPP 4.0 (概要)

Solaris PPP 4.0 では、ポイントツーポイントプロトコル (PPP) を使用することで、物理的に離れた場所にある 2 つのコンピュータがさまざまな媒体を介して互いに通信することができます。Solaris 9 オペレーティング環境には、基本インストールの一部に Solaris PPP 4.0 が組み込まれています。

この章では Solaris PPP 4.0 について紹介します。ここでは、次の内容を説明します。

Solaris PPP 4.0 の基本

Solaris PPP 4.0 は、TCP/IP プロトコル群に含まれるデータリンクプロトコルとしてポイントツーポイントプロトコル (PPP)を実装しています。PPP は、2 つの端点にあるマシン間でデータを電話回線などの通信媒体を介して転送する方法について記述しています。

PPP は、1990 年代の初期から、通信リンクを介してデータグラムを送信するために幅広く使用されてきたインターネット標準です。PPP 標準は、Internet Engineering Task Force (IETF) のポイントツーポイントワーキンググループによって RFC 1661 に定義されています。PPP は一般に、ホームコンピュータなどのリモートマシンがインターネットサービスプロバイダ (ISP) を呼び出したり、着呼を受信するように構成されている企業サーバーを呼び出したりするときに使用されます。

Solaris PPP 4.0 は、広く普及している Australian National University (ANU) PPP–2.4 に基づいて PPP 標準を実装しています。PPP リンクは非同期と同期の両方をサポートしています。

Solaris PPP 4.0 の互換性

さまざまなバージョンの PPP 標準がインターネットコミュニティで広く使用されています。ANU PPP-2.4 は、Linux、BSD 系統の主要 OS (FreeBSD、OpenBSD、NetBSD)、および Tru64 UNIX で採用されています。

Solaris PPP 4.0 は、Solaris オペレーティング環境で実行されているマシンに ANU PPP-2.4 の高度な構成機能を提供します。Solaris PPP 4.0 が実行されているマシンは、PPP リンクを簡単に設定できます。ANU PPP-2.4 が実行されているマシンだけでなく、PPP 標準が実行されているマシンであればリンク先にすることができます。

ANU ベースの PPP 以外で Solaris PPP 4.0 と正常に相互運用できるものは、次のとおりです。

- Solaris 2.4 から Solaris 8 までのオペレーティング環境で稼働する Solaris PPP (別名 `asppp`)
- Solstice™ PPP 3.0.1
- Windows 98 DUN
- Cisco IOS 12.0 (同期)

使用する Solaris PPP のバージョン

Solaris PPP 4.0 は、Solaris 9 オペレーティング環境がサポートする PPP です。Solaris 9 オペレーティング環境には、 以前の非同期 PPP (`asppp`) ソフトウェアは組み込まれていません。非同期 PPP の構成については、『Solaris 8 System Administrator Collection』(`http://docs.sun.com`) を参照してください。

Solaris PPP 4.0 を使用する理由

現在 `asppp` を使用しているユーザーには、Solaris PPP 4.0 への移行をお勧めします。2 つの Solaris PPP 間の技術的な相違点は次のとおりです。

- 転送モード

  `asppp` は非同期通信だけに対応します。Solaris PPP 4.0 は非同期通信と同期通信の両方に対応します。

- 構成プロセス

  `asppp` の設定には、`asppp.cf` 構成ファイル、3 つの UUCP ファイル、および `ifconfig` コマンドが必要です。さらに、マシンにログインするユーザーのために、あらかじめインタフェースを構成しておく必要があります。

  Solaris PPP 4.0 の設定では、PPP 構成ファイルのオプションを定義するか、オプションを指定して `pppd` コマンドを発行するか、あるいは両者を組み合わせて使用するだけです。Solaris PPP はインタフェースを動的に作成して削除します。各ユーザーのために PPP インタフェースを構成する必要はありません。

- **`asppp`** が提供しない **Solaris PPP 4.0** の機能

- MS-CHAPv1 および MS-CHAPv2 認証
- ADSL ブリッジをサポートする PPP over Ethernet (PPPoE)
- PAM 認証
- プラグインモジュール群
- IPv6 アドレス指定
- Deflate 圧縮または BSD 圧縮を使用するデータ圧縮

Solaris PPP 4.0 のアップグレードパス

既存の `asppp` 構成を Solaris PPP 4.0 に変換する場合は、このリリースが提供する変換スクリプトを使用できます。詳細は、579 ページの「`asppp` から Solaris PPP 4.0 に変換する方法」を参照してください。

PPP の詳細情報

PPP に関する多くの情報は印刷物やオンラインで入手可能です。参考資料のいくつかを以降で示します。

PPP に関する専門技術者向けのリファレンスブック

ANU PPP など、幅広く使用されている PPP については、次の図書を参照してください。

- 『*PPP Design, Implementation, and Debugging*』第 2 版、Carlson, James 著、Addison-Wesley 発行、2000
- 『*Using and Managing PPP*』Sun, Andrew 著、O'Reilly & Associates 発行、1999

PPP に関する Web サイト

PPP の一般的な情報については、次の Web サイトを参照してください。

- `pppd` に関するよくある質問 (FAQ) などについては、Sun Microsystems の Internet Engineering グループが提供する Web サイト (`http://playground.sun.com/pppd`) を参照してください。
- ANU PPP については、Australian National University の PPP リポジトリ (`http://.pserver.samba.org/cgi-bin/cvsweb/ppp/`) を参照してください。
- 技術情報、FAQ、Solaris システム管理、および前バージョンの PPP については、Sun Microsystem のシステム管理者の資源 (`http://www.sun.com/bigadmin/home/index.html`) を参照してください。
- さまざまな PPP のモデム構成とアドバイスについては、Stokely Consulting が提供する Web Project Management & Software Development の Web サイト (`http://www.stokely.com/unix.serial.port.resource/ppp.slip.htm`) を参照してください。

PPP に関する RFC (Requests for Comments)

PPP に関する有用な Internet RFC は次のとおりです。

- RFC 1661 と RFC 1662。PPP プロトコルの主な機能を解説している
- RFC 1334。パスワード認証プロトコル (PAP) とチャレンジハンドシェーク認証プロトコル (CHAP) などの認証プロトコルを解説している
- RFC 2516。PPP over Ethernet (PPPoE) を解説している

PPP RFC のコピーを入手するには、IETF RFC の Web ページ
(`http://www.ietf.org/rfc.html`) で RFC の番号を指定してください。

PPP に関するマニュアルページ

Solaris PPP 4.0 の実装については、次のマニュアルページを参照してください。

- `pppd(1M)`
- `chat(1M)`
- `pppstats(1M)`
- `pppoec(1M)`
- `pppoed(1M)`
- `sppptun(1M)`
- `snoop(1M)`

PPP のマニュアルページについては、`http://docs.sun.com` を参照するか、`man` コマンドを使用してください。

PPP 構成と用語

この節では、PPP 構成とこのマニュアルで使用する用語について説明します。

Solaris PPP 4.0 は次の構成をサポートします。

- スイッチ型のアクセス構成 (ダイアルアップ)
- 固定型の構成 (専用回線)

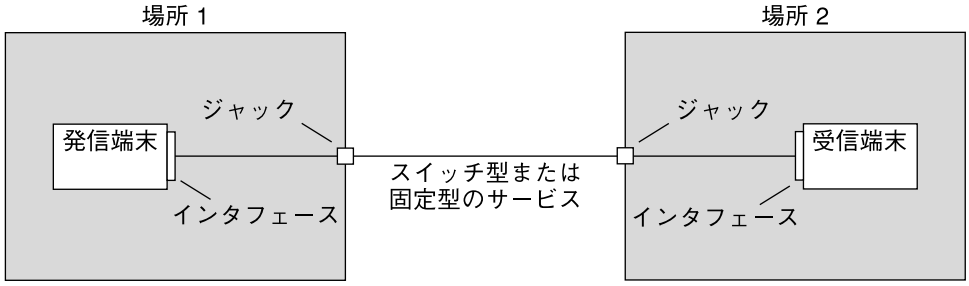


図 **29–1** PPP リンクの構成要素

上図は、基本的な PPP リンクを示しています。リンクの構成要素は、次のようになります。

- 2 つのマシン。通常、ピアと呼ばれ、物理的に互いに離れた場所に配置されている。ピアは、サイトの要件によってパーソナルコンピュータ、エンジニアリングワークステーション、大規模サーバー、商用ルーターなどが考えられる
- 各ピアに対するシリアルインタフェース。Solaris マシンのインタフェースは、構成する PPP が非同期か同期かによって、`cua`、`hih` などが考えられる
- シリアルケーブル、モデム接続などの物理リンク、またはネットワークプロバイダが提供する T1 回線や T3 回線などの専用回線

ダイアルアップ PPP の概要

もっともよく使用される PPP 構成は、ダイアルアップリンクです。ダイアルアップリンクでは、ローカルピアがリモートピアをダイアルアップして接続を確立し、PPP を実行します。ダイアルアッププロセスでは、ローカルピアがリモートピアの電話番号を呼び出してリンクを開始します。

一般的なダイアルアップの使用例では、ユーザーの自宅にあるコンピュータが、着呼を受信するように構成されている ISP 側のピアを呼び出します。別のダイアルアップの使用例では、企業サイトの建物内にあるローカルマシンが PPP リンクを使用して、別の建物内にあるピアにデータを転送します。

このマニュアルでは、ダイアルアップ接続を開始するローカルピアは、ダイアルアップマシンと呼びます。着呼を受信するピアは、ダイアルインサーバーと呼びます。このピアはサーバーと呼ばれますが、ダイアルアウトマシンがターゲットにするマシンに過ぎません。

PPP はクライアントサーバープロトコルではありません。PPP のドキュメントの中には、「クライアント」や「サーバー」という用語を、電話呼び出しによる確立のことを示すために使用しているものがあります。ダイアルインサーバーは、ファイルサーバーやネームサーバーのような真の意味でのサーバーではありません。ダイアルイン

サーバーという用語は、ダイアルインマシンが複数のダイアルアウトマシンにネットワークでのアクセス可能性を「提供」していることから、PPP 用語として幅広く使用されています。それでもダイアルインサーバーは、現実には、ダイアルアウトマシンのターゲットピアにすぎません。

ダイアルアップ PPP リンクの構成要素

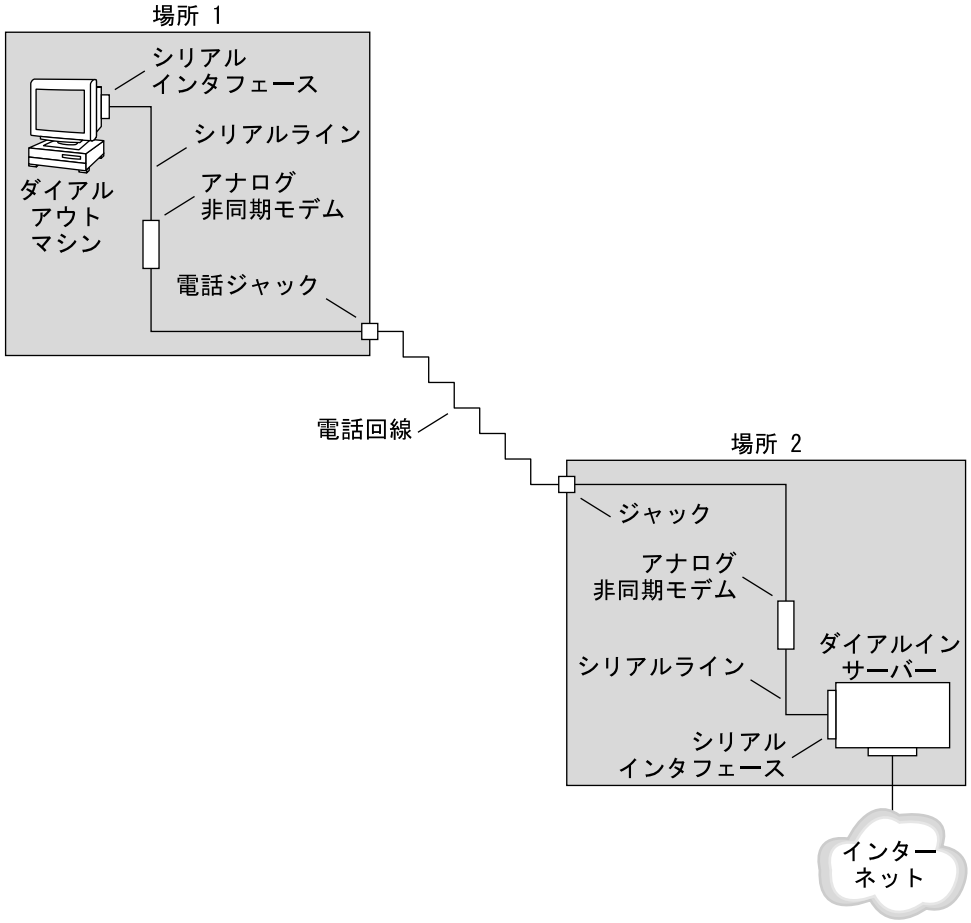


図 **29–2** 基本的なアナログダイアルアップ PPP リンク

リンクのダイアルアウト側 (場所 1) の構成は、次の要素から成ります。

- ダイアルアウトマシン。一般に、個々の家庭のパーソナルコンピュータやワークステーション

- ダイアルアウトマシン上のシリアルインタフェース。/dev/cua/a または /dev/cua/b は、Solaris ソフトウェアが実行されているマシン上で発呼に使用する標準のシリアルインタフェースである
- 電話のジャックに接続される非同期モデムまたは ISDN 端末アダプタ (TA)
- 電話会社の電話回線やサービス

リンクのダイアルイン側 (場所 2) の構成は、次の要素から成ります。

- 電話ネットワークに接続される電話のジャックまたは類似のコネクタ
- 非同期モデムまたは ISDN TA
- ダイアルインサーバー上のシリアルインタフェース。ttya または ttyb は、ダイアルインサーバー上で着呼に使用するシリアルインタフェースである
- ダイアルインサーバー。企業のイントラネットなどのネットワークや ISP のインスタンス内からグローバルインターネットに接続される

ダイアルアウトマシンで ISDN 端末アダプタを使用する

外付けの ISDN TA はモデムよりも高速ですが、両者の構成方法は基本的に同じです。両者の主な相違は chat スクリプト間の構成にあります。ISDN TA の場合、chat スクリプトの記述では、TA の製造元に固有のコマンドが必要になります。ISDN TA 用の chat スクリプトについては、552 ページの「外部 ISDN TA 用 chat スクリプト」を参照してください。

ダイアルアップ通信中の動作

ダイアルアウトとダイアルインの両方のピアにある PPP 構成ファイルには、リンクを設定するための命令群が含まれています。ダイアルアップリンクが開始されると、次のプロセスが発生します。

1. ダイアルアウトマシン上のユーザーまたはプロセスは、pppd コマンドを実行してリンクを開始する
2. ダイアルアウトマシンは PPP 構成ファイルからダイアルインサーバーの電話番号などを読み取り、シリアル回線を介して命令群をモデムに送信する
3. モデムは電話番号をダイアルして、ダイアルインサーバー側のモデムと電話接続を確立する。

   ダイアルアウトマシンは、必要に応じて、ダイアルインサーバーにコマンドを送信し、サーバー側の PPP を呼び出す
4. ダイアルインサーバーに接続されているモデムは、ダイアルアウトマシン側のモデムとリンクのネゴシエーションを開始する。

   ダイアルアウトマシンが、モデムとダイアルインサーバーに送信する一連のテキスト文字列は、*chat* スクリプトと呼ばれるファイルに格納されている
5. モデム同士のネゴシエーションが完了すると、ダイアルアウトマシン側のモデムは「CONNECT」を通知する

6. 両方のピア側の PPP は確立フェーズに入る。このフェーズでは、リンク制御プロトコル (LCP) が基本的なリンクパラメータと認証の使用をネゴシエートする
7. ピアは、必要に応じて、互いを認証する
8. PPP のネットワーク制御プロトコル (NCP) は、IPv4 や IPv6 などのネットワークプロトコルの使用をネゴシエートする

ダイアルアウトマシン側のユーザーは、ダイアルインサーバーから到達可能なネットワーク上のホストに `rlogin`、`telnet`、または類似のコマンドを実行できます。

専用回線 PPP の概要

固定型の専用回線の PPP 構成は、リンクで接続された 2 つのピアから成っています。リンクは、プロバイダからリースされたスイッチ型または非スイッチ型のデジタルサービスで構成されています。Solaris PPP 4.0 は、全二重でポイントツーポイントの専用回線媒体を介して動作します。通常、会社では、ネットワークプロバイダから専用リンクをレンタルして、ISP または他のリモートサイトに接続します。

ダイアルアップリンクと専用回線リンクの比較

ダイアルアップと専用回線のリンクはともに、通信媒体で接続されている 2 つのピアから成っています。次の表は、2 つのリンクタイプの相違をまとめています。

専用回線	ダイアルアップ回線
システム管理者または電源障害による電源切断がないかぎり常時接続されている	ユーザーがリモートピアを呼び出そうとするとき開始される
同期通信を使用する	非同期通信を使用する
プロバイダからのレンタル	既存の電話回線を使用する
同期装置を必要とする	低コストのモデムを使用する
専用インタフェースを必要とする	通常のコンピュータに組み込まれている標準のシリアルインタフェースを使用する

専用回線 PPP リンクの構成要素

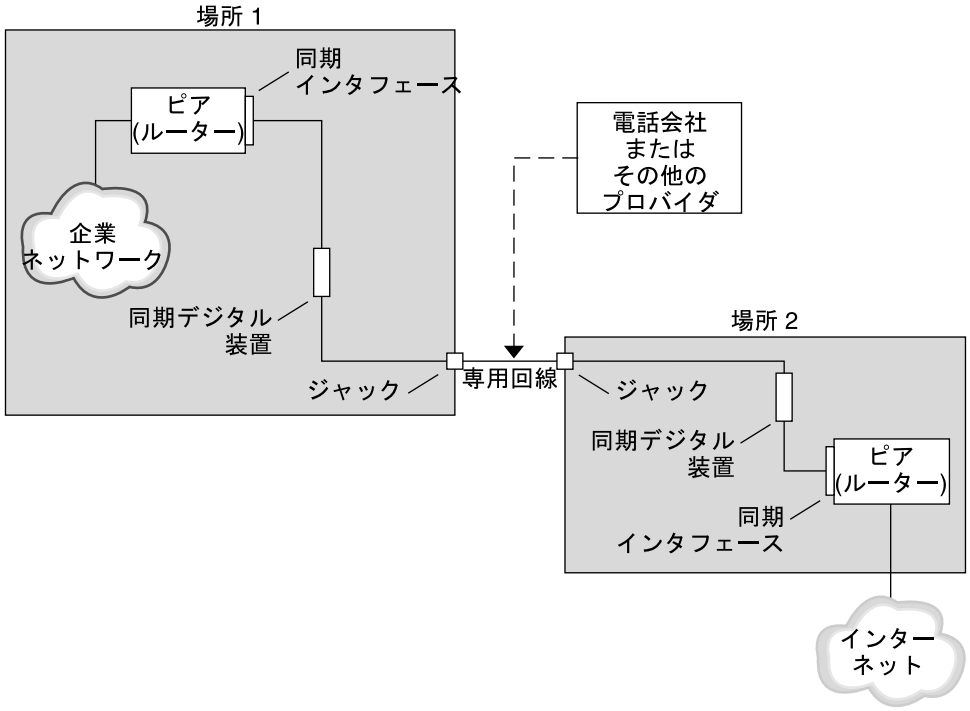


図 **29–3** 専用回線の基本的な構成

専用回線リンクの構成要素は次のとおりです。

- **2** つのピア。リンクの両端に 1 つずつ存在する。各ピアは、ワークステーションかサーバーである。通常ピアは、ネットワークまたはインターネットともう一方の側のピアとの間のルーターとして機能する
- 同期インタフェース。各ピア上に存在する。Solaris ソフトウェアが実行されている一部のマシンは、専用回線に接続するために、HSI/S などの同期インタフェースカードを購入する必要がある。UltraSPARC™ ワークステーションなどのマシンには同期インタフェースが内蔵されている
- **CSU/DSU** 同期デジタル装置。各ピア上に存在し、同期ポートを専用回線に接続する。

  現場の事情によって、CSU は DSU に組み込まれていたり、個人で所有していたり、プロバイダからリースしていたりする。DSU はマシンに標準の同期シリアルインタフェースを提供する。フレームリレーを使用する場合、フレームリレーアクセスデバイス (FRAD) が、シリアルインタフェースに適合するように調整する
- 専用回線。スイッチ型または非スイッチ型のデジタルサービスを提供する。専用回線のデジタルサービスには、SONET/SDH、Frame Relay PVC、T1 などがある

注 – SONET は、オクテット同期リンクと呼ばれています。PPP は、SONET 回線で非同期フレームと類似のフレーム機構を使用します。PPP は予想されるビット同期プロトコルは使用していません。

専用回線通信中の動作

ほとんどのタイプの専用回線では、ピアは互いにダイアルすることはありません。会社では専用回線サービスを購入して、2 つの定められた場所の間を明示的に接続します。場合によって、専用回線の各端にある 2 つのピアは同じ会社でも物理的に離れた場所に存在することもあります。別の事例では、会社が、ISP に接続されている専用回線上にルーターを設定している場合があります。

専用回線の固定型のリンクは設定が簡単ですが、ダイアルアップリンクほどは普及していません。固定型のリンクは chat スクリプトを必要としません。専用回線の場合、両方のピアは互いを知っているので、認証を使用しないのが普通です。2 つのピアがリンクを介して PPP を開始すると、専用回線に障害が発生したり、どちらかのピアが明示的にリンクを終了したりしないかぎり、PPP はアクティブな状態を続けます。

Solaris PPP 4.0 が実行されている専用回線上のピアは、ダイアルアップリンクを定義する構成ファイルとほぼ同じものを使用します。

専用回線を介した通信を開始する場合、次のプロセスが発生します。

1. 各ピアマシンは、`pppd` コマンドを起動プロセスや管理スクリプトの一部として実行する
2. 両方のピアは自分の PPP 構成ファイルを読み取る
3. 両方のピアは通信パラメータをネゴシエートする
4. IP リンクが確立される

PPP 認証

認証は、要求しているのがユーザー本人であることを確認するためのプロセスです。従来の UNIX のログインの流れは、次のように簡単な認証形式です。

1. `login` コマンドを入力すると、ユーザーに名前とパスワードの入力を求めるプロンプトが表示される
2. 次に `login` は、ユーザーを認証するために、入力された名前とパスワードをパスワードデータベースから探そうとする
3. データベース中にユーザー名とパスワードが存在する場合、ユーザーは認証されて、システムへのアクセスが許可される。データベース中にユーザー名とパスワードが存在しない場合、ユーザーはシステムへのアクセスを拒否される

デフォルトでは、Solaris PPP 4.0 は、デフォルトの経路が指定されていないマシン上では認証を要求しません。したがって、デフォルトの経路が指定されていないローカルマシンはリモート呼び出しを認証しません。逆に、マシンにデフォルトの経路が定義されていれば、Solaris PPP 4.0 は、デフォルトでリモート呼び出しを認証します。

必要な場合、自分のマシンに PPP リンクを設定しようとしている呼び出し側の識別情報を、PPP 認証プロトコルを使って確認できます。逆に、呼び出し側を認証するピアをローカルマシンが呼び出す必要がある場合は、PPP 認証情報をローカルマシンに構成しておく必要があります。

認証する側と認証される側

PPP リンク上の呼び出し側マシンは、リモートピアに対して識別情報を示す必要があるので、認証される側とみなされます。ピアは、認証する側とみなされます。認証する側は、呼び出し側の識別情報をセキュリティプロトコル用の適切な PPP ファイルから探し、その呼び出し側を認証したり認証を拒否したりします。

多くの場合、PPP 認証をダイアルアップリンクに構成します。呼び出しが開始されると、ダイアルアウトマシンが認証される側になります。ダイアルインサーバーは認証する側になります。サーバーはデータベースを秘密ファイルの形式で保持します。このデータベースには、サーバーに PPP リンクを設定する許可が与えられているすべてのユーザーが記述されています。許可が与えられているユーザーは信頼できる呼び出し側とみなされます。

一部のダイアルアウトマシンには、ダイアルアウトマシンの呼び出しに対する応答でリモートピアに認証情報の提供を要求するものがあります。このような場合は、役割が逆転し、リモートピアは認証される側になり、ダイアルアウトマシンは認証する側になります。

注 – PPP 4.0 は専用回線でピアによる認証を禁止していませんが、通常は使用しません。専用回線規約では、回線の両端に存在する両者が互いをよく知っており、信頼していることが特徴となっています。しかし、PPP 認証は管理が簡単なので、専用回線にも認証を実装することをまじめに検討する必要があります。

PPP の認証プロトコル

PPP の認証プロトコルは、パスワード認証プロトコル (PAP) とチャレンジハンドシェーク認証プロトコル (CHAP) です。 各プロトコルは、ローカルマシンにリンクする許可が与えられている各呼び出し側に対して、識別情報が格納された秘密データベースやセキュリティ資格情報を使用します。PAP については、555 ページの「パスワード認証プロトコル (PAP)」を参照してください。CHAP については、559 ページの「チャレンジハンドシェーク認証プロトコル (CHAP)」を参照してください。

PPP 認証を使用する理由

PPP リンクでの認証は任意です。また、認証では、ピアが信頼されていることは確認しますが、データに機密保護を提供していません。機密保護では、IPsec、PGP、SSL、Solaris セキュアシェルなどの暗号化ソフトウェアを使用します。

注 – Solaris PPP 4.0 は、RFC 1968 に記述されている PPP Encryption Control Protocol (ECP) を実装していません。

次の場合に、PPP 認証の実装を検討してください。

- 会社が、公衆電話交換網を介してユーザーから着呼を受け取る
- 会社のファイアウォールを介してネットワークにアクセスする場合やセキュリティで保護されたトランザクションに関係する場合に、会社のセキュリティポリシーでリモートユーザーに認証資格情報の提供を要求している
- 標準の UNIX パスワードデータベース (`/etc/passwd`、NIS、NIS+、LDAP、または PAM) と照合して呼び出し側を認証したい。この場合は PAP 認証を使用する
- 会社のダイアルインサーバーがネットワークのインターネット接続も提供するこの場合は PAP 認証を使用する
- シリアル回線が、リンクのどちらか端にあるネットワークやマシン上のパスワードデータベースよりもセキュリティの保護が弱いこの場合は CHAP 認証を使用する

PPPoE による DSL ユーザーのサポート

多くのネットワークプロバイダと自宅で仕事している個人は、デジタル加入者回線 (DSL) 技術を使用して、高速なネットワークアクセスを実現します。DSL ユーザーをサポートするために、Solaris PPP 4.0 は PPP over Ethernet (PPPoE) 機能を組み込んでいます。PPPoE 技術を使用することで、複数のホストが 1 つの Ethernet リンクを介して 1 つ以上の地点に PPP セッションを実行できます。

次の場合に、PPPoE を使用する必要があります。

- DSL ユーザー (自分自身も含む場合もある) をサポートする。DSL サービスプロバイダは、DSL 回線を介してサービスを受け取るために、ユーザーに PPPoE トンネルの構成を要求することがある
- サイトが、顧客に PPPoE を提供する ISP である

この節では、PPPoE に関連する用語と基本的な PPPoE 技術の概要について説明します。

PPPoE の概要

PPPoE は、RedBack Networks が生み出した独自のプロトコルです。PPPoE は、別バージョンの標準 PPP ではなく検出プロトコルです。PPPoE のシナリオでは、最初に PPP 通信を開始するマシンが、PPPoE を実行しているピアを検出する必要があります。PPPoE プロトコルは、Ethernet ブロードキャストパケットを使ってピアを検出します。

検出プロセスを終了したら、PPPoE は、開始したホスト (*PPPoE* クライアント) からピア (*PPPoE* アクセスサーバー) まで Ethernet ベースのトンネルを設定します。トンネリングとは、あるプロトコルを、TCP/IP プロトコルスタック上で同等か上位の位置にある別のプロトコルで実行する方法です。PPPoE を使用して、Solaris PPP 4.0 は PPP に Ethernet IEEE 802.2 を介したトンネルを作成します。PPP と Ethernet IEEE 802.2 はともにデータリンクプロトコルです。設定された PPP 接続は、PPPoE クライアントとアクセスサーバーの間で専用リンクのように動作します。PPPoE については、564 ページの「DSL サポート用の PPPoE トンネルの作成」を参照してください。

PPPoE の構成要素

次の図に示すように、PPPoE 構成には、消費者、電話会社、およびサービスプロバイダという 3 つの関係者が存在します。

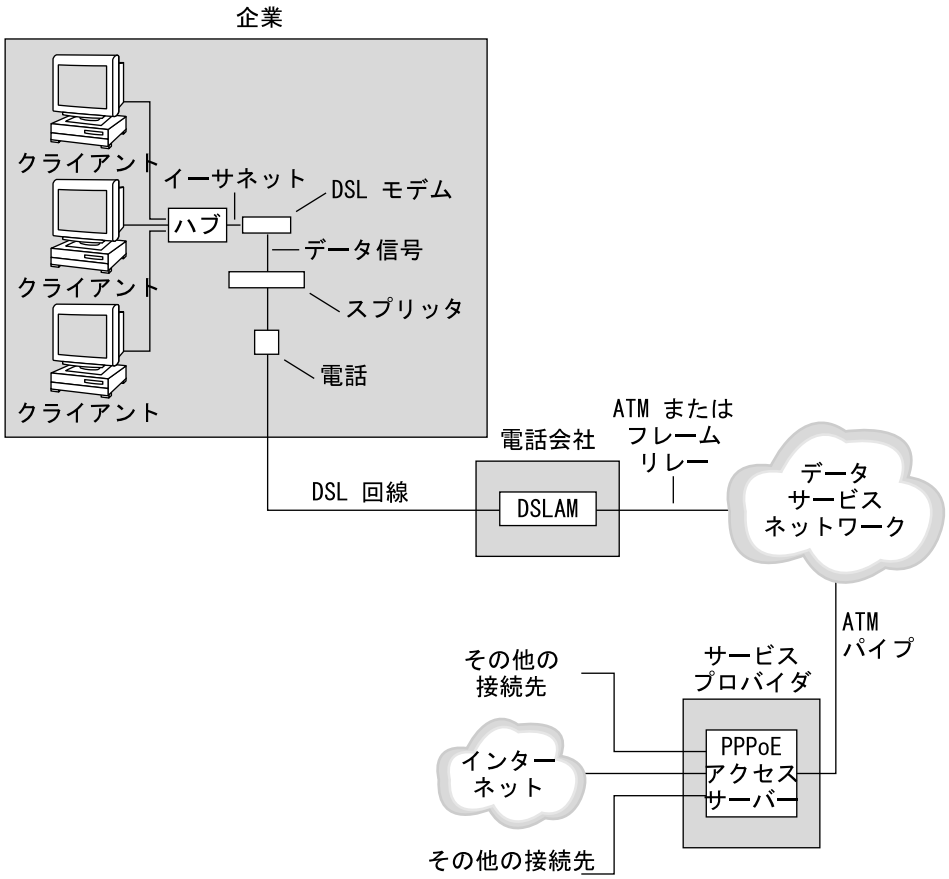


図 **29–4** PPPoE トンネル内の関係者

PPPoE の消費者

システム管理者として、消費者の PPPoE 構成を助けることがあります。PPPoE 消費者の一般的なタイプは、DSL 回線を介して PPPoE を実行する個人です。別の PPPoE 消費者は、上図に示すように、従業員が PPPoE トンネルを実行できるように DSL 回線を購入する会社です。

企業消費者が PPPoE を使用する主な理由は、高速の DSL 機器を介して多くのホストに PPP 通信を提供するためです。通常、単独の PPPoE クライアントは、個人で *DSL* モデムを持ちます。また、ハブに接続されているクライアントのグループは、Ethernet 回線によって同じハブに接続されている DSL モデムを共有することがあります。

注 – DSL 機器は技術的にはモデムではなくブリッジです。ただし、実際にはこれらのデバイスをモデムと呼んでいるので、このマニュアルでは、「DSL モデム」という用語を使用します。

PPPoE が実行を開始したら、DSL モデムに接続されている Ethernet 回線上のトンネルを介して PPP を実行します。その回線はスプリッタに接続され、スプリッタは電話回線に接続しています。

電話会社の PPPoE

PPPoE のシナリオでは、電話会社は中間に位置します。 電話会社は、電話回線を介して受信する信号を、デジタル加入者線アクセスマルチプレクサ *(DSLAM)* と呼ばれるデバイスを使って分割します。 DSLAM は分割した信号を別の線、電話サービス用アナログ線、および PPPoE 用デジタル線に送り出します。デジタル線は ATM データネットワークを介してトンネルを DSLAM から ISP まで延長します。

サービスプロバイダの PPPoE

ISP は、ATM データネットワークから渡される PPPoE 転送をブリッジを介して受信します。ISP では、PPPoE が実行されているアクセスサーバーが PPP リンクのピアとして機能します。アクセスサーバーは、図 29–2 で紹介したダイアルインサーバーと機能的に類似していますが、アクセスサーバーがモデムを使用しない点が異なります。アクセスサーバーは、個々の PPPoE セッションをインターネットアクセスなどの通常の IP トラフィックに変換します。

ISP のシステム管理者は、アクセスサーバーの構成と維持を行います。

PPPoE トンネルのセキュリティ

PPPoE トンネルは最初からセキュリティ対策が行われていません。PAP または CHAP を使用することで、トンネルを介して実行している PPP リンクにユーザー認証を提供できます。

第 30 章

PPP リンクの計画 (手順)

PPP リンクの設定には、作業計画や PPP と無関係な作業など、さまざまな個別の作業が含まれています。この章では、もっとも一般的な PPP リンク、認証、および PPPoE を計画する方法について説明します。

第 30 章に続く各章では、特定リンクの設定方法について構成例を使って説明します。これらの構成例はこの章で紹介します。

ここでは、次の内容を説明します。

全体的な PPP 計画 (作業マップ)

PPP では、実際にリンクを設定する前に作業計画を立てる必要があります。さらに、PPPoE トンネルを使用する場合は、まず PPP リンクを設定し、それからトンネルを提供する必要があります。次の作業マップは、この章で説明する大規模な作業計画を示しています。構成するリンクタイプによっては、一般的な作業だけで十分な場合があります。また、リンク、認証、および PPPoE の各作業が必要になる場合もあります。

表 **30–1** PPP 計画 (作業マップ)

作業	説明	参照先
ダイアルアップ PPP リンクを計画する	ダイアルアウトマシンまたはダイアルインサーバーの設定に必要な情報を収集する	454 ページの「ダイアルアップ PPP リンクの計画」

表 **30–1** PPP 計画 (作業マップ)　(続き)

作業	説明	参照先
専用回線リンクを計画する	専用回線にクライアントを設定するための必要情報を収集する	457 ページの「専用回線リンクの計画」
PPP リンクの認証を計画する	PPP リンクに PAP 認証または CHAP 認証を構成するための必要情報を収集する	460 ページの「リンクへの認証計画」
PPPoE トンネルを計画する	PPP リンクが実行できる PPPoE トンネルを設定するための必要情報を収集する	465 ページの「PPPoE トンネルを介した DSL サポートの計画」

ダイアルアップ PPP リンクの計画

ダイアルアップリンクはもっともよく使用される PPP リンクです。この節では、次の内容について説明します。

- ダイアルアップリンクの計画情報
- 第 31 章で使用されるリンク例の説明

通常は、マシンをダイアルアップ PPP リンク、ダイアルアウトマシン、またはダイアルインサーバーの一方の端に構成するだけです。ダイアルアップ PPP の概要については、441 ページの「ダイアルアップ PPP の概要」を参照してください。

ダイアルアウトマシンを設定する前に

ダイアルアウトマシンを構成する前に、次の表に示されている情報を収集します。

注 – この節の計画情報には、認証や PPPoE について収集する情報は含まれていません。 認証計画については、460 ページの「リンクへの認証計画」を参照してください。 PPPoE 計画については、465 ページの「PPPoE トンネルを介した DSL サポートの計画」を参照してください。

表 **30–2** ダイアルアウトマシンの情報

情報	作業
最大モデム速度	モデムの製造元が提供するマニュアルを参照する
モデム接続コマンド (AT コマンド)	モデムの製造元が提供するマニュアルを参照する

表 **30–2** ダイアルアウトマシンの情報　(続き)

情報	作業
リンクの一方の端で使用するダイアルインサーバーの名前	ダイアルインサーバーの識別が簡単な名前を作成する
ダイアルインサーバーで必要なログインシーケンス	ダイアルインサーバーの管理者に問い合わせるか、ダイアルインサーバーが ISP 側に存在すれば、ISP のマニュアルを参照する

ダイアルインサーバーを設定する前に、次の表に示されている情報を収集します。

ダイアルインサーバーを設定する前に

ダイアルインサーバーを構成する前に、次の表に示されている情報を収集します。

注 – この節の計画情報には、認証や PPPoE について収集する情報は含まれていません。 認証計画については、460 ページの「リンクへの認証計画」を参照してください。 PPPoE 計画については、465 ページの「PPPoE トンネルを介した DSL サポートの計画」を参照してください。

表 **30–3** ダイアルインサーバーの情報

情報	作業
最大モデム速度	モデムの製造元が提供するマニュアルを参照する
ダイアルインサーバーの呼び出しが許可されている人のユーザー名	479 ページの「ダイアルインサーバーのユーザーを構成する方法」 で説明するようなホームディレクトリを設定する前に、予想されるユーザーの名前を入手する
PPP 通信の専用 IP アドレス	会社での IP アドレスの委譲に責任を持つ担当者からアドレスを入手する

例 — ダイアルアップ PPP の構成

第 31 章 で紹介する作業では、従業員が週に 2、3 日自宅で作業することを許している小さい企業の要件を実施します。一部の従業員は、ホームマシンに Solaris オペレーティング環境が必要になります。また、社内イントラネット上にある作業マシンにリモートログインすることも必要になります。

作業では、次の機能を持つ基本的なダイアルアップリンクを設定します。

- ダイアルアウトマシンが、社内イントラネットを呼び出す従業員の自宅に存在する
- ダイアルインサーバーは、従業員からの着呼を受信するように構成された社内イントラネット上のマシンである

- UNIX スタイルのログインを使用して、ダイアルアウトマシンを認証する。Solaris PPP 4.0 の強力な認証方法は、この会社のセキュリティポリシーには必要ない

次の図は、第 31 章で設定されているリンクを示します。

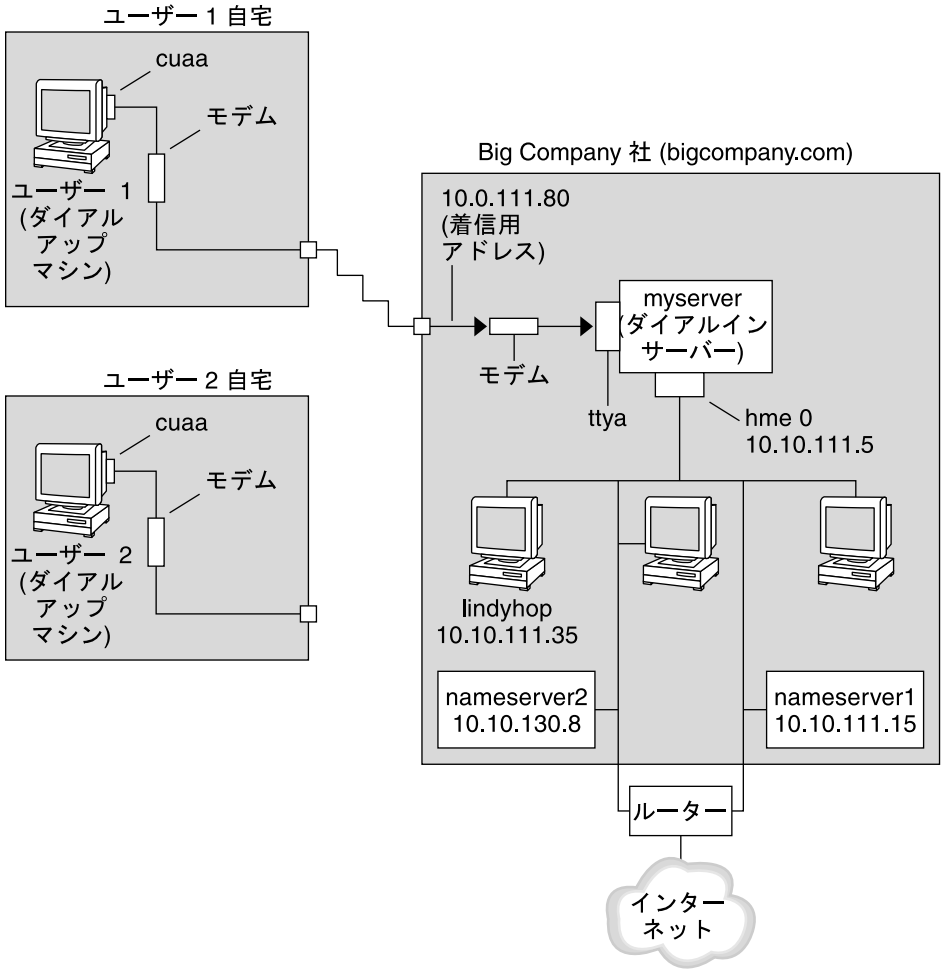


図 **30–1** ダイアルアップリンクの例

この図では、リモートホストが電話回線上のモデルを介して Big Company 社のイントラネットにダイアルアウトしています。もう一台のホストが Big Company 社にダイアルアウトするように構成されていますが、現在アクティブではありません。Big Company 社のダイアルインサーバーに接続されているモデムによって、リモートユーザーからの呼び出しに 1 つずつ応答しています。PPP 接続はピア間で確立しています。ダイアルアウトマシンは、イントラネット上のホストマシンにリモートログインできます。

ダイアルアップ PPP の詳細情報に進む手順

作業	参照先
ダイアルアウトマシンを設定する	表 31–2
ダイアルインマシンを設定する	表 31–4
ダイアルアップリンクの概要を把握する	441 ページの「ダイアルアップ PPP の概要」
PPP のファイルとコマンドについて理解する	533 ページの「ファイルおよびコマンド行での PPP オプションの使用」

専用回線リンクの計画

専用回線リンクの設定では、プロバイダからリースしているスイッチ型または非スイッチ型サービスの一方の端にピアを構成する必要があります。

この節では、次の内容について説明します。

- 専用回線リンクの計画情報
- 図 30–2 に示されているリンク例の説明

専用回線リンクの概要については、444 ページの「専用回線 PPP の概要」を参照してください。 専用回線の設定作業については、第 32 章 を参照してください。

専用回線リンクを設定する前に

会社がネットワークプロバイダから専用回線リンクをレンタルしている場合は、リンクの自分側の端だけにシステムを構成します。リンクのもう一方の端にあるピアは、別の管理者が維持しています。この管理者は、会社から離れた場所にいるシステム管理者か、ISP 側のシステム管理者のどちらかです。

専用回線リンクに必要なハードウェア

リンク媒体自身の他に、リンクの端には次のハードウェアが必要です。

- システム用の同期インタフェース
- 自分の同期装置 (CSU/DSU)
- 自分のシステム

一部のネットワークプロバイダでは、顧客宅内機器 (CPE) として、ルーター、同期インタフェース、および CSU/DSU が必要です。ただし、必要な機器は、プロバイダや国別の政府規制によって変わります。ネットワークプロバイダでは、必要な装置で専用回線と共に提供されないものは、それに関する情報を提供しています。

専用回線を収集するための情報

自分の側にある専用回線の端にピアを構成する前に、次の表に示されている項目や情報を収集する必要があります。

表 **30–4** 専用回線リンクの計画

情報	作業
インタフェースのデバイス名	インタフェースカードのマニュアルを参照する
同期インタフェースカードの構成手順	インタフェースカードのマニュアルを参照する。この情報は、HSI/S インタフェースを構成する場合に必要である。他のタイプのインタフェースカードでは、構成する必要がない場合がある
(任意) リモートピアの IP アドレス	サービスプロバイダのマニュアルを参照するか、リモートピアのシステム管理者に問い合わせる。 この情報は、2 つのピア間で IP アドレスがネゴシエートされない場合にだけ必要である
(任意) リモートピアの名前	サービスプロバイダのマニュアルを参照するか、リモートピアのシステム管理者に問い合わせる
(任意) リンクの速度	サービスプロバイダのマニュアルを参照するか、リモートピアのシステム管理者に問い合わせる
(任意) リモートピアで使用する圧縮	サービスプロバイダのマニュアルを参照するか、リモートピアのシステム管理者に問い合わせる

例 — 専用回線リンクの構成

第 32 章の作業は、LocalCorp 社と呼ばれる中規模会社で従業員がインターネットにアクセスできるように、専用回線リンクの構成を実装する方法を示しています。現在、従業員のコンピュータは、会社の私設イントラネットに接続されています。

LocalCorp 社では、高速なトランザクションとイントラネット上の多くの資源に迅速にアクセスすることが必要となっています。LocalCorp 社は、サービスプロバイダの Far ISP 社との間に専用回線を設定する契約を結びます。これにより、LocalCorp 社は電話会社の Phone East 社から T1 回線をリースします。Phone East 社は LocalCorp 社と Far ISP 社との間に専用回線を設置し、LocalCorp 社に構成済みの CSU/DSU を提供します。

作業では、次の特徴を持つ専用回線リンクを設定します。

- LocalCorp 社はシステムをゲートウェイルーターとして設定する。これにより、パケットは専用回線を介してインターネット上のホストに転送される
- Far ISP 社でも顧客からの専用回線を接続するルーターとしてピアを設定する

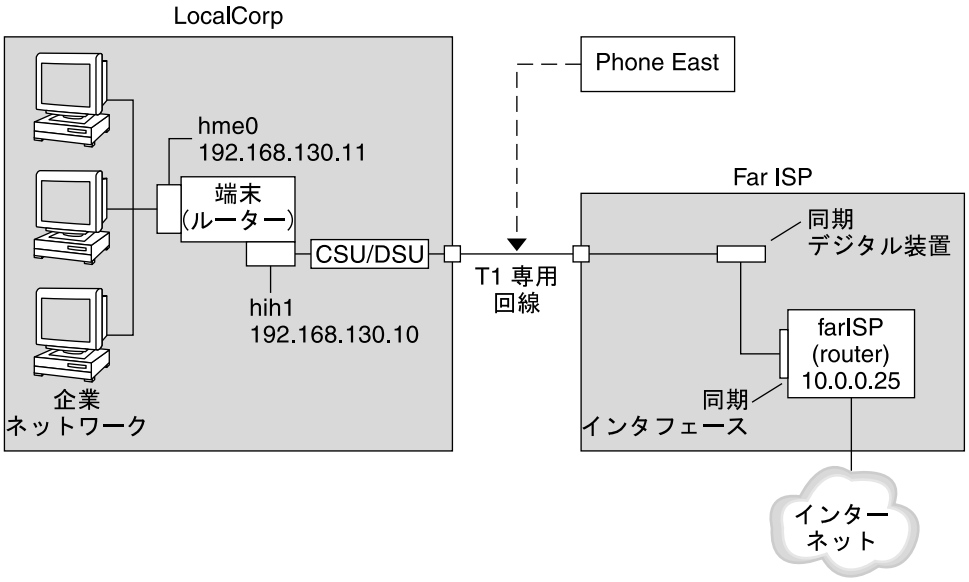


図 **30–2** 専用回線の構成例

この図では、LocalCorp 社側の PPP に設定されているマシンは、`hme0` インタフェースを介して社内イントラネットに接続されているルーターを示しています。さらにマシンは、HSI/S インタフェース (hih1) を介して CSU/DSU デジタル装置に接続されています。CSU/DSU は設置された専用回線に接続しています。LocalCorp 社の管理者が HSI/S インタフェースと PPP ファイルの構成を終了した後で、`/etc/init.d/pppd start` コマンドを入力すると、LocalCorp 社と Far ISP 社間でリンクが開始されます。

専用回線の詳細情報

作業	参照先
専用回線にクライアントを設定する	第 32 章
専用回線の概要を把握する	444 ページの「専用回線 PPP の概要」

リンクへの認証計画

この節では、PPP リンク上で認証を行うための計画情報を提供します。第 33 章は、自分のサイトで PPP 認証を実装するための作業を示しています。

PPP には、PAP と CHAP の 2 種類の認証があります。PAP の詳細は、555 ページの「パスワード認証プロトコル (PAP)」を参照してください。CHAP の詳細は、559 ページの「チャレンジハンドシェーク認証プロトコル (CHAP)」を参照してください。

認証をリンクに設定する前に、自分のサイトのセキュリティポリシーに最適な認証プロトコルを選択する必要があります。認証プロトコルの選択が終了したら、ダイアルインマシンまたは呼び出し側のダイアルアウトマシンまたは両方のマシンに秘密ファイルと PPP 構成ファイルを設定します。自分のサイトに最適な認証プロトコルを選択するには、448 ページの「PPP 認証を使用する理由」を参照してください。

この節では、次の内容について説明します。

- PAP 認証と CHAP 認証の計画情報
- 図 30–3 と 図 30–4 に示されている認証事例の説明

認証の設定作業については、第 33 章を参照してください。

PPP 認証を設定する前に

自分のサイトに認証を設定することを必須として PPP の全体計画に組み込む必要があります。認証を実装する前に、ハードウェアの組み立てや、ソフトウェアの構成、リンクの動作確認を行う必要があります。

表 **30–5** 認証構成の前提条件

情報	参照先
ダイアルアップリンクの構成作業	第 31 章
リンクのテスト作業	第 35 章
サイトのセキュリティ要件	会社のセキュリティポリシーポリシーを設定していなければ、PPP 認証の設定を機にセキュリティポリシーを設定する
自分のサイトに PAP または CHAP を選択する場合のヒント	448 ページの「PPP 認証を使用する理由」。これらのプロトコルについては、555 ページの「接続時の呼び出し元の認証」を参照してください。

例 — PPP の認証構成

この節では、第 33 章 の手順で使用されている認証事例について説明します。

例 — PAP 認証による構成

492 ページの「PAP 認証の設定」での作業は、PPP リンク上で PAP 認証を設定する方法を示しています。手順では、455 ページの「例 — ダイアルアップ PPP の構成」で紹介した架空の Big Company 社の PAP 事例を使用します。

Big Company 社では、自社のユーザーが自宅で仕事できるようにしたいと考えています。システム管理者は、ダイアルインサーバーに接続するシリアル回線にセキュリティ対策をしたいと考えています。NIS パスワードデータベースを使用する UNIX スタイルのログインは、これまで BigCompany 社のネットワークで問題なく機能を果たしてきました。システム管理者は、PPP リンクを介してネットワークに進入してくる呼び出しに UNIX スタイルの認証機構を設定したいと考えています。その結果、システム管理者は PAP 認証を使用する次のシナリオを実装します。

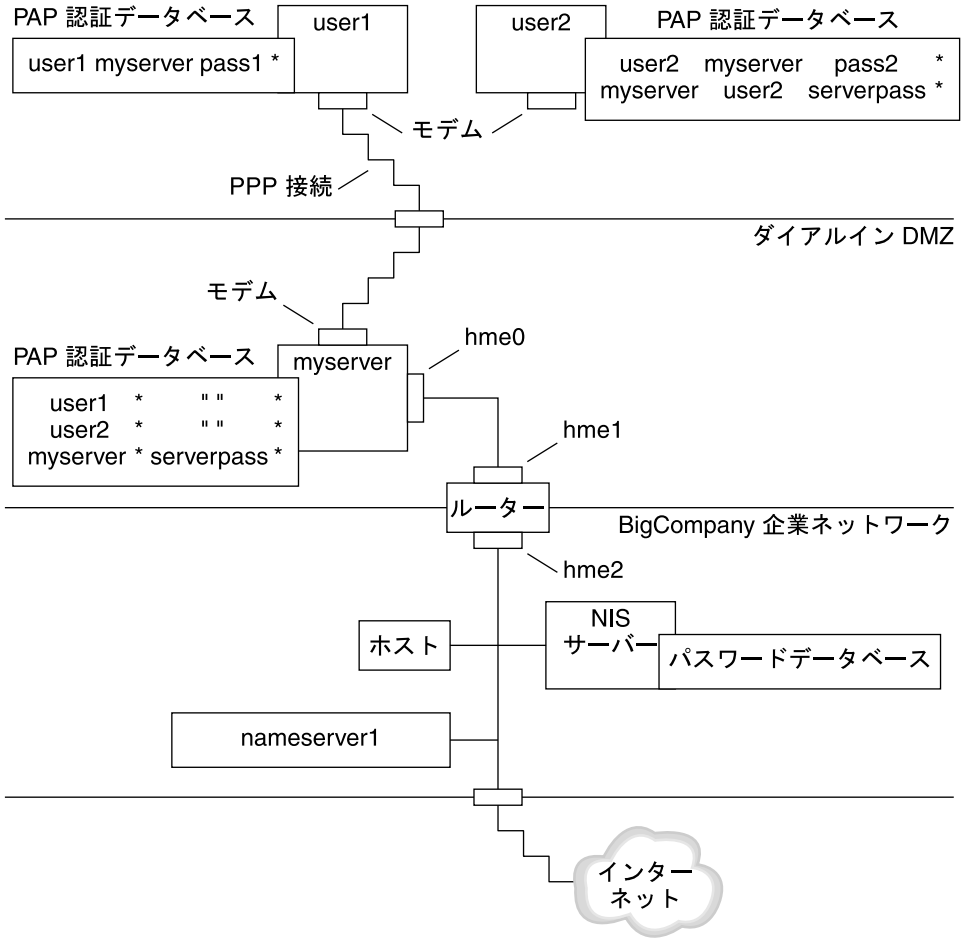


図 **30–3** 例 — PAP 認証のシナリオ (自宅で仕事する)

システム管理者は専用のダイアルイン DMZ を作成します。これは、ルーターによって会社のネットワークの後方部と分離されています。用語の DMZ は、軍隊用語の非武装地帯から来ています。DMZ はセキュリティ目的のために分離されたネットワークです。通常、DMZ には、Web サーバー、匿名 (anonymous) ftp サーバー、データベース、モデムサーバーなど、会社が一般に公開する資源が含まれています。ネットワーク設計者は通常、DMZ をファイアウォールと会社のインターネット接続の中間に設置します。

図 30–3 に示すように、DMZ に存在するのは、ダイアルインサーバーの `myserver` とルーターだけです。ダイアルインサーバーはリンクの設定時に、呼び出し側に PAP 資格 (ユーザー名とパスワードを含む) の提出を要求します。さらに、ダイアルインサー

バーは PAP の `login` オプションも使用します。したがって、呼び出し側の PAP のユーザー名とパスワードは、ダイアルインサーバーのパスワードデータベースにある UNIX のユーザー名とパスワードに正確に一致する必要があります。

PPP リンクが設定されたら、呼び出し側のパケットはルーターに転送されます。ルーターはパケットを会社のネットワーク上かインターネット上の宛先に転送します。

例 — CHAP 認証による構成

500 ページの「CHAP 認証の設定」 での作業は、CHAP 認証の設定方法を示しています。手順では、458 ページの「例 — 専用回線リンクの構成」で紹介した架空の LocalCorp 社の CHAP 事例を使用します。

LocalCorp 社は、ISP の専用回線を介してインターネットに接続できます。LocalCorp 社では、重いネットワークトラフィックが発生するので、社内のテクニカルサポート部に独立した私設ネットワークが必要になっています。部署のフィールドエンジニアは、問題解決のための情報を入手するために遠隔地からテクニカルサポートのネットワークに頻繁にアクセスする必要があります。私設ネットワークのデータベースに格納されている機密情報を保護するには、リモートでの呼び出し側にログインの許可が与えられる前に、それらを認証する必要があります。

したがって、システム管理者は、ダイアルアップ PPP 構成に次の CHAP 認証シナリオを実装します。

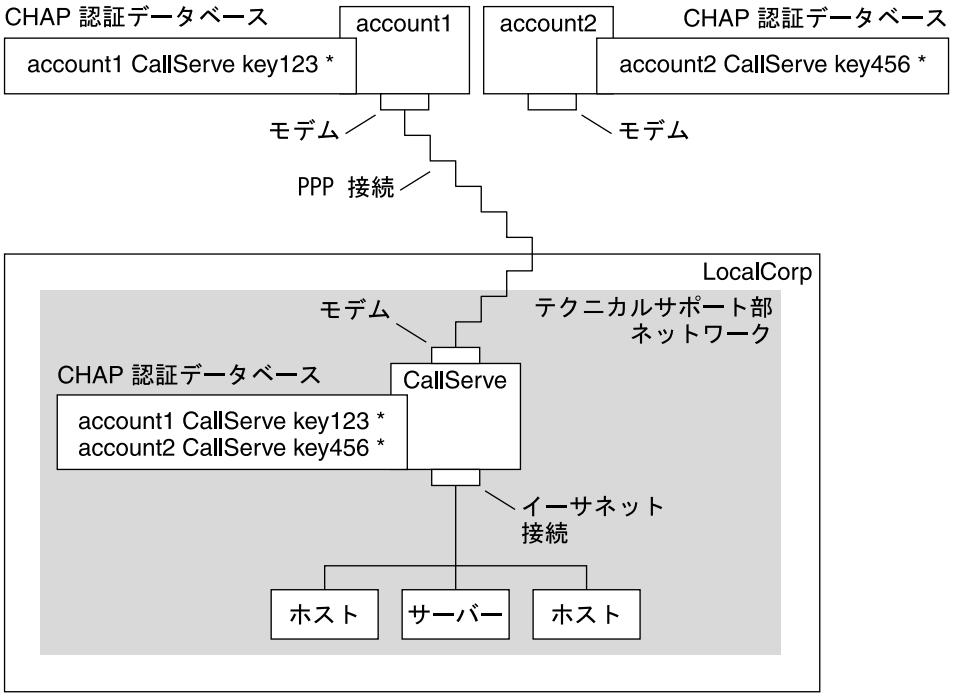


図 **30–4** 例 — CHAP 認証シナリオ (私設ネットワークを呼び出す)

テクニカルサポート部のネットワークから外部世界にリンクするのは、PPP リンクのダイアルインサーバー側の端に接続しているシリアル回線だけです。システム管理者は、各フィールドサービスエンジニアが所持する PPP 用ラップトップコンピュータを CHAP シークレットなどを組み込んだ CHAP セキュリティで構成します。ダイアルインサーバー上の CHAP シークレットデータベースには、テクニカルサポート内のネットワークに対する呼び出しが許されているすべてのマシンの CHAP 資格が含まれています。

認証の詳細情報

作業	参照先
PAP 認証を設定する	492 ページの「PAP 認証の設定」
CHAP 認証を設定する	500 ページの「CHAP 認証の設定」
PPP 認証の詳細を理解する	555 ページの「接続時の呼び出し元の認証」 と `pppd(1M)` マニュアルページ

PPPoE トンネルを介した DSL サポートの計画

一部の DSL プロバイダは、プロバイダの DSL 回線と高速のデジタルネットワーク上で PPP を実行するために、ユーザーのサイトに PPPoE トンネルを設定するように要求しています。PPPoE の概要については、448 ページの「PPPoE による DSL ユーザーのサポート」を参照してください。

PPPoE トンネルには、消費者、電話会社、ISP の 3 つの関係者が存在しています。 システム管理者は、消費者 (会社の PPPoE クライアントか自宅の消費者) 向けか ISP 側のサーバー上のどちらかに PPPoE を構成します。

この節では、クライアントとアクセスサーバーの両方で PPPoE を実行するための計画情報について説明します。この節の内容は次のとおりです。

- PPPoE ホストとアクセスサーバーの計画情報
- 467 ページの「例 — PPPoE トンネルの構成」で紹介されている PPPoE シナリオの説明

PPPoE トンネルの設定作業については、第 34 章を参照してください。

PPPoE トンネルを設定する前に

構成前の作業は、トンネルをクライアント側に構成するかサーバー側に構成するかによって異なります。どちらの場合も、電話会社と契約を結ぶ必要があります。電話会社では、クライアントには DSL 回線を提供し、アクセスサーバーにはある形式のブリッジと ATM パイプを提供します。ほとんどの契約では、電話会社はユーザーのサイトに機器を設置します。

PPPoE クライアントを構成する前に

PPPoE クライアントの実装は、通常、次の機器から構成されます。

- 個人が使用するパーソナルコンピュータまたはシステム
- DSL モデム。通常は、電話会社かインターネットのアクセスプロバイダが設置する
- (任意) ハブ。複数のクライアントが関係するような会社の DSL 消費者向け
- (任意) スプリッタ。通常はプロバイダが設置する

多くの異なる DSL 構成が可能です。このような構成は、ユーザーや会社のニーズ、プロバイダが提供するサービスによって異なります。

表 **30–6** PPPoE クライアントの計画

情報	作業
個人や自分自身のために自宅の PPPoE クライアントを設定する場合に、PPPoE の領域外の設定情報を入手する	設定の手続きが必要なら、電話会社や ISP に問い合わせる
会社のサイトに PPPoE クライアントを設定する場合に、PPPoE クライアントの情報を得るためにユーザーの名前を入手する。PPPoE リモートクライアントを構成する場合は、DSL 機器を自宅に設置するための情報をユーザーに提供する必要がある	認可されたユーザーのリストを会社の管理者に問い合わせる
PPPoE クライアント上で使用できるインタフェースを探す	各マシン上で `ifconfig -a` コマンドを実行し、インタフェース名を探す
(任意) PPPoE クライアントのパスワードを入手する	割り当てるパスワードをユーザーに問い合わせる。このパスワードは UNIX のログイン用ではなく、リンクの認証用に使用する

PPPoE サーバーを構成する前に

PPPoE アクセスサーバーの計画は、データサービスネットワークへの接続を提供する電話会社と共同で行います。電話会社はユーザーのサイトに回線 (通常は ATM パイプ) を設置し、ユーザーのアクセスサーバーに、ある形式のブリッジを提供します。電話会社のブリッジが提供する Ethernet インタフェースだけでなく、会社が提供するインターネットなどのサービスにアクセスする Ethernet インタフェースも構成する必要があります。

表 **30–7** PPPoE アクセスサーバーの計画

情報	作業
データサービスネットワークの回線に使用するインタフェース	`ifconfig -a` コマンドを実行して、インタフェースを特定する
PPPoE サーバーが提供するサービスの種類	管理者やネットワーク計画者に要件やヒントを問い合わせる
(任意) 消費者に提供するサービスの種類	管理者やネットワーク計画者に要件やヒントを問い合わせる
(任意) リモートクライアントのホスト名とパスワード	ネットワーク計画者や契約交渉の担当者に問い合わせる。ホスト名とパスワードは UNIX のログインではなく、PAP 認証や CHAP 認証に使用する

例 — PPPoE トンネルの構成

この節では、第 34 章で説明する作業の例として、PPPoE トンネルの例を示します。図では、トンネル内のすべてのパーティシパントを示していますが、ユーザーはクライアント側かサーバー側のどちらかの端を管理するだけです。

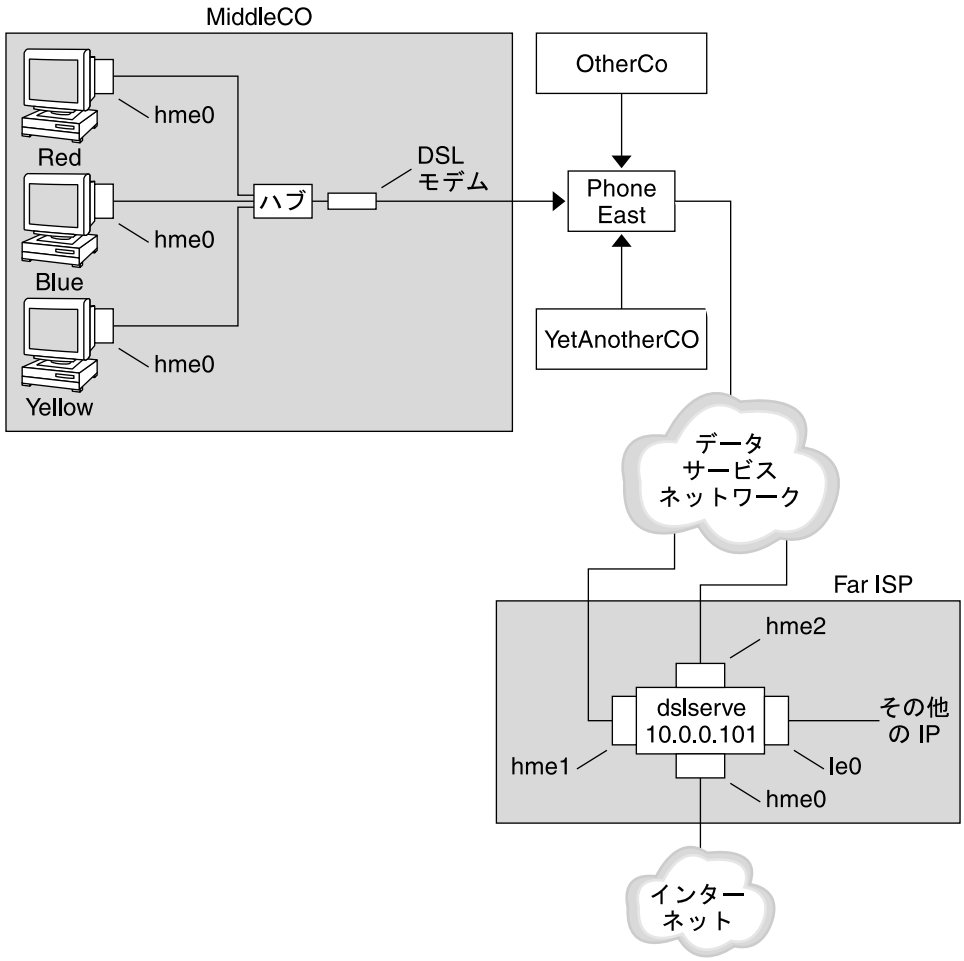


図 **30–5** 例 — PPPoE トンネル

この例では、MiddleCo 社は従業員に高速なインターネットアクセスを提供することを望んでいます。MiddleCo 社は Phone East 社から DSL パッケージを購入し、Phone East 社はサービスプロバイダの Far ISP 社と契約を結びます。Far ISP 社は、Phone East 社から DSL を購入する顧客にインターネットサービスや IP サービスを提供します。

例 — PPPoE クライアントの構成

MiddleCo 社は、サイトに DSL の 1 回線を提供する Phone East 社からパッケージを購入します。パッケージには、MiddleCo 社の PPPoE クライアント用に認証された ISP への専用接続が含まれています。システム管理者は予想される PPPoE クライアントをハブに配線します。Phone East 社の技術者はハブを DSL 機器に配線します。

例 — PPPoE サーバーの構成

FarISP 社では、Phone East 社との契約を履行するために、同社のシステム管理者がアクセスサーバー (`dslserve`) を構成します。このサーバーには、次の 4 つのインタフェースがあります。

- `le0` – ローカルネットワークと接続する主要なネットワークインタフェース
- `hme0` – FarISP 社が顧客にインターネットサービスを提供するためのインタフェース
- `hme1` – 認証された PPPoE トンネル用に MiddleCo 社が使用するインタフェース
- `hme2` – PPPoE トンネル用に別の顧客が使用するインタフェース

PPPoE の詳細情報

作業	参照先
PPPoE クライアントを設定する	508 ページの「PPPoE クライアントの設定」
PPPoE のアクセスサーバーを設定する	511 ページの「PPPoE アクセスサーバーの設定」
PPPoE の詳細情報を入手する	564 ページの「DSL サポート用の PPPoE トンネルの作成」 および `pppoed(1M)`、`pppoec(1M)`、`sppptun(1M)` のマニュアルページ

第 31 章

ダイアルアップ PPP リンクの設定 (手順)

この章では、もっとも一般的な PPP リンクであるダイアルアップリンクの構成作業について説明します。ここでは、次の内容を説明します。

ダイアルアップの PPP リンクを設定する主な作業 (作業マップ)

ダイアルアップ PPP の設定は、モデムの構成、ネットワークデータベースファイルの変更、および 表 36–1 で説明している PPP 構成ファイルの変更によって行います。

次の表は、ダイアルアップ PPP リンクの両側を構成するための主な作業を示しています。通常は、リンクのどちらか一方 (ダイアルアウトマシンかダイアルインサーバー) だけを構成します。

表 **31–1** ダイアルアップの PPP リンクの設定 (作業マップ)

作業	説明	参照先
1. 構成前の情報を収集する	リンクを設定する前に、ピアのホスト名、ターゲットの電話番号、モデムの速度など必要なデータを集める	454 ページの「ダイアルアップ PPP リンクの計画」
2. ダイアルアウトマシンを構成する	リンクを介して呼び出しを行うマシンに PPP を設定する	表 31–2

表 **31–1** ダイアルアップの PPP リンクの設定 (作業マップ)　(続き)

作業	説明	参照先
3. ダイアルインサーバーを構成する	着呼を受信するマシンに PPP を設定する	表 31–4
4. ダイアルインサーバーを呼び出す	pppd コマンドを入力して、通信を開始する	482 ページの「ダイアルインサーバーの呼び出し方法」

ダイアルアウトマシンの構成

この節の作業では、ダイアルアウトマシンの構成方法について説明します。この作業では、図 30–1 で紹介した自宅からのダイアルイン事例を使用します。 予想されるユーザーにマシンを渡す前に、会社での作業があります。あるいは、自分のホームマシンを設定できるように経験豊富なユーザーに指導する必要があります。ダイアルアウトマシンを設定する人は必ずそのマシンのスーパーユーザー権限を持つ必要があります。

ダイアルアウトマシンの構成作業 (作業マップ)

表 **31–2** ダイアルアウトマシンの設定 (作業マップ)

作業	説明	参照先
1. 構成前の情報を収集する	リンクを設定する前に、ピアのホスト名、ターゲットの電話番号、モデムの速度など必要なデータを集める	454 ページの「ダイアルアップ PPP リンクの計画」
2. モデムとシリアルポートを構成する	モデムとシリアルポートを設定する	472 ページの「モデムとシリアルポートの構成方法 (ダイアルアウトマシン)」
3. シリアル回線通信を構成する	シリアル回線上の伝送特性を構成する	473 ページの「シリアル回線を介した通信を定義する方法」
4. ダイアルアウトマシンとピア間の対話を定義する	通信データを収集し、その情報を使用して chat スクリプトを作成する	474 ページの「ピアを呼び出すための命令群を作成する方法」
5. 特定のピア情報を構成する	個々のダイアルインサーバーを呼び出すための PPP オプションを構成する	475 ページの「個々のピアとの接続を定義する方法」
6. ピアを呼び出す	pppd コマンドを入力して、通信を開始する	482 ページの「ダイアルインサーバーの呼び出し方法」

ダイアルアップ PPP のテンプレートファイル

Solaris PPP 4.0 はテンプレートファイルを提供します。各テンプレートファイルには、特定の PPP 構成ファイルのために一般的なオプションが含まれています。次の表は、ダイアルアップリンクの設定に使用できるテンプレートのサンプルと、それらと同等の Solaris PPP 4.0 ファイルを示します。

テンプレートファイル	**PPP** 構成ファイル	参照先
`/etc/ppp/options.tmpl`	`/etc/ppp/options`	538 ページの「`/etc/ppp/options.tmpl` テンプレート」
`/etc/ppp/options.ttya.tmpl`	`/etc/ppp/options.`*ttyname*	540 ページの「`options.ttya.tmpl` テンプレートファイル」
`/etc/ppp/myisp-chat.tmpl`	chat スクリプトを格納するためのユーザー指定の名前を持つファイル	548 ページの「`/etc/ppp/myisp-chat.tmpl` chat スクリプトテンプレート」
`/etc/ppp/peers/myisp.tmpl`	`/etc/ppp/peers/`*peer-name*	544 ページの「`/etc/ppp/peers/myisp.tmpl` テンプレートファイル」

テンプレートファイルを使用するように決めたら、そのファイルの名前を同等の PPP 構成ファイルの名前に変更します。chat ファイルのテンプレート (`/etc/ppp/myisp-chat.tmpl`) だけは例外です。chat スクリプトには任意の名前を指定できます。

ダイアルアウトマシン上にデバイスを構成する

ダイアルアウト PPP マシンを設定するための最初の作業は、シリアル回線にデバイス (モデムとシリアルポート) を構成することです。

注 – モデムに適用する作業は、通常 ISDN TA にも適用します。

以降の手順を実行する前に、次の作業を終了しておく必要があります。

- Solaris 9 オペレーティング環境をダイアルアウトマシンにインストールする
- モデムの最適速度を決定する
- ダイアルアウトマシンに使用するシリアルポートを決定する
- ダイアルアウトマシンのルートパスワードを取得する

計画情報については、表 30–2を参照してください。

▼ モデムとシリアルポートの構成方法 (ダイアルアウトマシン)

1. モデムの設定を行う

   さまざまなタイプのモデムを使用できますが、通常のモデムは Solaris PPP 4.0 用に正しく設定されて出荷されています。次の表は、Solaris PPP 4.0 を使用するモデムの基本的な設定を示しています。

表 **31–3** ダイアルアップ PPP のモデム設定

パラメータ	設定
DCD	キャリアに従う
DTR	モデムがハングアップするように Low に設定する (モデムをオンフックにする)
Flow Control	全二重ハードウェアのフロー制御用 RTS/CTS
Attention Sequences	使用不可

   リンクの設定で問題が発生し、原因がモデムにあれば、まずモデムの製造元のマニュアルを参照します。また、Web 上の多くのサイトが、役に立つモデムの設定情報を提供しています。最後に、523 ページの「モデムの問題を診断する方法」 でモデム問題を解決するためのヒントを見つけることができます。

2. モデムケーブルをダイアルアウトマシンのシリアルポートと電話ジャックに接続します。

3. ダイアルアウトマシン上でスーパーユーザーになります。

4. 『*Solaris* のシステム管理 (上級編)』の「シリアルポートツールによる端末とモデムの設定」で説明するように `admintool` を実行します。

   a. モデムを接続しているポート **(**ポート **a** かポート **b)** をクリックします。

      「シリアルポートの設定」ウィンドウが表示されます。

   b. モデム方向を「発信専用」として指定します。

      ただし、モデムを「発着信両用」としても設定できます (`admintool` のデフォルトのテンプレート)。「発信専用」を選択すると、侵入者に対してセキュリティが強力になります。

   注 – `admintool` でボーレートやタイムアウトを設定できますが、`pppd` デーモンはこれらの設定を無視します。

5. 「**OK**」をクリックして変更を有効にします。

ダイアルアウトマシン上に通信を構成する

この節の手順では、ダイアルアウトマシンのシリアル回線に通信を構成する方法を示します。これらの手順を使用する前に、472 ページの「モデムとシリアルポートの構成方法 (ダイアルアウトマシン)」 で説明しているように、モデムとシリアルポートを設定しておく必要があります。

次の作業は、ダイアルアウトマシンが、PPP 構成ファイルで定義されているオプションに基づいて、ダイアルインサーバーと通信を正常に開始する方法を示します。 次のファイルを作成する必要があります。

- `/etc/ppp/options`
- `/etc/ppp/options.`*ttyname*
- chat スクリプト
- `/etc/ppp/peers/`*peer-name*

Solaris PPP 4.0 は、PPP 構成ファイルにテンプレートを提供します。これらのテンプレートは要求に合わせて変更できます。これらのファイルについては、471 ページの「ダイアルアップ PPP のテンプレートファイル」を参照してください。

▼ シリアル回線を介した通信を定義する方法

1. ダイアルアウトマシン上でスーパーユーザーになります。
2. 次のオプションを指定して、`/etc/ppp/options` と呼ばれるファイルを作成します。

   `lock`

   `/etc/ppp/options` ファイルは、ローカルマシンが実行するすべての通信に適用されるグローバルパラメータの定義に使用されます。`lock` オプションによって、`/var/spool/locks/LK.xxx.yyy.zzz` 形式の UUCP スタイルのロックが可能です。

   注 – ダイアルアウトマシンが `/etc/ppp/options` ファイルを持たない場合は、スーパーユーザーだけが `pppd` コマンドを実行できます。 ただし、`/etc/ppp/options` は空でもかまいません。

   `/etc/ppp/options` については、538 ページの「/etc/ppp/options 構成ファイル」を参照してください。

3. **(省略可能)** 特定のシリアルポートから通信を起動する方法を定義するために、`/etc/ppp/options.`*ttyname* と呼ばれるファイルを作成します。

   次の例は、デバイス名として `/dev/cua/a` を持つポートの `/etc/ppp/options.`*ttyname* ファイルを示しています。

```
# vi /etc/ppp/options.cua.a
crtscts
```

PPP オプション `crtscts` は、`pppd` デーモンに、シリアルポート a のハードウェアフロー制御をオンにするように指示します。

`/etc/ppp/options.ttyname` ファイルについては、539 ページの「`/etc/ppp/options.`*ttyname* 構成ファイル」を参照してください。

4. モデム速度を **478** ページの「モデム速度を設定する方法」 で説明しているとおりに設定します。

▼ ピアを呼び出すための命令群を作成する方法

ダイアルアウトマシンが PPP リンクを開始する前に、ピアになるダイアルインサーバーの情報を収集する必要があります。情報を収集したら、この情報を使用して chat スクリプトを作成します。chat スクリプトには、ダイアルアウトマシンとピア間の実際の対話を記述します。

1. ダイアルアウトマシンのモデムの実行速度を決定します。
   詳細は、545 ページの「モデム速度の設定」を参照してください。
2. ダイアルインサーバーのサイトから次の情報を入手します。
   - サーバーの電話番号
   - 必要な場合、使用している認証プロトコル
   - chat スクリプトでピアが必要とするログインシーケンス
3. ダイアルインサーバーサイトのネームサーバーの名前と **IP** アドレスを入手します。
4. 特定ピアへの呼び出しを開始するための命令群を **chat** スクリプトに置きます。
   たとえば、次の chat スクリプト (`/etc/ppp/mychat`) を作成して、ダイアルインサーバー (`myserver`) を呼び出します。

```
SAY "Calling the peer\n"
        TIMEOUT 10
        ABORT BUSY
        ABORT 'NO CARRIER'
        ABORT ERROR
        REPORT CONNECT
        "" AT&F1&M5S2=255
        TIMEOUT 60
        OK ATDT1-123-555-1234
        CONNECT \c
        SAY "Connected; logging in.\n"
        TIMEOUT 5
        ogin:--ogin: pppuser
        TIMEOUT 20
        ABORT 'ogin incorrect'
        ssword: \qmypassword
        "% " \c
```

```
SAY "Logged in.  Starting PPP on peer system.\n"
ABORT 'not found'
"" "exec pppd"
~ \c
```

スクリプトには、ログインシーケンスを必要とする Solaris ダイアルインサーバーを呼び出すための命令群が含まれています。各命令については、550 ページの「UNIX 方式ログイン用に拡張された基本の chat スクリプト」を参照してください。chat スクリプトの作成については、546 ページの「ダイアルアップリンクでの会話の定義」を参照してください。

注 – chat スクリプトを直接呼び出さないでください。connect オプションの引数に chat スクリプトのファイル名を指定して、スクリプトを呼び出します。

ピアが Solaris か UNIX ベースの類似のオペレーティングシステムを実行する場合は、ダイアルアウトマシンのテンプレートとして前述の chat スクリプトの利用をお勧めします。

▼ 個々のピアとの接続を定義する方法

1. ダイアルアウトマシン上でスーパーユーザーになります。
2. 次の /etc/resolv.conf ファイルを作成して、**DNS** データベースを更新します。

```
domain bigcompany.com
nameserver 10.10.111.15
nameserver 10.10.130.8
```

domain bigcompany.com	ピアの DNS ドメインが bigcompany.com であることを示す
nameserver 10.10.111.15 **nameserver 10.10.130.8**	bigcompany.com 側にあるネームサーバーの IP アドレスの一覧を示す

DNS の実装については、『*Solaris* のシステム管理 *(*ネーミングとディレクトリサービス *: DNS*、*NIS*、*LDAP* 編*)*』の「シリアルポートツールによる端末とモデムの設定」を参照してください。

3. ホスト情報として最初に **DNS** データベースが検索されるように、/etc/nsswitch.conf ファイルを編集します。

```
hosts:      dns [NOTFOUND=return] files
```

4. /etc/ppp/peers ディレクトリを作成して、ピア用のファイルを追加します。
たとえば、次のファイルを作成して、ダイアルインサーバー (myserver) を定義します。

```
# cd /etc/ppp
# mkdir peers
# cd peers
# vi myserver
/dev/cua/a
57600
noipdefault
defaultroute
idle 120
noauth
connect "chat -U ’mypassword’ -T 1-123-555-1213 -f /etc/ppp/mychat"
```

/dev/cua/a	myserver を呼び出すためのシリアルインタフェースとして、デバイス (/dev/cua/a) を使用する必要があることを示す
57600	リンクの速度を定義する
noipdefault	ピア (myserver) のトランザクションでは、ダイアルアウトマシンは最初に 0.0.0.0 の IP アドレスを持つことを示す。 myserver は、すべてのダイアルアップセッションのダイアルアウトマシンに IP アドレスを割り当てる
idle 120	120 秒のアイドル時間が経過するとリンクがタイムアウトになることを示す
noauth	ダイアルアウトマシンとの接続をネゴシエートするとき、ピア (myserver) は認証資格を提供する必要がないことを示す
connect "chat -U ’mypassword’ -T 1-123-555-1213 -f /etc/ppp/mychat"	connect オプションとその引数を示す。引数には、ピアの電話番号、呼び出しの命令群を持つ chat スクリプト (/etc/ppp/mychat) などが指定されている

次に進む手順

作業	参照先
別のダイアルアウトマシンを構成する	472 ページの「モデムとシリアルポートの構成方法 (ダイアルアウトマシン)」
別のコンピュータにダイアルアウトすることで、モデムの接続性をテストする	cu(1C) と tip(1) のマニュアルページ。これらのユーティリティは、モデムが正しく構成されて、別のマシンとの接続が確立できる場合にテストを支援できる
PPP 構成ファイルの詳細情報を入手する	533 ページの「ファイルおよびコマンド行での PPP オプションの使用」

作業	参照先
ダイアルインサーバーの構成を開始する	477 ページの「ダイアルインサーバーにデバイスを構成する」

ダイアルインサーバーの構成

この節の作業では、ダイアルアウトマシンからの呼び出しを PPP リンクを介して受信するダイアルインサーバー (ピアマシン) を構成します。作業では、図 30–1 で紹介したダイアルインサーバー (`myserver`) の構成方法を示します。

ダイアルインサーバーの構成作業 (作業マップ)

表 **31–4** ダイアルインサーバーの設定 (作業マップ)

作業	説明	参照先
1. 構成前の情報を収集する	リンクを設定する前に、ピアのホスト名、ターゲットの電話番号、モデムの速度など必要なデータを集める	454 ページの「ダイアルアップ PPP リンクの計画」
2. モデムとシリアルポートを構成する	モデムとシリアルポートを設定する	478 ページの「モデムとシリアルポートの構成方法 (ダイアルインサーバー)」
3. ピア情報の呼び出しを構成する	ダイアルインサーバーへの呼び出しが許可されているすべてのダイアルアウトマシンにユーザー環境と PPP オプションを設定する	479 ページの「ダイアルインサーバーのユーザーを構成する方法」
4. シリアル回線通信を構成する	シリアル回線上の伝送特性を構成する	481 ページの「シリアル回線を介した通信を定義する方法 (ダイアルインサーバー)」

ダイアルインサーバーにデバイスを構成する

次の手順では、モデムとシリアルポートをダイアルインサーバーに構成する方法について説明します。

手順を実行する前に、ピアであるダイアルインサーバー上で次の作業を終了しておく必要があります。

- Solaris 9 オペレーティング環境のインストール
- モデムの最適速度を決定する
- 使用するシリアルポートの決定

モデムとシリアルポートの構成方法 (ダイアルインサーバー)

1. モデムの製造元が発行するマニュアルに従ってモデムのプログラムを作成します。
   詳細は、472 ページの「モデムとシリアルポートの構成方法 (ダイアルアウトマシン)」を参照してください。
2. モデムをダイアルインサーバー上のシリアルポートに接続します。
3. ダイアルインサーバー上でスーパーユーザーになります。
4. 『*Solaris* のシステム管理 (上級編)』の「シリアルポートツールによる端末とモデムの設定」で説明しているように、`admintool` を使ってシリアルポートを構成します。
   `admintool` を使用して、次の作業を行います。
   a. モデムを接続しているシリアルポート **(**ポート **a** かポート **b)** を選択します。
      「シリアルポートの設定」ウィンドウが表示されます。
   b. モデム方向を「着信専用」として指定します。

      注 – Solaris PPP 4.0 は、モデムに対して双方向通信をサポートしています。

   c. 「**OK**」をクリックして変更を有効にします。

▼ モデム速度を設定する方法

次の手順では、ダイアルインサーバーのモデム速度を設定する方法について説明します。Sun Microsystems のコンピュータを使用する際のモデム速度については、545 ページの「モデム速度の設定」を参照してください。

1. ダイアルインサーバーにログインします。
2. **tip** コマンドを使用して、モデムにアクセスします
   `tip` によるモデム速度の設定については、`tip`(1) のマニュアルページを参照してください。
3. 固定 **DTE** レートでモデムを構成します。
4. 『*Solaris* のシステム管理 (上級編)』の「シリアルポートツールによる端末とモデムの設定」で説明しているように、`ttymon` または `admintool` を使ってシリアルポートをそのレートで固定します。

次に進む手順

作業	参照先
ダイアルインサーバーに別のシリアルポートとモデムを構成する	478 ページの「モデムとシリアルポートの構成方法 (ダイアルインサーバー)」
ダイアルインサーバーを呼び出すユーザー情報を構成する	479 ページの「ダイアルインサーバーのユーザーを構成する方法」

ダイアルインサーバーのユーザーを設定する

ダイアルインサーバーの設定プロセスでは、既知の各リモート呼び出し側に関する情報を構成する必要があります。

この節の手順を開始する前に、次の作業を終了しておく必要があります。

- リモートダイアルアウトマシンからログインが許されているすべてのユーザーの UNIX ユーザー名を入手する
- 478 ページの「モデムとシリアルポートの構成方法 (ダイアルインサーバー)」 で説明しているとおりに、モデムとシリアル回線を設定する
- IP アドレスを専用化して、リモートユーザーからの着呼に割り当てる。潜在的な呼び出し側の数がダイアルインサーバー上のモデムとシリアルポートの数を超える場合に、オプションですべての呼び出しを専用の IP アドレスにすると役立つ。専用 IP アドレスについては、562 ページの「呼び出し元の IP アドレス指定スキーマの作成」を参照してください。

▼ ダイアルインサーバーのユーザーを構成する方法

1. ダイアルインサーバー上でスーパーユーザーになります。
2. 各リモート **PPP** ユーザーに対して、ダイアルインサーバー上で新しいアカウントを作成します。

   `admintool` または Solaris 管理コンソールを使用して、新しいユーザーを作成できます。Solaris 管理コンソールを使って新しいユーザーを作成するには、『*Solaris* のシステム管理 (基本編)』の「ユーザーアカウントの設定 (作業マップ)」を参照してください。`admintool` を使って新しいユーザーを作成するには、`admintool(1M)` を参照してください。

注 – 残りの手順では、`admintool` を使ってアカウントを作成する方法を示します。Solaris 管理コンソールを使ってアカウントを作成する場合と同じパラメータを使用できます。

---

3. 「ユーザーの追加 **(Add User)**」テンプレートを使用して、新しいユーザーを作成します。

   たとえば、次の表は、ダイアルアウトマシン (myhome) 上の user1 に対して pppuser と呼ばれるアカウントを PPP 関連パラメータに記入する方法を示しています。

テンプレートパラメータ	値	定義
ユーザー名	pppuser	リモートユーザーのユーザーアカウント名。このアカウント名は、chat スクリプトのログインシーケンスで指定されているアカウント名と一致する必要がある。たとえば、pppuser は、474 ページの「ピアを呼び出すための命令群を作成する方法」 の chat スクリプトにあるアカウント名である
ログインシェル	/usr/bin/pppd	リモートユーザーのデフォルトのログインシェル。ログインシェル (/usr/bin/pppd) は最初から呼び出し側を専用 PPP 環境に制限する
「ホームディレクトの作成」のパス	/export/home/pppuser	ホームディレクトリ (/export/home/pppuser) は、呼び出し側が正常にダイアルインサーバーにログインするとき設定される

4. 各呼び出し側に対して、$HOME/.ppprc ファイルを作成します。このファイルには、ユーザーの **PPP** セッションに固有のさまざまなオプションが格納されています。

   たとえば、pppuser に対して、次の .ppprc ファイルを作成します。

```
# cd /export/home/pppuser
# vi .ppprc
noccp
```

   noccp は、リンク上で圧縮制御をオフにします。

次に進む手順

作業	参照先
ダイアルインサーバーに追加のユーザーを設定する	479 ページの「ダイアルインサーバーのユーザーを構成する方法」
ダイアルインサーバーを介した通信を構成する	481 ページの「シリアル回線を介した通信を定義する方法 (ダイアルインサーバー)」

ダイアルインサーバーを介した通信を構成する

次の作業は、ダイアルインサーバーが、次の PPP 構成ファイルで定義されているオプションに基づいて、任意のダイアルアウトマシンと通信を開始する方法を示します。

- `/etc/ppp/options`
- `/etc/ppp/options.`*ttyname*

これらのファイルについては、533 ページの「ファイルおよびコマンド行での PPP オプションの使用」を参照してください。

先に進む前に、次の作業を終了しておく必要があります。

- 478 ページの「モデムとシリアルポートの構成方法 (ダイアルインサーバー)」 で説明しているとおりに、ダイアルインサーバーにシリアルポートとモデムを構成する
- 479 ページの「ダイアルインサーバーのユーザーを構成する方法」 で説明しているとおりに、ダイアルインサーバーの予想されるユーザー情報を構成する

シリアル回線を介した通信を定義する方法 (ダイアルインサーバー)

1. ダイアルインサーバー上でスーパーユーザーになります。
2. 次の引数を指定して、`/etc/ppp/options` ファイルを作成します。

   `nodefaultroute`

   `nodefaultroute` は、サーバーの経路が定義されていないことを示します。

   注 – ダイアルインサーバーが `/etc/ppp/options` ファイルを持たない場合は、スーパーユーザーだけが `pppd` コマンドを実行できます。 ただし、`/etc/ppp/options` ファイルは空でもかまいません。

   ---

3. `/etc/options.`*ttyname* ファイルを作成して、シリアルポート (*ttyname*) を介して受信される呼び出しの制御方法を定義します。

   次の `/etc/options.ttya` ファイルでは、ダイアルインサーバーのシリアルポート (`/dev/ttya`) が着呼を制御する方法を定義しています。

   ```
   :10.0.0.80
   xonxoff
   ```

`:10.0.0.80`	シリアルポート (`ttya`) を介して呼び出しているすべてのピアに IP アドレス (10.0.0.80) を割り当てる

xonxoff	ソフトウェアのフロー制御を有効にすることで、シリアル回線はモデムからの通信を制御できる

次に進む手順

この章のすべての手順を実行すると、ダイアルアップリンクの構成が完成します。

作業	参照先
別のコンピュータにダイアルアウトすることで、モデムの接続性をテストする	`cu(1C)` と `tip(1)` のマニュアルページ。これらのユーティリティは、モデムが正しく構成されて、別のマシンとの接続が確立できる場合にテストを支援できる
ダイアルインサーバーのオプションを追加して構成する	477 ページの「ダイアルインサーバーの構成」
ダイアルアウトマシンを追加して構成する	470 ページの「ダイアルアウトマシンの構成」
リモートマシンがダイアルインサーバーを呼び出す	482 ページの「ダイアルインサーバーの呼び出し」

ダイアルインサーバーの呼び出し

ダイアルアウトマシンがダイアルインサーバーを呼び出すことで、ダイアルアップ PPP リンクを確立します。ダイアルアウトマシンの PPP 構成ファイルに `demand` オプションを指定することで、ダイアルアウトマシンがサーバーを呼び出すように指示できます。リンクの確立でもっとも一般的な方法は、ユーザーがダイアルアウトマシン上で `pppd` コマンドを実行することです。

次の作業に進む前に、次のどちらかの作業か両方の作業を終了しておく必要があります。

- 470 ページの「ダイアルアウトマシンの構成」 で説明しているとおりに、ダイアルアウトマシンを設定する
- 477 ページの「ダイアルインサーバーの構成」 で説明しているとおりに、ダイアルインサーバーを設定する

▼ ダイアルインサーバーの呼び出し方法

1. `root` ではなく、通常のユーザーアカウントを使用して、ダイアルアウトマシンにログインします。

2. pppd コマンドを実行して、ダイアルインサーバーを呼び出します。
   たとえば、次のコマンドは、ダイアルアウトマシンとダイアルインサーバー (myserver) 間のリンクを開始します。

```
% pppd 57600 call myserver
```

pppd	pppd デーモンを呼び出すことで呼び出しを開始する
57600	ホストとモデム間の回線速度を設定する
call myserver	pppd の call オプションを呼び出して、475 ページの「個々のピアとの接続を定義する方法」 で作成された /etc/ppp/peers/myserver ファイルのオプション群を読み取る

3. サーバーのネットワーク上にあるホスト **(**図 **30–1** に示されている lindyhop ホストなど**)** にアクセスします。

```
ping lindyhop
```

   リンクが正しく動作している場合、標準的な Telnet のログインシーケンスが端末のウィンドウに表示されます。リンクが正しく動作していない場合は、第 35 章を参照してください。

4. **PPP** セッションを終了します。

```
% pkill -TERM -x pppd
```

次に進む手順

この章のすべての手順を実行すると、ダイアルアップリンクの構成が完成します。

作業	参照先
ユーザーがダイアルアウトマシン上で作業を開始する	482 ページの「ダイアルインサーバーの呼び出し方法」
リンク上の問題を修正する	第 35 章
この章で使用するファイルとオプションについてさらに学習する	533 ページの「ファイルおよびコマンド行での PPP オプションの使用」

第 32 章

専用回線 PPP リンクの設定 (手順)

この章では、専用回線を使用した、ピア間での PPP リンクを設定する方法について説明します。 主に次の内容について説明します。

専用回線の設定 (作業マップ)

専用回線リンクの設定は、ダイアルアップリンクのそれに比べて、比較的簡単です。ほとんどの場合、CSU/DSU、ダイアルサービス、または認証を設定する必要はありません。CSU/DSU の設定は複雑なので、これを設定する必要がある場合は、製造元のマニュアルを参照してください。

次の表の作業マップでは、基本的な専用回線リンクの設定に必要な作業について説明しています。

注 – SVC (Switched Virtual Circuit) や Switched 56 サービスを使用するフレームリレーなど、専用回線の中には、対するピアのアドレスを「ダイアル」するために、CSU/DSU を必要とするものもあります。

表 32–1 専用回線リンクの設定 (作業マップ)

作業	説明	参照先
1. 構成前の情報を収集する	接続の設定に必要な情報を収集する	表 30–4

表 **32–1** 専用回線リンクの設定 (作業マップ)　(続き)

作業	説明	参照先
2. 専用回線への接続に使用するハードウェアを設定する	CSU/DSU および同期インタフェースカードを取り付ける	486 ページの「同期デバイスの設定方法」
3. 必要に応じて、インタフェースカードを設定する	専用回線に接続する際に使用するインタフェーススクリプトを設定する	486 ページの「同期デバイスの設定方法」
4. リモートピアに関する情報に基づいて設定する	ローカルマシンとリモートピア間の通信方法を定義する	488 ページの「専用回線上のマシンの設定方法」
5. 専用回線への接続を開始する	マシンを設定し、起動プロセスの一部として、PPP が専用回線を介して開始されるようにする	488 ページの「専用回線上のマシンの設定方法」

専用回線上の同期デバイスの設定

この節では、専用回線のトポロジに必要な機器を設定する方法について説明します。専用回線のトポロジについては、458 ページの「例 — 専用回線リンクの構成」 で紹介しています。専用回線への接続に必要な同期デバイスには、インタフェースとモデムが含まれています。

同期デバイスを設定する際の前提条件

次の手順に従う前に、下記の項目を確認する必要があります。

- プロバイダによって設置された専用回線が動作していること
- 同期装置 (CSU/DSU)
- システムに Solaris 9 オペレーティング環境リリースがインストールされていること
- システムに必要な同期インタフェースカード

▼ 同期デバイスの設定方法

1. 必要に応じて、インタフェースカードをローカルマシンに取り付けます。
   製造元のマニュアルの手順に従います。
2. **CSU/DSU** とインタフェースをケーブルで接続します。必要に応じて、**CSU/DSU** と専用回線のジャックまたは同様のコネクタをケーブルで接続します。

3. 製造元またはネットワークプロバイダのマニュアルの手順に従って、**CSU/DSU** を設定します。

注 – 専用回線を貸し出しているプロバイダが、接続用の CSU/DSU を提供および設定する場合もあります。

---

4. 必要に応じて、インタフェースのマニュアルの手順に従って、インタフェースカードを設定します。

   インタフェースカードの設定時に、インタフェースの起動スクリプトを作成します。図 30–2 のような専用回線設定では、LocalCorp にあるルーターは、HSI/S インタフェースカードを使用します。

   次のスクリプト `hsi-conf` によって、HSI/S インタフェースが開始されます。

```
#!/bin/ksh
/opt/SUNWconn/bin/hsi_init hih1 speed=1536000 mode=fdx loopback=no \
nrzi=no txc=txc rxc=rxc txd=txd rxd=rxd signal=no 2>&1> /dev/null
```

`hih1`	使用されている同期ポートが HSI/S であることを示す
`speed=1536000`	CSU/DSU の速度を 1536000 に設定する

次に進む手順

作業	参照先
専用回線上のローカルマシンの設定	488 ページの「専用回線上のマシンの設定方法」

専用回線上のマシンの設定

この節では、ルーターを専用回線の終端でローカルピアとして機能するように設定する方法について説明します。ここでは、458 ページの「例 — 専用回線リンクの構成」で紹介した専用回線を例として使用します。

専用回線上のローカルマシンを設定する際の前提条件

以降の手順を実行する前に、次の作業を終了しておく必要があります。

- 486 ページの「専用回線上の同期デバイスの設定」の説明に従って、接続に使用する同期デバイスをセットアップおよび設定する
- 専用回線上のローカルマシンのスーパーユーザーパスワードを取得する
- ローカルマシンがネットワークのルーターとして動作し、専用回線プロバイダのサービスを使用するように設定する

▼ 専用回線上のマシンの設定方法

1. ローカルマシン **(**ルーター**)** のスーパーユーザーになります。
2. リモートピア用のエントリをルーターの `/etc/hosts` ファイルに追加します。

```
# vi /etc/hosts
#
# Internet host table
#
127.0.0.1       localhost
192.168.130.10  local2-peer        loghost
192.168.130.11  local1-net
10.0.0.25     farISP
```

   サンプル `/etc/hosts` は、架空の LocalCorp のローカルルーター用のファイルです。サービスプロバイダのリモートピア `farISP` の IP アドレスおよびホスト名をメモしておきます。

3. プロバイダのピアに関する情報を保持する `/etc/ppp/peers/`*peer-name* ファイルを作成します。

   サンプルの専用回線への接続用に、`/etc/ppp/peers/farISP` ファイルを作成します。

```
#vi /etc/ppp/peers/farISP
init ’/etc/ppp/conf_hsi’
local
/dev/hih1
sync
noauth
192.168.130.10:10.0.0.25
nodefaultroute
passive
persist
noccp
nopcomp
novj
noaccomp
```

次の表では、/etc/ppp/peers/farISP で使用されているオプションおよびパラメータについて説明しています。

オプション	定義
init '/etc/ppp/conf_hsi'	スクリプト /etc/ppp/conf_hsi のパラメータを使用して、接続を開始し HSI インタフェースを設定する
local	データ端末レディー (DTR) 信号の状態を変更しないように、またデータキャリア検出 (DCD) 入力信号を無視するように、pppd デーモンに指示する
/dev/hih1	同期インタフェースのデバイス名を指定する
sync	接続の同期エンコーディングを確立する
noauth	接続時の認証を無効にする
192.168.130.10:10.0.0.25	ローカルピアおよびリモートピアの IP アドレスをコロンで区切って定義する
passive	最大数の LCP Configure-Request を発行したら、ピアが起動するまで待機するように、ローカルマシンの pppd デーモンに指示する
persist	接続が解除された後でもう一度接続を開始するように、pppd デーモンに指示する
noccp、nopcomp、novj、noaccomp	CCP (Compression Control Protocol)、プロトコルフィールドの圧縮、Van Jacobson 圧縮、およびアドレスとコントロールフィールドの圧縮をそれぞれ無効にする。これらの圧縮形式を使用すると、ダイアルアップリンクでの伝送速度は速くなるが、専用回線での伝送速度は遅くなる可能性がある

4. demand という初期設定スクリプトを作成します。こうすると、起動プロセスの一部として **PPP** リンクが開始されます。

```
# cd /etc/ppp/
# vi demand
if [ -f /var/run/ppp-demand.pid ] &&
           /usr/bin/kill -s 0 `/bin/cat /var/run/ppp-demand.pid`
        then
                :
        else
                /usr/bin/pppd call farISP
        fi
```

demand スクリプトには、専用回線リンクを確立するための pppd コマンドが含まれています。次の表では、$PPPDIR/demand の内容について説明しています。

コーディング例	説明
`echo "Starting Solaris PPP 4.0\c"`	起動プロセス中に、「Starting Solaris PPP 4.0」と表示する

`if ps -e | grep '\<pppd\> /dev/null 2>&1 ; then`
`echo "\npppd daemon is still running"`
`echo "or in the process of exiting"`
`exit 0`
	既存の pppd デーモンを検索する
	pppd が検出されたら、メッセージを送信し、demand スクリプトを終了する

`echo "\nEstablishing PPP session...\n"`	起動中に、「Establishing PPP session」と表示する
`/usr/bin/pppd call farISP`	/etc/ppp/peers/farISP にあるオプションを使用して、pppd コマンドを実行する

Solaris PPP 4.0 の起動スクリプト /etc/rc2.d/S47pppd によって、demand スクリプトが、Solaris の起動プロセスの一部として呼び出されます。/etc/rc2.dS47pppd にある次の行は、$PPPDIR/demand というファイルが存在するかどうかを調べます。

```
if [ -f $PPPDIR/demand ]; then
            . $PPPDIR/demand
    fi
```

$PPPDIR/demand が検出された場合は、それが実行されます。$PPPDIR/demand の一連の処理の実行中に、接続が確立されます。

次に進む手順

この章のすべての手順を実行すると、専用回線接続の構成が完了します。

作業	参照先
ユーザーに、インターネット上のマシン、またはリモートピアによって提供されている他のネットワーク上のマシンとの通信を開始するように指示する	ユーザーに、telnet、ftp、rsh、またはローカルネットワークの外部にあるマシンにアクセスするための同様のコマンドを実行させる
リンク上の問題を修正する	障害追跡の情報については、528 ページの「専用回線の問題の解決」を参照
この章で使用するファイルとオプションについてさらに学習する	533 ページの「ファイルおよびコマンド行での PPP オプションの使用」

第 33 章

PPP 認証の設定 (手順)

この章では、PPP 認証の設定手順について説明します。ここでは、次の内容を説明します。

- 492 ページの「PAP 認証の設定」
- 500 ページの「CHAP 認証の設定」

ここでは、ダイアルアップリンクに認証を実装する方法について説明しています。これは、ダイアルアップリンクの方が、専用回線リンクよりも認証を設定することが多いためです。ただし、企業のセキュリティポリシーにより、専用回線に認証を設定する必要がある場合には、それも可能です。専用回線に認証を設定する場合は、この章の手順をガイドラインとして参照してください。

PPP 認証を使用する場合で、どのプロトコルを使用したらいいのかわからないときには、448 ページの「PPP 認証を使用する理由」 の節を参照してください。PPP 認証の詳細については、`pppd(1M)` のマニュアルページおよび 555 ページの「接続時の呼び出し元の認証」を参照してください。

PPP 認証の構成 (作業マップ)

次の作業マップに、PPP 認証に関連する作業を示します。

表 33–1 一般的な PPP 認証 (作業マップ)

作業	参照先
PAP 認証を設定する	492 ページの「PAP 認証の設定 (作業マップ)」
CHAP 認証を設定する	500 ページの「CHAP 認証の設定 (作業マップ)」

PAP 認証の設定

この節では、パスワード認証プロトコル (PAP) を使用して、PPP リンクに認証を実装する方法について説明します。 ここでは、460 ページの「例 — PPP の認証構成」の例を使用して、ダイアルアップリンクで PAP を動作させる方法について説明します。PAP 認証を実装する場合は、この手順を基準として使用してください。

以降の手順を実行する前に、次の作業を終了しておく必要があります。

- ダイアルインサーバーと信頼できる呼び出し元が所有するダイアルアウトマシン間で、ダイアルアップリンクを設定しテストする。
- ダイアルインサーバーでの認証に備えて、LDAP、NIS、NIS+、またはローカルファイル内でネットワークパスワードデータベースを管理しているマシンに対するスーパーユーザーとしてのアクセス権を取得することが理想的です。
- ローカルマシン、およびダイアルインサーバーまたはダイアルアウトマシンに対するスーパーユーザーとしての権限を取得する。

PAP 認証の設定 (作業マップ)

次の作業マップに、ダイアルインサーバーおよびダイアルアウトマシン上の信頼できる呼び出し元に対して実行する PAP 関連の作業を示します。

表 **33–2** PAP 認証についての作業マップ (ダイアルインサーバー)

作業	説明	参照先
1. 構成前の情報を収集する	ユーザー名など、認証に必要なデータを収集する	460 ページの「リンクへの認証計画」
2. 必要に応じて、パスワードデータベースを更新する	候補となるすべての呼び出し元が、サーバーのパスワードデータベースに含まれていることを確認する	493 ページの「PAP 資格データベースの作成方法 (ダイアルインサーバー)」
3. PAP データベースを作成する	将来接続する可能性のあるすべての呼び出し元のセキュリティ資格を `/etc/ppp/pap-secrets` に作成する	493 ページの「PAP 資格データベースの作成方法 (ダイアルインサーバー)」
4. PPP の構成ファイルを変更する	PAP 特有のオプションを `/etc/ppp/options` と `/etc/ppp/peers/`*peer-name* に追加する	495 ページの「PPP 構成ファイルに PAP サポートを追加する方法 (ダイアルインサーバー) 」

表 **33–3** PAP 認証についての作業マップ (ダイアルアウトマシン)

作業	説明	参照先
1. 構成前の情報を収集する	ユーザー名など、認証に必要なデータを収集する	460 ページの「リンクへの認証計画」
2. 信頼できる呼び出し元のマシン用の PAP データベースを作成する	信頼できる呼び出し元のセキュリティ資格と、必要であれば、ダイアルアウトマシンを呼び出す他のユーザーのセキュリティ資格を `/etc/ppp/pap-secrets` に作成する	497 ページの「信頼できる呼び出し元に PAP 認証資格を設定する方法 」
3. PPP の構成ファイルを変更する	PAP 特有のオプションを `/etc/ppp/options` と `/etc/ppp/peers/`*peer-name* に追加する	498 ページの「PPP 構成ファイルに PAP サポートを追加する方法 (ダイアルアウトマシン) 」

ダイアルインサーバーに PAP 認証を構成する

PAP 認証を設定するには、次の手順に従う必要があります。

- PAP 資格データベースを作成します。
- PAP をサポートするように PPP 構成ファイルを変更します。

▼ PAP 資格データベースの作成方法 (ダイアルインサーバー)

ここでは、`/etc/ppp/pap-secrets` ファイルを変更します。このファイルには、接続時に呼び出し元の認証に使用する PAP セキュリティ資格が含まれています。PPP リンクを行う両方のマシンに `/etc/ppp/pap-secrets` が必要です。

図 30–3 で紹介した PAP 構成のサンプルでは、PAP の `login` オプションが使用されています。このオプションを使用する場合は、ネットワークのパスワードデータベースも更新する必要がある可能性があります。 `login` オプションの詳細については、558 ページの「`/etc/ppp/pap-secrets` での `login` オプションの使用」を参照してください。

1. 候補となる信頼できる呼び出し元のリストを作成します。信頼できる呼び出し元とは、自分のリモートマシンからダイアルインサーバーを呼び出す権限を与えられているユーザーです。
2. ダイアルインサーバーのパスワードデータベースに、信頼できる呼び出し元全員の **UNIX** ユーザー名およびパスワードがあることを確認します。

注 – これは、この PAP 構成のサンプルにとって重要です。このサンプルでは、呼び出し元の認証に、PAP の `login` オプションを使用しています。PAP に `login` を実装しない場合は、呼び出し元の PAP ユーザー名と UNIX ユーザー名を一致させる必要はありません。標準の `/etc/ppp/pap-secrets` については、556 ページの「`/etc/ppp/pap-secrets` ファイル」 を参照してください。

候補となる信頼できる呼び出し元に UNIX 名とパスワードがない場合は、次の手順に従います。

a. 呼び出し元に関する情報がない場合は、その呼び出し元の管理者または他のシステム管理者に、そのリモートユーザーがダイアルインサーバーへのアクセス権を持っているかどうかを確認します。

b. 企業のセキュリティポリシーが指定する方法に従って、これらの呼び出し元に **UNIX** ユーザー名およびパスワードを作成します。

3. ダイアルインサーバーのスーパーユーザーとなり、`/etc/ppp/pap-secrets` ファイルを編集します。

Solaris PPP 4.0 では、`/etc/ppp` に `pap-secrets` ファイルがあります。このファイルには、PAP 認証の使用方法についてのコメントが含まれています。ただし、オプションについてのコメントは含まれていません。 コメントの最後に、次のオプションを追加することができます。

```
#
user1       myserver        ""          *
user2       myserver        ""          *
myserver    user2           serverpass  *
```

`/etc/ppp/pap-secrets` の `login` オプションを使用するには、信頼できる呼び出し元の UNIX 名をすべて入力する必要があります。3 番目のフィールドのどこに二重引用符 (" ") が記述されても、呼び出し元のパスワードは、サーバーのパスワードデータベースで参照できます。

エントリ `myserver * serverpass *` には、ダイアルインサーバー用の PAP ユーザー名およびパスワードが含まれています。図 30–3 では、信頼できる呼び出し元である `user2` は、リモートピアに認証を要求します。 そのため、`myserver` の `/etc/ppp/pap-secrets` ファイルには、`user2` との接続を確立する場合に使用する PAP 資格が含まれています。

次に進む手順

作業	参照先
PPP 構成ファイルを変更し、PAP 認証をサポートする	495 ページの「PPP 構成ファイルを PAP 用に変更する (ダイアルインサーバー)」
信頼できる呼び出し元のダイアルアウトマシンで、PAP 認証を設定する	496 ページの「信頼できる呼び出し元の PAP 認証の設定 (ダイアルアウトマシン) 」

PPP 構成ファイルを PAP 用に変更する (ダイアルインサーバー)

この節では、ダイアルインサーバーで PAP 認証をサポートするように、既存の PPP 構成ファイルを更新する方法について説明します。

▼ PPP 構成ファイルに PAP サポートを追加する方法 (ダイアルインサーバー)

ここでは、481 ページの「シリアル回線を介した通信を定義する方法 (ダイアルインサーバー)」で紹介した PPP 構成ファイルを例として使用します。

1. ダイアルインサーバーにスーパーユーザーとしてログインします。
2. 認証オプションを `/etc/ppp/options` ファイルに追加します。

   たとえば、既存の `/etc/ppp/options` ファイルに、次の太字のオプションを追加すると、PAP 認証を実装することができます。

```
lock
idle 120
nodefaultroute
name myserver
auth
require-pap
user myserver
remotename user2
login
```

name myserver	myserver をローカルマシン上のユーザーの PAP 名として設定する。 login オプションを使用する場合は、PAP 名をパスワードデータベースにあるそのユーザーの UNIX ユーザー名と一致させる必要がある
auth	接続を確立する前に、サーバーが呼び出し元を認証する必要があることを明示する
require-pap	呼び出し元に、PAP 資格を要求する
user myserver	myserver をローカルマシンのユーザー名として定義する
remotename user2	user2 をローカルマシンに認証資格を要求するピアとして定義する
login	ローカルマシンは、認証を要求されたときにはいつでも、/etc/ppp/pap-secrets ファイルにある PAP の login オプションを使用することを示す

3. **473** ページの「シリアル回線を介した通信を定義する方法」の説明に従って、/etc/ppp/options.*ttyname* ファイルを作成します。
4. **479** ページの「ダイアルインサーバーのユーザーを構成する方法」の説明に従って、リモート呼び出し元の $HOME/.ppprc ファイルをそれぞれ設定します。

次に進む手順

作業	参照先
ダイアルインサーバーの信頼できる呼び出し元の PAP 認証資格を設定する	496 ページの「信頼できる呼び出し元の PAP 認証の設定 (ダイアルアウトマシン) 」

信頼できる呼び出し元の PAP 認証の設定 (ダイアルアウトマシン)

この節では、信頼できる呼び出し元のダイアルアウトマシンで、PAP 認証を設定する手順について説明します。 システム管理者は、マシンで PAP 認証を設定し、それらを将来接続する可能性のある呼び出し元に配布することができます。また、リモート呼び出し元にすでにマシンがある場合は、この節の手順を指示することもできます。

信頼できる呼び出し元に PAP を設定するには、次の 2 つの手順を実行します。

- 呼び出し元の PAP セキュリティ資格を設定します。

- 呼び出し元のダイアルアウトマシンが PAP 認証をサポートするように設定します。

▼ 信頼できる呼び出し元に PAP 認証資格を設定する方法

ここでは、2 人の信頼できる呼び出し元の PAP 資格を設定する方法について説明します。これらのうちの 1 人は、リモートピアに認証資格を要求します。この手順では、システム管理者が、信頼できる呼び出し元のダイアルアウトマシンで PAP 資格を作成することを前提にしています。

1. ダイアルアウトマシンのスーパーユーザーになります。

   図 30–3で紹介した PAP 構成のサンプルでは、`user1` がダイアルアウトマシンを所有しています。

2. 呼び出し元の **pap-secrets** データベースを変更します。

   Solaris PPP 4.0 には、`/etc/ppp/pap-secrets` ファイルがあります。このファイルには、便利な情報が含まれていますが、オプションについては触れていません。 次のオプションをこの `/etc/ppp/pap-secrets` ファイルに追加できます。

   ```
   # user1    myserver  pass1    *
   ```

   `user1` のパスワードである `pass1` は、接続を通して、読み取り可能な ASCII 形式になることに注意してください。`myserver` は、呼び出し元 `user1` がピアで使用する名前です。

3. 他のダイアルアウトマシンのスーパーユーザーになります。

   PAP 認証の例では、呼び出し元 `user2` がこのダイアルアウトマシンを所有しています。

4. 呼び出し元の **pap-secrets** データベースを変更します。

   次のオプションを既存の `/etc/ppp/pap-secrets` ファイルの終わりに追加できます。

   ```
   # user2     myserver   pass2       *
   myserver  user2      serverpass  *
   ```

   この例では、`/etc/ppp/pap-secrets` に 2 つのエントリがあります。最初のエントリには、`user2` が認証のためにダイアルインサーバー `myserver` に渡す PAP セキュリティ資格が含まれています。

   `user2` は、接続のネゴシエーションの一部として、ダイアルインサーバーに PAP 資格を要求します。そのため、`/etc/ppp/pap-secrets` の 2 つ目の行に、`myserver` に要求される PAP 資格も含まれています。

注 – ほとんどの ISP は認証資格を提供しません。そのため、ここで検討しているシナリオの認証資格については、現実的ではありません。

次に進む手順

作業	参照先
その他の呼び出し元に、PAP 資格を作成する	493 ページの「PAP 資格データベースの作成方法 (ダイアルインサーバー)」
ダイアルアウトマシンが PAP 認証をサポートするように設定する	497 ページの「信頼できる呼び出し元に PAP 認証資格を設定する方法 」

PPP 構成ファイルを PAP 用に変更する (ダイアルアウトマシン)

この節では、信頼できる呼び出し元のダイアルアウトマシンで PAP 認証をサポートするように、既存の PPP 構成ファイルを更新する方法について説明します。

ここでは、次のパラメータを使用して、図 30–3 で紹介した `user2` が所有するダイアルアウトマシン上で、PAP 認証を設定します。`user2` は、ダイアルイン `myserver` からの呼び出しを含む着信呼び出し元に、認証を要求します。

▼ PPP 構成ファイルに PAP サポートを追加する方法 (ダイアルアウトマシン)

ここでは、473 ページの「シリアル回線を介した通信を定義する方法」で紹介した PPP 構成ファイルを例として使用します。 この手順に従って、図 30–3 で示した `user2` が所有するダイアルアウトマシンを設定します。

1. ダイアルアウトマシンにスーパーユーザーとしてログインします。
2. `/etc/ppp/options` ファイルを変更します。

   次の `/etc/ppp/options` ファイルには、太字で示した PAP サポート用のオプションが含まれています。

```
#vi /etc/ppp/options
lock
nodefaultroute
name user2
auth
require-pap
```

name user2	user2 をローカルマシン上のユーザーの PAP 名として設定する。 login オプションを使用する場合は、PAP 名をパスワードデータベースにあるそのユーザーの UNIX ユーザー名と一致させる必要がある
auth	接続を確立する前に、ダイアルアウトマシンが呼び出し元を認証する必要があることを明示する
require-pap	ダイアルアウトマシンからの呼び出しを戻すときに、ピアに PAP 資格を要求する

3. リモートマシン myserver 用の /etc/ppp/peers/*peer-name* ファイルを作成します。

   次のサンプルは、475 ページの「個々のピアとの接続を定義する方法」で作成した既存の /etc/ppp/peers/myserver ファイルに、PAP サポートを追加する方法を示しています。

```
# cd /etc/ppp
# mkdir peers
# cd peers
# vi myserver
/dev/cua/a
57600
noipdefault
defaultroute
idle 120
user user2
remotename myserver
connect "chat -U 'mypassword' -f /etc/ppp/mychat"
```

   太字で示した新しいオプションにより、ピア myserver に関する PAP 要件が追加されます。

user user2	user2 をローカルマシンのユーザー名として定義する
remotename myserver	myserver をローカルマシンに認証資格を要求するピアとして定義する

次に進む手順

作業	参照先
ダイアルインサーバーを呼び出して、PAP 認証の設定をテストする	ダイアルインサーバーを呼び出す手順については、482 ページの「ダイアルインサーバーの呼び出し方法」を参照
PAP 認証の詳細を理解する	555 ページの「パスワード認証プロトコル (PAP)」

CHAP 認証の設定

この節では、チャレンジハンドシェーク認証プロトコル (CHAP) を使用して、PPP リンクに認証を実装する方法について説明します。 ここでは、図 30–4 の例を使用して、私設ネットワークへのダイアルアップで CHAP を動作させる方法について説明します。 CHAP 認証を実装する場合は、この手順を基準として使用してください。

以降の手順を実行する前に、次の作業を終了しておく必要があります。

- ダイアルインサーバーと信頼できる呼び出し元が所有するダイアルアウトマシン間で、ダイアルアップリンクを設定しテストします。
- ローカルマシン (ダイアルインサーバーまたはダイアルアウトマシン) に対するスーパーユーザーとしてのアクセス権を取得します。

CHAP 認証の設定 (作業マップ)

表 **33–4** CHAP 認証についての作業マップ (ダイアルインサーバー)

作業	説明	参照先
1. CHAP シークレットをすべての信頼できる呼び出し元に割り当てる	CHAP シークレットを作成する、または呼び出し元に作成させる	501 ページの「CHAP 資格データベースの作成方法 (ダイアルインサーバー)」
2. chap-secrets データベースを作成する	すべての信頼できる呼び出し元のセキュリティ資格を `/etc/ppp/chap-secrets` ファイルに追加する	501 ページの「CHAP 資格データベースの作成方法 (ダイアルインサーバー)」
3. PPP の構成ファイルを変更する	CHAP 特有のオプションを `/etc/ppp/options` と `/etc/ppp/peers/`*peer-name* に追加する	503 ページの「PPP 構成ファイルに CHAP サポートを追加する方法 (ダイアルインサーバー) 」

表 **33–5** CHAP 認証についての作業マップ (ダイアルアウトマシン)

作業	説明	参照先
1. 信頼できる呼び出し元のマシン用の CHAP データベースを作成する	信頼できる呼び出し元のセキュリティ資格と、必要であれば、ダイアルアウトマシンを呼び出す他のユーザーのセキュリティ資格を `/etc/ppp/chap-secrets` に作成する	501 ページの「CHAP 資格データベースの作成方法 (ダイアルインサーバー)」
2. PPP の構成ファイルを変更する	CHAP 特有のオプションを `/etc/ppp/options` ファイルに追加する	505 ページの「PPP 構成ファイルに CHAP サポートを追加する方法 (ダイアルアウトマシン) 」

ダイアルインサーバーに CHAP 認証を構成する

CHAP 認証を設定するには、最初に `/etc/ppp/chap-secrets` ファイルを変更します。このファイルには、CHAP シークレットを含む CHAP セキュリティ資格が含まれています。このセキュリティ資格を使用して、接続時に呼び出し元を認証します。

注 – UNIX の認証メカニズムまたは PAM の認証メカニズムを CHAP とともに使用することはできません。たとえば、493 ページの「PAP 資格データベースの作成方法 (ダイアルインサーバー)」で説明したような PPP `login` オプションを使用することはできません。認証時に、PAM または UNIX スタイルの認証が必要な場合は、代わりに PAP を選択してください。

次に、私設ネットワークにあるダイアルインサーバーの CHAP 認証を実装します。PPP リンクは、外部のネットワークに接続する場合にだけ使用します。ネットワークにアクセスできるのは、ネットワーク管理者からアクセス権を与えられている呼び出し元だけです。その中には、システム管理者が含まれることもあります。

▼ CHAP 資格データベースの作成方法 (ダイアルインサーバー)

1. 信頼できる呼び出し元のユーザー名をすべて含むリストを作成します。信頼できる呼び出し元とは、私設ネットワークを呼び出す権限を与えられているユーザーです。
2. 各ユーザーに **CHAP** シークレットを割り当てます。

注 – CHAP シークレットには、容易に予想しにくいものを選択してください。CHAP シークレットの内容については、予想しにくいものにするということ以外の制限はありません。

CHAP シークレットを割り当てる方法は、企業のセキュリティポリシーにより違います。管理者がシークレットを作成するか、呼び出し元が自分のシークレットを作成する必要があります。自分が CHAP シークレットを割り当てる立場にない場合は、信頼できる呼び出し元によって、または信頼できる呼び出し元のために作成された CHAP シークレットを取得することを忘れないでください。

3. ダイアルインサーバーのスーパーユーザーとなり、`/etc/ppp/chap-secrets` ファイルを変更します。

   Solaris PPP 4.0 には、`/etc/ppp/chap-secrets` ファイルがあります。このファイルには、便利な情報が含まれていますが、オプションについては触れていません。サーバー `CallServe` 用の次のオプションを既存の `/etc/ppp/chap-secrets` ファイルの最後に追加することができます。

```
account1  CallServe   key123   *
account2  CallServe   key456   *
```

   `key123` は、信頼できる呼び出し元 `account1` の CHAP シークレットです。 `key456` は、信頼できる呼び出し元 `account2` の CHAP シークレットです。

次に進む手順

作業	参照先
その他の信頼できる呼び出し元に、CHAP 資格を作成する	501 ページの「CHAP 資格データベースの作成方法 (ダイアルインサーバー)」
PPP 構成ファイルを更新し、CHAP をサポートする	503 ページの「PPP 構成ファイルに CHAP サポートを追加する方法 (ダイアルインサーバー) 」
信頼できる呼び出し元のダイアルアウトマシンで、CHAP 認証を設定する	504 ページの「信頼できる呼び出し元の CHAP 認証の設定 (ダイアルアウトマシン) 」

PPP 構成ファイルを CHAP 用に変更する (ダイアルインサーバー)

この節では、ダイアルインサーバーで CHAP 認証をサポートするように、既存の PPP 構成ファイルを更新する方法について説明します。

▼ PPP 構成ファイルに CHAP サポートを追加する方法 (ダイアルインサーバー)

1. ダイアルインサーバーにスーパーユーザーとしてログインします。
2. /etc/ppp/options ファイルを変更します。
   太字で表示されているオプションを追加して、CHAP がサポートされるようにします。

```
# vi /etc/ppp/options
lock
nodefaultroute
name CallServe
auth
require-chap
```

name CallServe	CallServe をローカルマシン上のユーザーの CHAP 名として定義する (ダイアルインサーバー)
auth	ローカルマシンで呼び出し元を認証してから、接続を確立する
require-chap	接続を確立する前に、ピアに CHAP 資格を要求する

3. 信頼できる呼び出し元をサポートするために必要なその他の **PPP** 構成ファイルを作成します。
   479 ページの「ダイアルインサーバーのユーザーを構成する方法」および 481 ページの「シリアル回線を介した通信を定義する方法 (ダイアルインサーバー)」を参照してください。

次に進む手順

作業	参照先
信頼できる呼び出し元の CHAP 認証資格を設定する	501 ページの「CHAP 資格データベースの作成方法 (ダイアルインサーバー)」

信頼できる呼び出し元の CHAP 認証の設定 (ダイアルアウトマシン)

この節では、信頼できる呼び出し元のダイアルアウトマシンで、CHAP 認証を設定する手順について説明します。 企業のセキュリティポリシーによって、管理者と信頼できる呼び出し元のどちらが CHAP 認証を設定するのかが決まります。

リモート呼び出し元が CHAP を設定する場合は、呼び出し元の CHAP シークレットが、ダイアルインサーバーの `/etc/ppp/chap-secrets` ファイルに記述されている CHAP シークレットと一致していることを確認します。その後、呼び出し元に、この節で説明している CHAP 設定の手順を指示します。

信頼できる呼び出し元に CHAP を設定するには、次の 2 つの手順を実行します。

- 呼び出し元の CHAP セキュリティ資格を作成します。
- 呼び出し元のダイアルアウトマシンが CHAP 認証をサポートするように設定します。

▼ 信頼できる呼び出し元に CHAP 認証資格を設定する方法

ここでは、2 人の信頼できる呼び出し元に、PAP 資格を設定する方法について説明します。この手順では、システム管理者が、信頼できる呼び出し元のダイアルアウトマシンで CHAP 資格を作成することを前提にしています。

1. ダイアルアウトマシンのスーパーユーザーになります。

   463 ページの「例 — CHAP 認証による構成」で紹介した CHAP 構成のサンプルでは、信頼できる呼び出し元 `account1` がダイアルアウトマシンを所有しています。

2. **chap-secrets** データベースを呼び出し元 `account1` 用に変更します。

   Solaris PPP 4.0 には、`/etc/ppp/chap-secrets` ファイルがあります。このファイルには、便利な情報が含まれていますが、オプションについては触れていません。 次のオプションをこの既存の `/etc/ppp/chap-secrets` ファイルに追加できます。

   ```
   # account1  CallServe   key123   *
   ```

   `CallServe` は、`account1` がアクセスを試みているピアの名前です。`key123` は、`account1` と `CallServer` 間での接続に使用する CHAP シークレットです。

3. 他のダイアルアウトマシンのスーパーユーザーになります。

   呼び出し元 `account2` がこのマシンを所有しているとします。

4. `/etc/ppp/chap-secrets` データベースを呼び出し元 `account2` 用に変更します。

   ```
   # account2  CallServe   key456   *
   ```

   `account2` に、シークレット `key456` が、ピア `CallServe` への接続に使用する CHAP 資格として設定されます。

次に進む手順

作業	参照先
信頼できる呼び出し元のダイアルアウトマシンで、CHAP 資格を作成する	501 ページの「CHAP 資格データベースの作成方法 (ダイアルインサーバー)」
ダイアルアウトマシンが CHAP 認証をサポートするように設定する	504 ページの「信頼できる呼び出し元に CHAP 認証資格を設定する方法 」

▼ CHAP を構成ファイルに追加する (ダイアルアウトマシン)

次の手順に従って、463 ページの「例 — CHAP 認証による構成」で紹介した呼び出し元 `account1` が所有するダイアルアウトマシンを設定します。

PPP 構成ファイルに CHAP サポートを追加する方法 (ダイアルアウトマシン)

1. ダイアルアウトマシンにスーパーユーザーとしてログインします。
2. `/etc/ppp/options` ファイルが次のオプションを持つことを確認します。

```
# vi /etc/ppp/options
lock
nodefaultroute
```

3. リモートマシン `CallServe` 用の `/etc/ppp/peers/`*peer-name* ファイルを作成します。

```
# mkdir /etc/ppp/peers
# vi CallServe
/dev/cua/a
57600
noipdefault
defaultroute
idle 120
user account1
connect "chat -U 'mypassword' -f /etc/ppp/mychat"
```

オプション **`user account1`** により、`account1` が、`CallServe` に提供される CHAP ユーザー名として設定されます。前のファイルの他のオプションについては、475 ページの「個々のピアとの接続を定義する方法」 の `/etc/ppp/peers/myserver` ファイルにある同様のオプションの説明を参照してください。

次に進む手順

作業	参照先
ダイアルインサーバーを呼び出して、CHAP 認証をテストする	482 ページの「ダイアルインサーバーの呼び出し方法」
CHAP 認証の詳細を理解する	559 ページの「チャレンジハンドシェーク認証プロトコル (CHAP)」

第 34 章

PPPoE トンネルの設定 (手順)

この章では、PPPoE トンネルの両端、つまり PPPoE クライアントと PPPoE アクセスサーバーを設定する方法について説明します。 ここでは、次の内容を説明します。

- 507 ページの「PPPoE トンネル設定の主な作業 (作業マップ)」
- 508 ページの「PPPoE クライアントの設定」
- 511 ページの「PPPoE アクセスサーバーの設定」

ここでは、465 ページの「PPPoE トンネルを介した DSL サポートの計画」で紹介したシナリオを例として使用します。 PPPoE の概要については、448 ページの「PPPoE による DSL ユーザーのサポート」を参照してください。

PPPoE トンネル設定の主な作業 (作業マップ)

次の表に、PPPoE クライアントと PPPoE アクセスサーバーを構成するための主な作業をリストします。サイトで PPPoE を実装するには、PPPoE トンネルの自分の側だけ、つまりクライアント側かアクセスサーバー側のどちらかを設定します。

表 **34–1** PPPoE クライアントの設定 (作業マップ)

作業	説明	参照先
1. PPPoE のインタフェースを構成する	Ethernet インタフェースを PPPoE トンネルで使用するために定義する	509 ページの「PPPoE クライアントのインタフェースを構成する方法」
2. PPPoE アクセスサーバーに関する情報を構成する	PPPoE トンネルのサービスプロバイダ側にあるアクセスサーバーのパラメータを定義する	509 ページの「PPPoE アクセスサーバーピアを定義する方法」

表 **34–1** PPPoE クライアントの設定 (作業マップ)　(続き)

作業	説明	参照先
3. PPP 構成ファイルを設定する	まだクライアントの PPP 構成ファイルを定義していない場合は、定義する	473 ページの「シリアル回線を介した通信を定義する方法」
4. トンネルを作成する	アクセスサーバーを呼び出す	手順 5

表 **34–2** PPPoE アクセスサーバーの設定 (作業マップ)

作業	説明	参照先
1. PPPoE のインタフェースを構成する	Ethernet インタフェースを PPPoE トンネルで使用するために定義する	511 ページの「アクセスサーバーの PPPoE 用インタフェースを構成する方法」
2. アクセスサーバーが提供するサービスを構成する	予想される PPPoE クライアントが「発見」できるように、提供するサービスを説明する	512 ページの「アクセスサーバーのクライアントにサービスを提供する方法」
3. PPP 構成ファイルを設定する	まだクライアントの PPP 構成ファイルを定義していない場合は、定義する	481 ページの「ダイアルインサーバーを介した通信を構成する」
4. (省略可能) インタフェースの使用を限定する	PPPoE オプションと PAP 認証を使用して、特定の Ethernet インタフェースの使用を特定のクライアントに限定する	512 ページの「インタフェースの使用を特定のクライアントに限定する方法」

PPPoE クライアントの設定

DSL モデムを介してクライアントマシンに PPP サービスを提供するには、まずモデムまたはハブに接続されているインタフェースで PPPoE を構成する必要があります。次に、PPP 構成ファイルを変更して、PPPoE の反対側のアクセスサーバーを定義する必要があります。

PPPoE クライアント設定の前提条件

PPPoE クライアントを設定する前に、以下を行っておく必要があります。

- PPPoE トンネルを使用するため、クライアントマシンに Solaris 8 Update 6 以降のリリースをインストールする
- サービスプロバイダに連絡して PPPoE アクセスサーバーに関する情報を得る
- クライアントマシンが使用するデバイス (DSL モデム、スプリッタなど) を電話会社またはサービスプロバイダに取り付けてもらう、あるいは自分で取り付ける

▼ PPPoE クライアントのインタフェースを構成する方法

1. **PPPoE** クライアント上でスーパーユーザーになります。
2. **DSL** 接続のある **Ethernet** インタフェースの名前を `/etc/ppp/pppoe.if` ファイルに追加します。

   たとえば、DSL モデムに接続するネットワークインタフェースに `hme0` を使用する PPPoE クライアントの場合は、`/etc/ppp/pppoe.if` に次のエントリを追加します。

   ```
   hme0
   ```

   `/etc/ppp/pppoe.if` の詳細は、565 ページの「`/etc/ppp/pppoe.if` ファイル」を参照してください。
3. **PPPoE** を使用するためのインタフェースを構成します。

   ```
   # /etc/init.d/pppd start
   ```
4. **(**省略可能**)** インタフェースが **PPPoE** に **plumb** されたことを確認します。

   ```
   # /usr/sbin/sppptun query
   hme0:pppoe
   hme0:pppoed
   ```

   `/usr/sbin/sppptun` コマンドを使ってインタフェースを手動で PPPoE に plumb することもできます。 手順については、566 ページの「`/usr/sbin/sppptun` コマンド」を参照してください。

▼ PPPoE アクセスサーバーピアを定義する方法

`/etc/ppp/peers/`*peer-name* ファイルでアクセスサーバーを定義します。アクセスサーバーで使用されるオプションの多くは、ダイアルインサーバーをダイアルアップシナリオで定義するのにも使用できます。`/etc/ppp/peers.`*peer-name* の詳細は、543 ページの「`/etc/ppp/peers/`*peer-name* ファイル」を参照してください。

1. **PPPoE** クライアント上でスーパーユーザーになります。
2. `/etc/ppp/peers/`*peer-name* ファイルでサービスプロバイダの **PPPoE** アクセスサーバーを定義します。

   たとえば、次のファイル `/etc/ppp/peers/dslserve` は、467 ページの「例 — PPPoE トンネルの構成」で紹介した FarISP にあるアクセスサーバー `dslserve` を定義しています。

   ```
   # cat /etc/ppp/peers/dslserve
   sppptun
   plugin pppoe.so
   connect "/usr/lib/inet/pppoec hme0"
   noccp
   ```

```
noauth
user Red
password redsecret
noipdefault
defaultroute
```

このファイルのオプションの定義については、573 ページの「 アクセスサーバーピアを定義するための /etc/ppp/peers/*peer-name* ファイル」 を参照してください。

3. **PPPoE** クライアント上の他の **PPP** 構成ファイルを変更します。

   a. **470** ページの「ダイアルアウトマシンの構成」 で説明したダイアルアウトマシンを構成する手順に従って、/etc/ppp/options を構成します。

   b. /etc/ppp/options.sppptun ファイルを作成し、**PPPoE** に **plumb** されているインタフェースの **PPP** オプションを記述します。

      539 ページの「/etc/ppp/options.*ttyname* 構成ファイル」 で説明されている /etc/ppp/options.*ttyname* ファイルで使用できるオプションは、どれでも使用できます。 sppptun は pppd 構成で指定されているデバイス名なので、ファイル名には /etc/ppp/options.sppptun を使用する必要があります。

4. すべてのユーザーがクライアント上で **PPP** を起動できることを確認します。

   ```
   # touch /etc/ppp/options
   ```

5. **PPP** が **DSL** 回線上で動作できるかどうかをテストします。

   ```
   # pppd debug updetach call dslserve
   ```

   dslserve は、467 ページの「例 — PPPoE トンネルの構成」 で示した ISP のアクセスサーバーに指定されている名前です。debug updetach オプションにより、デバッグ情報が端末のウィンドウに表示されます。

   PPP が正しく動作した場合、端末の出力には、接続がアクティブになるにつれて接続状況が表示されます。PPP が動作しない場合は、次のコマンドを実行してサーバーが正しく動作しているかどうかを確認します。

   ```
   # /usr/lib/inet/pppoec -i hme0
   ```

次に進む手順

作業	参照先
別の PPPoE クライアントを構成する	508 ページの「PPPoE クライアントの設定」
PPPoE についてさらに学ぶ	564 ページの「DSL サポート用の PPPoE トンネルの作成」
構成した PPPoE クライアントのユーザーが DSL 回線上で PPP の実行を開始する	**pppd call ISP-server-name** と入力してアプリケーションやサービスを実行する方法について説明する
PPPoE および PPP の障害追跡	第 35 章

作業	参照先
PPPoE のアクセスサーバーを構成する	511 ページの「PPPoE アクセスサーバーの設定」

PPPoE アクセスサーバーの設定

サービスプロバイダ会社の場合、DSL 接続を介してサイトに到達するクライアントに対してインターネットサービスやその他のサービスを提供できます。まず、PPPoE トンネルに使用するサーバー上のインタフェースを決定する必要があります。次に、ユーザーが使用できるサービスの内容を定義します。

▼ アクセスサーバーの PPPoE 用インタフェースを構成する方法

1. アクセスサーバー上でスーパーユーザーになります。
2. **PPPoE** トンネル専用の **Ethernet** インタフェースの名前を `/etc/ppp/pppoe.if` ファイルに追加します。

   たとえば、次の `/etc/ppp/pppoe.if` ファイルを 467 ページの「例 — PPPoE トンネルの構成」で示したアクセスサーバー `dslserve` に使用します。

   ```
   # cat /etc/ppp/pppoe.if
   hme1
   hme2
   ```

3. **PPPoE** を使用するためのインタフェースを構成します。

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

4. **(**省略可能**)** サーバー上のインタフェースが **PPPoE** に **plumb** されていることを確認します。

   ```
   # /usr/sbin/sppptun query
   hme1:pppoe
   hme1:pppoed
   hme2:pppoe
   hme2:pppoed
   ```

   この例は、インタフェース `hme1` および `hme2` が現在 PPPoE に plumb されていることを示しています。`/usr/sbin/sppptun` コマンドを使ってインタフェースを手動で PPPoE に plumb することもできます。 手順については、566 ページの「`/usr/sbin/sppptun` コマンド」を参照してください。

▼ アクセスサーバーのクライアントにサービスを提供する方法

1. アクセスサーバー上でスーパーユーザーになります。
2. /etc/ppp/pppoe ファイルで、アクセスサーバーが提供する広域サービスを定義します。

   次の /etc/ppp/pppoe ファイルは、図 30–5 で示したアクセスサーバー dslserve によって提供されるサービスをリストしています。

   ```
   device hme1,hme2
   service internet
       pppd "proxyarp 192.168.1.1:"
   service debugging
       pppd "debug proxyarp 192.168.1.1:"
   ```

   このファイルの例では、dslserve の Ethernet インタフェース hme1 および hme2 でインターネットサービスが宣言されています。また、Ethernet インタフェース上の PPP リンクでデバッグがオンに設定されています。
3. ダイアルインサーバーと同じ方法で **PPP** 構成ファイルを設定します。

   手順については、481 ページの「ダイアルインサーバーを介した通信を構成する」を参照してください。
4. pppoed デーモンを起動します。

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

   pppd もまた、/etc/ppp/pppoe.if にリストされるインタフェースを plumb します。

▼ 既存の /etc/ppp/pppoe ファイルを変更する方法

1. アクセスサーバー上でスーパーユーザーになります。
2. 必要に応じて /etc/ppp/pppoe を変更します。
3. pppoed デーモンに新しいサービスを認識させます。

   ```
   # pkill -HUP pppoed
   ```

▼ インタフェースの使用を特定のクライアントに限定する方法

次に、インタフェースを PPPoE クライアントのグループに限定する手順を説明します。この作業を実行する前に、インタフェースに割り当てているクライアントの実 Ethernet MAC アドレスを取得する必要があります。

注 – システムによっては、Ethernet インタフェース上で MAC アドレスを変更できます。この機能は便利ですが、セキュリティ対策としては考えないでください。

次の手順では、467 ページの「例 — PPPoE トンネルの構成」 で示した例を使用して、dslserve のインタフェースのうちの 1 つ hme1 を MiddleCo のクライアントに予約する方法を示しています。

1. **511** ページの「アクセスサーバーの **PPPoE** 用インタフェースを構成する方法」 で示した手順に従ってアクセスサーバーのインタフェースを構成します。
2. **512** ページの「アクセスサーバーのクライアントにサービスを提供する方法」 で示した手順に従ってサービスを定義します。
3. サーバーの /etc/ethers データベースにクライアントのエントリを作成します。

   次は、Red、Blue、および Yellow というクライアントのエントリの例です。

   ```
   8:0:20:1:40:30 redether
   8:0:20:1:40:10 yellowether
   8:0:20:1:40:25 blueether
   ```

   この例では、クライアントの Red、Yellow、および Blue の Ethernet アドレスに redether、yellowether、および blueether という記号名を割り当てています。MAC アドレスへの記号名の割り当ては任意です。
4. 特定のインタフェース上で提供されるサービスを限定するには、次の情報を /etc/ppp/pppoe.*device* ファイルで定義します。

   このファイル名で、*device* は定義するデバイスの名前です。

   ```
   # vi /etc/ppp/pppoe.hme1
   service internet
        pppd "name dslserve-hme1"
        clients redether,yellowether,blueether
   ```

   dslserve-hme1 はアクセスサーバーの名前で、pap-secrets ファイル内の同じエントリで使用されます。clients オプションは、インタフェース hme1 の使用を Ethernet 記号名が redether、yellowether、および blueether であるクライアントに限定します。

   /etc/ethers でクライアントの MAC アドレスに記号名を定義していない場合は、clients オプションの引数として数値アドレスを使用できます。この手順で便利なのは、ワイルドカードを使用できる点です。

   たとえば、clients 8:0:20:*:*:* のような数値アドレスを指定できます。このアドレスは、/etc/ethers にリストされているクライアントのうち 8:0:20 で始まる MAC アドレスを持つクライアントに対してだけアクセスを許可します。
5. アクセスサーバーの /etc/ppp/pap-secrets ファイルを作成します。

   ```
   Red         dslserve-hme1   redpasswd     *
   Blue        dslserve-hme1   bluepasswd    *
   Yellow      dslserve-hme1   yellowpassd   *
   ```

エントリは、dslserve の hme1 インタフェース上で PPP を実行することを許可されたクライアントの PAP 名およびパスワードです。

PAP 認証の詳細は、492 ページの「PAP 認証の設定」を参照してください。

次に進む手順

作業	参照先
PPPoE についてさらに学ぶ	564 ページの「DSL サポート用の PPPoE トンネルの作成」
PPPoE および PPP の障害追跡	529 ページの「PPPoE の問題の診断と解決」
PPPoE クライアントを構成する	508 ページの「PPPoE クライアントの設定」
クライアントの PAP 認証を構成する	496 ページの「信頼できる呼び出し元の PAP 認証の設定 (ダイアルアウトマシン) 」
サーバー上の PAP 認証を構成する	493 ページの「ダイアルインサーバーに PAP 認証を構成する」

第 35 章

一般的な問題の解決 (手順)

この章では、Solaris PPP 4.0 で発生する一般的な問題の診断と障害追跡に関する情報を提供します。この節の内容は次のとおりです。

James Carlson による「*PPP Design, Implementation, and Debugging*」やオーストラリア国立大学の Web サイトなどの情報源も、PPP の障害追跡に詳細なアドバイスを提供しています。詳細は、439 ページの「PPP に関する専門技術者向けのリファレンスブック」および 439 ページの「PPP に関する Web サイト」を参照してください。

PPP 問題の解決 (作業マップ)

次の作業マップを使用すれば、一般的な PPP の問題のためのアドバイスや解決方法をすばやく探すことができます。

表 **35–1** PPP の障害追跡 (作業マップ)

作業	定義	参照先
PPP リンクに関する診断情報を取得する	PPP 診断ツールを使って障害追跡の出力を取得する	517 ページの「pppd から診断情報を取得する方法」
PPP リンクのデバッグ情報を取得する	pppd debug コマンドを使って障害追跡の出力を生成する	518 ページの「PPP デバッグをオンに設定する方法」
ネットワークレイヤーでの一般的な問題を障害追跡する	一連の確認作業を行いネットワーク関連の PPP 問題を特定し解決する	519 ページの「ネットワークの問題を診断する方法」
一般的な通信の問題を障害追跡する	PPP リンクに影響を与える通信の問題を特定し解決する	521 ページの「通信の問題を診断し解決する方法」
構成の問題を障害追跡する	PPP 構成ファイルで問題を特定し解決する	523 ページの「PPP 構成の問題を診断する方法」
モデム関連の問題を障害追跡する	モデムの問題を特定し解決する	523 ページの「モデムの問題を診断する方法」
chat スクリプト関連の問題を障害追跡する	ダイアルアウトマシン上の chat スクリプトの問題を特定し解決する	524 ページの「chat スクリプトのデバッグ情報を取得する方法」
シリアル回線の速度の問題を障害追跡する	ダイアルインサーバー上で回線速度の問題を特定し解決する	527 ページの「シリアル回線の速度の問題を診断して解決する方法」
専用回線の一般的な問題を障害追跡する	専用回線の問題を特定し解決する	528 ページの「専用回線の問題の解決」
認証に関連する問題を障害追跡する	認証データベースに関連する問題を特定し解決する	528 ページの「認証の問題の診断と解決」
PPPoE の問題領域を障害追跡する	PPP 診断ツールを使用して、PPPoE の問題を特定し解決するための出力を得る	529 ページの「PPPoE の診断情報を取得する方法」

PPP の障害追跡のためのツール

PPP リンクは、一般に次の 3 つの主要な領域で障害が発生します。

- 接続の確立に失敗する
- 使用するにつれて接続パフォーマンスが低下する
- 接続のどちらかの側でネットワークに原因と考えられる問題が発生する

PPP が動作しているかどうかを確認するためのもっとも簡単な方法は、ping や traceroute のようなコマンドをピアのネットワーク上のホストに対して実行し、結果を調べることです。ただし、確立されている接続のパフォーマンスを監視したり、問題のある接続を障害追跡したりするには、PPP および UNIX のデバッグツールを使用する必要があります。

この節では、pppd および関連するログファイルから診断情報を取得する方法について説明します。この章の残りの節では、PPP 障害追跡ツールを使って発見し解決できる PPP に関する一般的な問題を説明します。

▼ pppd から診断情報を取得する方法

次に、ローカルマシン上の接続の現在の動作を表示する手順を説明します。

1. ローカルマシン上でスーパーユーザーになります。
2. **PPP** に設定されているシリアルデバイスを引数として pppd を実行します。

   # **pppd /dev/***ttyname* **debug updetach**

   次に、pppd をフォアグラウンドで実行したときに表示されるダイアルアップリンクおよび専用回線リンクの診断結果の例を示します。バックグラウンドで pppd debug を実行すると、作成される出力は /etc/ppp/connect-errors ファイルに送られます。

例 **35–1** 正常に動作しているダイアルアップ接続からの出力

```
# pppd /dev/cua/b debug updetach
have route to 0.0.0.0/0.0.0.0 via 172.21.0.4
serial speed set to 230400 bps
Using interface sppp0
Connect: sppp0 <--> /dev/cua/b
sent [LCP ConfReq id=0x7b <asyncmap 0x0> <magic 0x73e981c8> <pcomp> <accomp>]
rcvd [LCP Ident id=0x79 magic=0x0 "ppp-2.4.0b1 (Sun Microsystems, Inc., Dec  6
    2000 09:36:22)"]
Peer Identification: ppp-2.4.0b1 (Sun Microsystems, Inc., Dec  6 2000 09:36:22)
    rcvd [LCP ConfRej id=0x7b <asyncmap 0x0>]
sent [LCP Ident id=0x7c magic=0x0 "ppp-2.4.0b1 (Sun Microsystems, Inc., Nov 15
    2000 09:38:33)"
sent [LCP ConfReq id=0x7d <magic 0x73e981c8> <pcomp> <accomp>]
rcvd [LCP ConfAck id=0x7d <magic 0x73e981c8> <pcomp> <accomp>]
rcvd [LCP ConfAck id=0x78 <magic 0xdd4ad820> <pcomp> <accomp>]
sent [LCP ConfAck id=0x78 <magic 0xdd4ad820> <pcomp> <accomp>]
sent [LCP Ident id=0x7e magic=0x73e981c8 "ppp-2.4.0b1 (Sun Microsystems, Inc.,
    Nov 15 2000 09:38:33)"]
sent [IPCP ConfReq id=0x3d <addr 0.0.0.0> <compress VJ 0f 01>]
rcvd [LCP Ident id=0x7a magic=0xdd4ad820 "ppp-2.4.0b1 (Sun Microsystems, Inc.,
    Dec  6 2000 09:36:22)"]
Peer Identification: ppp-2.4.0b1 (Sun Microsystems, Inc., Dec  6 2000 09:36:22)
rcvd [IPCP ConfReq id=0x92 <addr 10.0.0.1> <compress VJ 0f 01>
sent [IPCP ConfAck id=0x92 <addr 10.0.0.1> <compress VJ 0f 01>
rcvd [IPCP ConfNak id=0x3d <addr 10.0.0.2>]]
```

例 **35–1** 正常に動作しているダイアルアップ接続からの出力　(続き)

```
sent [IPCP ConfReq id=0x3e <addr 10.0.0.2> <compress VJ 0f 01>]
rcvd [IPCP ConfAck id=0x3e <addr 10.0.0.2> <compress VJ 0f 01>]
local  IP address 10.0.0.2
remote IP address 10.0.0.1
```

例 **35–2** 正常に動作している専用回線接続からの出力

```
# pppd /dev/se_hdlc1 default-asyncmap debug updetach
pppd 2.4.0b1 (Sun Microsystems, Inc., Oct 24 2001 07:13:18) started by root, uid 0
synchronous speed appears to be 0 bps
init option: '/etc/ppp/peers/syncinit.sh' started (pid 105122)
Serial port initialized.
synchronous speed appears to be 64000 bps
Using interface sppp0
Connect: sppp0 <--> /dev/se_hdlc1
sent [LCP ConfReq id=0xe9 <magic 0x474283c6><pcomp> <accomp>]
rcvd [LCP ConfAck id=0xe9 <magic 0x474283c6><pcomp> <accomp>]
rcvd [LCP ConfReq id=0x22 <magic 0x8e3a53ff><pcomp> <accomp>]
sent [LCP ConfReq id=0x22 <magic 0x8e3a53ff><pcomp> <accomp>]
sent [LCP Ident id=0xea magic=0x474283c6 "ppp-2.4.0b1 (Sun Microsystems, Inc., Oct
    22 2001 14:31:44)"]
sent [IPCP ConfReq id=0xf7 <addr 0.0.0.0> <compress VJ Of o1>]]
sent [CCP ConfReq id=0x3f <deflate 15> <deflate(old#) 15> <bsd v1 15>]
rcvd [LCP Ident id=0x23 magic=0x8e3a53ff "ppp-2.4.0b1 (Sun Microsystems, Inc., Oct
    22 2001 14:31:44)"]
Peer Identification: ppp-2.4.0b1 (Sun Microsystems, Inc., Oct 22 2001 14:31:44)
rcvd [IPCP ConfReq id=0x25 <addr 10.0.0.1> <compress VJ Of 01>]
sent [IPCP ConfAck id=0x25 <addr 10.0.0.1> <compress VJ Of 01>]
rcvd [CCP ConfReq id=0x3 <deflate 15> <deflate(old#) 15 <bsd v1 15>]
sent [CCP ConfAck id=0x3 <deflate 15> <deflate(old#) 15 <bsd v1 15>]
rcvd [IPCP ConfNak id=0xf8 <addr 10.0.0.2>]
rcvd [IPCP ConfReq id=0xf7 <addr 10.0.0.2> <compress VJ Of 01>]
rcvd [CCP ConfAck id=0x3f <deflate 15> <deflate(old#) 15 <bsd v1 15>]
Deflate (15) compression enabled
rcvd [IPCP ConfAck id=0xf8 <addr 10.0.0.2> <compress VJ Of 01>]
local  IP address 10.0.0.2
remote IP address 10.0.0.1
```

▼ PPP デバッグをオンに設定する方法

次に、pppd コマンドを使ってデバッグ情報を取得する方法を示します。

注 – 手順 1 から手順 3 までは各ホストごとに 1 度実行するだけでかまいません。その後、手順 4 に進んでホストのデバッグをオンに設定できます。

---

1. pppd からの出力を保持するためのログファイルを作成します。

```
% touch /var/log/pppdebug
```

2. 次の pppd 用の syslog 機能を /etc/syslog.conf に追加します。

```
daemon.debug;local2.debug       /var/log/pppdebug
```

3. syslogd を再起動します。

```
% pkill -HUP -x syslogd
```

4. pppd の次の構文を使用して、特定のピアに対する呼び出しのデバッグをオンに設定します。

```
% pppd debug call peer-name
```

peer-name は、/etc/ppp/peers ディレクトリにあるファイル名でなければなりません。

5. ログファイルの内容を表示します。

```
% tail -f /var/log/pppdebug
```

ログファイルの例については、例 35–3を参照してください。

PPP のパフォーマンスに影響を与えるネットワークの問題の解決

PPP リンクがアクティブになっているのに、リモートネットワーク上のホストにほとんど接続できない場合、ネットワークに問題のある可能性があります。この節では、PPP リンクに影響を与えるネットワークの問題を特定し解決する方法について説明します。

▼ ネットワークの問題を診断する方法

1. ローカルマシン上でスーパーユーザーになり、問題のある接続を切断します。

2. 次のオプションを **PPP** 構成に追加して、構成ファイルのオプションのプロトコルを無効にします。

```
noccp novj nopcomp noaccomp default-asyncmap
```

このオプションは、もっとも単純で圧縮を行わない PPP を使用可能にします。コマンド行でこれらのオプションを引数として pppd に指定してみます。これまで接続できなかったホストに接続できれば、次のいずれかの位置にオプションを追加します。

- /etc/ppp/peers/*peer-name*、call オプションの後
- /etc/ppp/options、広域的に適用する場合

3. デバッグをオンに設定してリモートピアを呼び出します。

   ```
   % pppd debug call peer-name
   ```

4. chat の -v オプションを使用して、**chat** プログラムから冗長ログを取得します。

   たとえば、PPP 構成ファイルで次の形式を使用します。

   ```
   connect 'chat -v -f /etc/ppp/chatfile'
   ```

   /etc/ppp/chatfile は、お使いの chat ファイルの名前を表します。

5. **Telnet** または他のアプリケーションを使ってリモートホストに接続し、問題を再度発生させてみます。

   デバッグログを調べます。これでもリモートホストに接続できない場合は、PPP の問題はネットワークに関連している可能性があります。

6. リモートホストの **IP** アドレスが登録されているインターネットアドレスであることを確認します。

   組織によっては、ローカルネットワーク内では通用するが、インターネットへは経路指定できない内部 IP アドレスを割り当てる場合があります。リモートホストが社内にある場合、インターネットに接続するためには、管理者は、NAT (名前 - アドレス変換) またはプロキシサーバーを設定する必要があります。リモートホストが社内にない場合は、遠隔組織に問題を報告する必要があります。

7. 経路指定テーブルを調べます。

   a. ローカルマシンとピアの両方で経路指定テーブルを確認します。

   b. 経路指定テーブルで、ピアからリモートシステムへのパスおよびリモートシステムからピアへの戻りのパスにあるルーターをすべて確認します。

      中間ルーターの設定が間違っていないことを確認します。ピアへの戻りのパスに問題が見つかることがしばしばあります。

8. **(**省略可能**)** マシンがルーターである場合、オプションの機能を確認します。

   ```
   # ndd -set /dev/ip ip_forwarding 1
   ```

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

9. netstat -s および同様のツールから取得した統計を確認します。

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

   a. ローカルマシン上で統計を実行します。

   b. ピアを呼び出します。

   c. netstat -s によって生成された新しい統計を調べます。

   netstat -s によって生成されたメッセージを使用すると、次の表に示したネットワークの問題を特定して解決できます。

表 **35–2** PPP に影響を与える一般的なネットワークの問題

メッセージ	問題	解決方法
`IP packets not forwardable`	ローカルホストで送信経路が見つからない	ローカルホストの経路指定テーブルに欠如している送信経路を追加する
`ICMP input destination unreachable`	ローカルホストで送信経路が見つからない	ローカルホストの経路指定テーブルに欠如している送信経路を追加する
`ICMP time exceeded`	2 つのルーターが同じ着信アドレスを互いに送信し、パケットが互いに何度も往復し、TTL (存続時間) の値を超過した	`traceroute` を使ってルーティングループの源を見つけ、エラーになっているルーターの管理者に連絡する。`traceroute` の詳細は、`traceroute(1M)` のマニュアルページを参照
`IP packets not forwardable`	ローカルホストで送信経路が見つからない	ローカルホストの経路指定テーブルに欠如している送信経路を追加する
`ICMP input destination unreachable`	ローカルホストで送信経路が見つからない	ローカルホストの経路指定テーブルに欠如している送信経路を追加する

10. **DNS 構成を確認します。**

    ネームサービス構成に問題があると、IP アドレスを解釈処理できないため、アプリケーションは障害を発生します。

    ネームサービスの問題を解決するための情報については、『*Solaris* のシステム管理 *(*ネーミングとディレクトリサービス *: DNS*、*NIS*、*LDAP* 編*)*』の「DNS の障害追跡 (参照情報)」を参照してください。

一般的な通信の問題の解決

通信の問題は、2 つのピアが接続の確立に失敗したときに発生します。これらの問題は、実際は、chat スクリプトの構成が不適切であるために発生するネゴシエーションの問題であることがしばしばあります。この節では、通信の問題を解決するための情報を提供します。問題のある chat スクリプトによって発生するネゴシエーションの問題の解決方法については、表 35–5を参照してください。

▼ 通信の問題を診断し解決する方法

1. ローカルマシン上でスーパーユーザーになり、ピアを呼び出します。
2. デバッグをオンに設定してリモートピアを呼び出します。

   `% pppd debug call` *peer-name*

通信の問題によっては、問題解決のためにピアからデバッグ情報を取得する必要がある場合があります。

3. 次の表に示す通信問題のリストに、取得したログの内容に対応する症状があるかどうかを確認します。

表 **35–3** PPP に影響を与える一般的な通信の問題

症状	問題	解決方法
`too many Configure-Requests` メッセージ	あるピアが他のピアを認識できない	次の問題を確認する a. マシンまたはモデムの配線が間違っていないか b. モデムの構成に不適切なビット設定がないか、あるいは間違ったフロー制御がないか c. chat スクリプトが誤っていないか。この場合は、表 35–5を参照
`pppd debug` の出力は LCP が起動していることを示しているが、より上位のプロトコルが失敗したか、あるいは CRC エラーを示している	非同期制御文字マップ (ACCM) が正しく設定されていない	`default-async` オプションを使用して ACCM を標準のデフォルトである FFFFFFFF に設定する。まずコマンド行で `pppd` のオプションとして `default-async` を使用してみる。問題が解決したら、`default-async` を `/etc/ppp/options` または call オプションの後の `/etc/ppp/peers/`*peer-name* に追加する
`pppd debug` の出力は IPCP が起動していることを示しているが、すぐに終了してしまう	IP アドレスの設定が間違っている可能性がある	a. chat スクリプトを調べ、間違った IP アドレスがないか確認する b. chat スクリプトが問題ない場合は、ピアのデバッグログを要求し、ピアのログで IP アドレスを確認する
接続のパフォーマンスが非常に低い	フロー制御構成のエラー、モデム設定のエラー、不適切に設定された DTE レートなどにより、モデムが適切に構成されていない可能性がある	モデム構成を確認し、適宜調整する

PPP 構成の問題の解決

PPP の問題には、PPP 構成ファイルの問題が原因となっているものがあります。この節では、一般的な構成の問題を特定して解決するための情報を提供します。

▼ PPP 構成の問題を診断する方法

1. ローカルマシン上でスーパーユーザーになります。
2. デバッグをオンに設定してリモートピアを呼び出します。

   % pppd debug call *peer-name*

3. 次の表に示す構成問題のリストに、取得したログの内容に対応する症状があるかどうかを確認します。

表 **35–4** 一般的な PPP 構成の問題

症状	問題	解決方法
pppd debug の出力に「Could not determine remote IP address.」というエラーメッセージが含まれる	/etc/ppp/peers/*peer-name* ファイルにそのピアの IP アドレスが存在しない。ピアが、接続ネゴシエーション時に、自身の IP アドレスを提供しない	次の形式を使用して、pppd コマンド行、あるいは /etc/ppp/peers/*peer-name* でピアの IP アドレスを指定する :10.0.0.10
pppd debug の出力が CCP データ圧縮が失敗して接続が解除されたことを示す	ピアの PPP 圧縮設定が衝突している可能性がある	ピアの 1 つで /etc/ppp/options に noccp オプションを追加して CCP 圧縮を無効にする

モデム関連の問題の解決

モデムは、ダイアルアップリンクで問題の発生しやすい領域です。モデム構成でもっともよく発生する問題は、ピアからの応答がないことです。しかし、接続の問題の原因が本当にモデム構成の問題なのかどうかは、判定するのが難しい場合があります。

モデムの基本的な障害追跡に関するヒントは、『*Solaris* のシステム管理 (上級編)』の「端末とモデムの問題を解決する方法」を参照してください。モデムメーカーのマニュアルや Web サイトも、特定の装置に関する問題の解決に役立ちます。この節では、モデムの問題を特定して解決するためのヒントを提供します。

▼ モデムの問題を診断する方法

次の手順は、問題のあるモデム構成が接続の問題の原因となっているかどうかを判定するのに役立ちます。

1. **518** ページの「**PPP** デバッグをオンに設定する方法」 で説明した手順で、デバッグをオンに設定してピアを呼び出します。
2. 作成された /var/log/pppdebug ログを表示します。

出力に次の症状が認められる場合は、モデム構成に問題がある可能性があります。

- ピアから「recvd」メッセージが返されない
- 出力にピアからの LCP メッセージが含まれるが、接続は失敗し、ローカルマシンから「too many LCP Configure Requests」のメッセージが送信される

  このメッセージは、ローカルマシンはピアを認識できるが、ピアはローカルマシンを認識できないことを示します。
- 接続が SIGHUP 信号で終了する

3. ping を使用してさまざまなサイズのパケットを接続上に送信します。

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

   小さいパケットは受信されるが、大きいパケットはドロップされる場合、モデムに問題があることを示します。

4. インタフェース sppp0 上のエラーを確認します。

```
% netstat -ni
Name  Mtu  Net/Dest   Address      Ipkts    Ierrs Opkts    Oerrs Collis Queue
lo0   8232 127.0.0.0  127.0.0.1    826808   0     826808   0     0      0
hme0  1500 172.21.0.0 172.21.3.228 13800032 0     1648464  0     0      0
sppp0 1500 10.0.0.2   10.0.0.1     210      0     128      0     0      0
```

   インタフェースのエラーが時間がたつにつれて増えているようなら、モデム構成に問題がある可能性があります。

chat スクリプト関連の問題の解決

chat スクリプトは、ダイアルアップリンクで問題が発生しやすい領域です。この節では、chat からデバッグ情報を取得する手順と、一般的な問題を解決するためのヒントを示します。

▼ chat スクリプトのデバッグ情報を取得する方法

1. ダイアルアウトマシン上でスーパーユーザーになります。
2. /etc/ppp/peers/*peer-name* ファイルを編集してピアが呼び出されるようにします。
3. connect オプションで指定されている chat コマンドに引数として -v を追加します。

   ```
   connect "/usr/bin/chat -v -f /etc/ppp/chat-script-name"
   ```

4. /etc/ppp/connect-errors ファイルの **chat** スクリプトのエラーを表示します。

   以下は、chat で見られる主なエラーです。

```
Oct 31 08:57:13 deino chat[107294]: [ID 702911 local2.info] expect (CONNECT)
Oct 31 08:57:58 deino chat[107294]: [ID 702911 local2.info] alarm
Oct 31 08:57:58 deino chat[107294]: [ID 702911 local2.info] Failed
```

この例は、(CONNECT) 文字列を待つ間にタイムアウトしたことを示します。chat が失敗すると、pppd から次のメッセージを受け取ります。

```
Connect script failed
```

次の表に、chat スクリプトの一般的なエラーと、エラー解決のためのヒントを示します。

表 **35–5** chat スクリプトの一般的な問題

症状	問題	解決方法
pppd debug の出力に Connect script failed が含まれる	chat スクリプトは、以下のようにユーザー名とパスワードを指定している。 ogin: *user-name* ssword: *password* しかし、接続しようとしたピアはこの情報を要求していない	1. chat スクリプトからログインとパスワードを削除する 2. 再度ピアを呼び出してみる 3. まだメッセージが表示される場合は、ISP に連絡して正しいログインシーケンスを尋ねる
/usr/bin/chat -v ログに次のメッセージが含まれる。 "expect (login:)" alarm read timed out	chat スクリプトは、以下のようにユーザー名とパスワードを指定している。 ogin: pppuser ssword: \q\U しかし、接続しようとしているピアはこの情報を要求していない	1. chat スクリプトからログインとパスワードを削除する 2. 再度ピアを呼び出してみる 3. まだメッセージが表示される場合は、ISP に連絡して正しいログインシーケンスを尋ねる
pppd debug の出力に以下が含まれる。 possibly looped-back	ローカルマシンまたはそのピアがコマンド行で停止していて PPP を実行していない。chat スクリプト内に間違って設定されたログイン名とパスワードがある	1. chat スクリプトからログインとパスワードを削除する 2. 再度ピアを呼び出してみる 3. まだメッセージが表示される場合は、ISP に連絡して正しいログインシーケンスを尋ねる
pppd debug 出力は LCP が起動していることを示しているが、接続がすぐに終了してしまう	chat スクリプト内のパスワードが間違っている可能性がある	1. ローカルマシンの正しいパスワードを確認する 2. chat スクリプト内のパスワードを確認し、間違っている場合は修正する 3. 再度ピアを呼び出してみる 4. まだメッセージが表示される場合は、ISP に連絡して正しいログインシーケンスを尋ねる

表 **35–5** chat スクリプトの一般的な問題　*(続き)*

症状	問題	解決方法
ピアからのテキストがチルダ (~) で始まる	chat スクリプトは、以下のようにユーザー名とパスワードを指定している。 `ogin: pppuser ssword: \q\U` しかし、接続しようとしているピアはこの情報を要求していない	1. chat スクリプトからログインとパスワードを削除する 2. 再度ピアを呼び出してみる 3. まだメッセージが表示される場合は、ISP に連絡して正しいログインシーケンスを尋ねる
モデムが停止する	chat スクリプトに次の行が含まれており、ローカルマシンがピアからの CONNECT メッセージを待つように強制している。 `CONNECT ″`	chat スクリプトがピアからの CONNECT を待つようにするときは、次の行を使用する。 `CONNECT \c` chat スクリプトを `~ \c` で終了する
`pppd debug` の出力に以下が含まれる。 `LCP: timeout sending Config-Requests`	chat スクリプトに次の行が含まれており、ローカルマシンがピアからの CONNECT メッセージを待つように強制している。 `CONNECT ″`	chat スクリプトがピアからの CONNECT を待つようにするときは、次の行を使用する。 `CONNECT \c` chat スクリプトを `~ \c` で終了する
`pppd debug` の出力に以下が含まれる。 `Serial link is not 8-bit clean`	chat スクリプトに次の行が含まれており、ローカルマシンがピアからの CONNECT メッセージを待つように強制している。 `CONNECT ″`	chat スクリプトがピアからの CONNECT を待つようにするときは、次の行を使用する。 `CONNECT \c` chat スクリプトを `~ \c` で終了する
`pppd debug` の出力に以下が含まれる。 `Loopback detected`	chat スクリプトに次の行が含まれており、ローカルマシンがピアからの CONNECT メッセージを待つように強制している。 `CONNECT ″`	chat スクリプトがピアからの CONNECT を待つようにするときは、次の行を使用する。 `CONNECT \c` chat スクリプトを `~ \c` で終了する
`pppd debug` の出力に以下が含まれる。 `SIGHUP`	chat スクリプトに次の行が含まれており、ローカルマシンがピアからの CONNECT メッセージを待つように強制している。 `CONNECT ″`	chat スクリプトがピアからの CONNECT を待つようにするときは、次の行を使用する。 `CONNECT \c` chat スクリプトを `~ \c` で終了する

シリアル回線の速度の問題の解決

ダイアルインサーバーは、速度の設定の衝突が原因で問題が発生する可能性があります。次に示す手順は、接続の問題の原因がシリアル回線速度の衝突であることを特定するのに役立ちます。

速度の問題は、主に次のような原因で発生します。

- /bin/login のようなプログラムを介して PPP を起動し、回線の速度を指定した
- PPP を mgetty から起動し、偶然ビットレートを指定した

pppd は、もともと回線に設定されていた速度を /bin/login または mgetty によって設定された速度に変更し、このことが回線の障害を発生させます。

▼ シリアル回線の速度の問題を診断して解決する方法

1. ダイアルインサーバーにログインし、デバッグをオンに設定してピアを呼び出します。
   手順については、518 ページの「PPP デバッグをオンに設定する方法」を参照してください。
2. 作成された /var/log/pppdebug ログを表示します。
   出力に次のメッセージがないか確認します。

   ```
   LCP too many configure requests
   ```

   このメッセージは、PPP に設定されているシリアル回線の速度が衝突している可能性があることを示します。
3. **PPP** が /bin/login のようなプログラムを介して起動されているかどうかを調べ、設定されている回線速度を調べます。
   このような状況では、pppd はもともと設定されていた回線速度を /bin/login で指定されている速度に変更します。
4. ユーザーが **PPP** を mgetty コマンドから起動し、偶然にビットレートを指定していないかどうか確認します。
   この処理もまた、シリアル回線速度の衝突を引き起こします。
5. 次のようにしてシリアル回線速度の衝突の問題を解決します。
   a. モデムの **DTE** レートをロックします。
   b. **autobaud** を使用しないようにします。
   c. 設定後に回線速度を変更しないようにします。

専用回線の問題の解決

専用回線でもっとも一般的な問題は、パフォーマンスの低下です。ほとんどの場合、問題を解決するためには、電話会社に相談する必要があります。

表 **35–6** 一般的な専用回線の問題

症状	問題	解決方法
接続が開始しない	CSU BPV (CSU 極性違反) が原因の可能性がある。接続の一方の側が AMI 回線用に設定されており、もう一方の側が ESF の B8ZS (Bit 8 Zero Substitute) 用に設定されている。	米国またはカナダのユーザーは、この問題を CSU/DSU のメニューから直接解決できる。詳細は、CSU/DSU メーカーのマニュアルを参照。 その他の地域のユーザーは、プロバイダが CSU BPV の解決策を用意している可能性がある
接続のパフォーマンスが非常に低い	接続上でトラフィックが持続しているときに、`pppd debug` の出力が CRC エラーを示す。回線に、電話会社とネットワークの間の誤った設定によって生じた刻時の問題がある可能性がある。	電話会社に連絡し、「ループ刻時」を使用していたことを確認する。 構造化されていない専用回線では、刻時を提供する必要がある場合がある。北米のユーザーはループクロックを使用すること。

認証の問題の診断と解決

表 **35–7** 一般的な認証の問題

症状	問題	解決方法
`pppd debug` の出力が「`Peer is not authorized to use remote address` *address*」というメッセージを示す	PAP 認証を使用しており、リモートピアの IP アドレスが `/etc/ppp/pap-secrets` ファイルに存在しない	`/etc/ppp/pap-secrets` ファイルで、ピアのエントリの後にアスタリスク (*) を追加する
`pppd debug` の出力は LCP が起動していることを示しているが、その直後に終了してしまう	特定のセキュリティプロトコルのデータベースでパスワードが間違っている可能性がある	`/etc/ppp/pap-secrets` または `/etc/ppp/chap-secrets` ファイルでピアのパスワードを確認する

PPPoE の問題の診断と解決

PPP および標準の UNIX ユーティリティを使用して PPPoE の問題を特定できます。この節では、PPPoE のデバッグ情報を取得し、PPPoE 関連の問題を解決する方法について説明します。

PPPoE の診断情報を取得する方法

接続上の問題が発生し、原因が PPPoE だと思われるとき、次の診断ツールを使って障害追跡情報を取得できます。

1. **PPPoE** トンネルを実行しているマシン、つまり **PPPoE** クライアントまたは **PPPoE** アクセスサーバーでスーパーユーザーになります。
2. **518** ページの「**PPP** デバッグをオンに設定する方法」 で説明した手順で、デバッグをオンに設定します。
3. ログファイル /var/log/pppdebug の内容を表示します。
   以下の例は、PPPoE トンネルとの接続で生成されたログファイルの一部です。

   例 **35–3** PPPoE トンネルとの接続のログファイル

```
Sep  6 16:28:45 enyo pppd[100563]: [ID 702911 daemon.info] Plugin
  pppoe.so loaded.
Sep  6 16:28:45 enyo pppd[100563]: [ID 860527 daemon.notice] pppd
  2.4.0b1 (Sun Microsystems, Inc.,
Sep  5 2001 10:42:05) started by troot, uid 0
Sep  6 16:28:46 enyo pppd[100563]: [ID 702911 daemon.debug] connect option:
   '/usr/lib/inet/pppoec
-v hme0' started (pid 100564)
Sep  6 16:28:46 enyo pppd[100563]: [ID 702911 daemon.info] Serial connection established.
Sep  6 16:28:46 enyo pppd[100563]: [ID 702911 daemon.info] Using interface sppp0
Sep  6 16:28:46 enyo pppd[100563]: [ID 702911 daemon.notice] Connect: sppp0
   <--> /dev/sppptun
Sep  6 16:28:46 enyo pppd[100563]: [ID 702911 daemon.debug] /etc/ppp/pap-secrets
  is apparently empty
Sep  6 16:28:46 enyo pppd[100563]: [ID 702911 daemon.debug] /etc/ppp/chap-secrets
  is apparently empty
Sep  6 16:28:46 enyo pppd[100563]: [ID 702911 daemon.debug] sent
  [LCP ConfReq id=0xef <mru 1492>
asyncmap 0x0 <magic 0x77d3e953><pcomp><acomp>
Sep  6 16:28:46 enyo pppd[100563]: [ID 702911 daemon.debug] rcvd
  [LCP ConfReq id=0x2a <mru 1402>
asyncmap 0x0 <magic 0x9985f048><pcomp><acomp
```

   デバッグの出力によって問題を特定できない場合は、次の手順に進みます。

4. **PPPoE** から診断メッセージを取得します。

```
# pppd connect "/usr/lib/inet/pppoec -v interface-name"
```

pppoec は、診断情報を stderr に送信します。pppd をフォアグラウンドで実行する場合、出力が画面に表示されます。pppd をバックグラウンドで実行する場合、出力は /etc/ppp/connect-errors に送られます。

次の例は、PPPoE トンネルがネゴシエートされたときに生成されるメッセージです。

例 **35–4** PPPoE 診断メッセージ

```
Connect option: '/usr/lib/inet/pppoec -v hme0' started (pid 100564)
/usr/lib/inet/pppoec: PPPoE Event Open (1) in state Dead (0): action SendPADI (2)
/usr/lib/inet/pppoec: Sending PADI to ff:ff:ff:ff:ff:ff: 18 bytes
/usr/lib/inet/pppoec: PPPoE State change Dead (0) -> InitSent (1)
/usr/lib/inet/pppoec: Received Active Discovery Offer from 8:0:20:cd:c1:2/hme0:pppoed
/usr/lib/inet/pppoec: PPPoE Event rPADO+ (5) in state InitSent (1): action SendPADR+ (5)
/usr/lib/inet/pppoec: Sending PADR to 8:0:20:cd:c1:2: 22 bytes
/usr/lib/inet/pppoec: PPPoE State change InitSent (1) -> ReqSent (3)
/usr/lib/inet/pppoec: Received Active Discovery Session-confirmation from
   8:0:20:cd:c1:2/hme0:pppoed
/usr/lib/inet/pppoec: PPPoE Event rPADS (7) in state ReqSent (3): action Open (7)
/usr/lib/inet/pppoec: Connection open; session 0002 on hme0:pppoe
/usr/lib/inet/pppoec: PPPoE State change ReqSent (3) -> Convers (4)
/usr/lib/inet/pppoec: connected
```

診断メッセージによって問題を特定できない場合は、次の手順に進みます。

5. snoop を実行し、トレースをファイルに保存します。

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

   ```
   # snoop -o pppoe-trace-file
   ```

6. snoop トレースファイルを表示します。

   ```
   # snoop -i pppoe-trace-file -v pppoe
   ```

例 **35–5** PPPoE snoop トレース

```
ETHER: ----- Ether Header -----
ETHER:
ETHER: Packet 1 arrived at 6:35:2.77
ETHER: Packet size = 32 bytes
ETHER: Destination = ff:ff:ff:ff:ff:ff, (broadcast)
ETHER: Source      = 8:0:20:78:f3:7c, Sun
ETHER: Ethertype = 8863 (PPPoE Discovery)
ETHER:
PPPoE: ----- PPP Over Ethernet -----
PPPoE:
PPPoE: Version = 1
PPPoE: Type = 1
PPPoE: Code = 9 (Active Discovery Initiation)
PPPoE: Session Id = 0
PPPoE: Length = 12 bytes
PPPoE:
PPPoE: ----- Service-Name -----
```

例 **35–5** PPPoE snoop トレース　(続き)

```
PPPoE: Tag Type = 257
PPPoE: Tag Length = 0 bytes
PPPoE:
PPPoE: ----- Host-Uniq -----
PPPoE: Tag Type = 259
PPPoE: Tag Length = 4 bytes
PPPoE: Data = Ox00000002
PPPoE:
.
.
.
ETHER: ----- Ether Header -----
ETHER:
ETHER: Packet 5 arrived at 6:35:2.87
ETHER: Packet size = 60 bytes
ETHER: Destination = 8:0:20:78:f3:7c, Sun)
ETHER: Source      = 0:2:fd:39:7f:7,
ETHER: Ethertype = 8864 (PPPoE Session)
ETHER:
PPPoE: ----- PPP Over Ethernet -----
PPPoE:
PPPoE: Version = 1
PPPoE: Type = 1
PPPoE: Code = 0 (PPPoE Session)
PPPoE: Session Id = 24383
PPPoE: Length = 20 bytes
PPPoE:
PPP: ----- Point-to-Point Protocol -----
PPP:
PPP-LCP: ----- Link Control Protocol -----
PPP-LCP:
PPP-LCP: Code = 1 (Configure Request)
PPP-LCP: Identifier = 80
PPP-LCP: Length = 18
```

第 36 章

Solaris PPP 4.0 (リファレンス)

この章では、Solaris PPP 4.0 について詳細で概念的な情報を提供します。トピックは次のとおりです。

ファイルおよびコマンド行での PPP オプションの使用

Solaris PPP 4.0 には、PPP 構成を定義するのに使用するオプションが多数含まれます。これらのオプションは、PPP 構成ファイルまたはコマンド行で使用するほか、ファイルでの使用とコマンド行での使用を組み合わせることもできます。この節では、PPP オプションの構成ファイルでの使用と PPP コマンドの引数としての使用について詳細に説明します。

PPP オプションを定義する場所

Solaris PPP 4.0 は柔軟に構成できます。PPP オプションを次の場所で定義できます。

- PPP 構成ファイル
- コマンド行で実行される PPP コマンド

- 前記 2 つの場所の組み合わせ

次の表に、PPP 構成ファイルとコマンドをリストします。

表 **36–1** PPP 構成ファイルとコマンドの概要

ファイルまたはコマンド	定義	参照先
`/etc/ppp/options`	たとえば、マシンがピアにピア自身の認証を要求するかどうかなど、システム上のすべての PPP リンクにデフォルトで適用される特性を含むファイル。このファイルがない場合、スーパーユーザー以外のユーザーは PPP の使用を禁止される	538 ページの「`/etc/ppp/options` 構成ファイル」
`/etc/ppp/options.`*ttyname*	シリアルポート *ttyname* 上のすべての通信の特性を記述するファイル	539 ページの「`/etc/ppp/options.`*ttyname* 構成ファイル」
`/etc/ppp/peers`	通常、ダイアルアウトマシンが接続するピアに関する情報を含むディレクトリ。このディレクトリ内のファイルは、`pppd` コマンドの `call` オプションで使用される	542 ページの「ダイアルインサーバーと通信するための情報の指定」
`/etc/ppp/peers/`*peer-name*	リモートピア *peer-name* の特性、たとえば、ピアの電話番号やピアとの接続をネゴシエートするための chat スクリプトなどを含むファイル	543 ページの「`/etc/ppp/peers/`*peer-name* ファイル」
`/etc/ppp/pap-secrets`	パスワード認証プロトコル (PAP) の認証に必要なセキュリティ資格を含むファイル	556 ページの「`/etc/ppp/pap-secrets` ファイル」
`/etc/ppp/chap-secrets`	チャレンジハンドシェーク認証プロトコル (CHAP) の認証に必要なセキュリティ資格を含むファイル	559 ページの「`/etc/ppp/chap-secrets` ファイル」
`~/.ppprc`	PPP ユーザーのホームディレクトリ内のファイル。ダイアルインサーバーでもっともよく使用される。このファイルには、各ユーザーの構成に関する特定の情報が含まれる	541 ページの「ユーザー独自のオプションの設定」
`pppd` *options*	PPP リンクの開始および PPP リンクの特性の説明のためのコマンドとオプション	535 ページの「PPP オプションの処理方法」

`pppd` コマンドで使用できる PPP ファイルおよびすべてのオプションの説明については、`pppd(1M)` のマニュアルページを参照してください。すべての PPP 構成ファイルのサンプルテンプレートは、`/etc/ppp` にあります。

PPP オプションの処理方法

Solaris PPP 4.0 のすべての操作は、ユーザーが pppd コマンドを実行すると起動する pppd デーモンによって処理されます。ユーザーがリモートピアを呼び出すと、以下が発生します。

1. pppd デーモンが以下を構文解析する
   - /etc/ppp/options
   - $HOME/.ppprc
   - /etc/ppp/options または $HOME/.ppprc の中で file または call オプションによって開かれたファイル
2. pppd がコマンド行を走査して使用中のデバイスを判定する。デーモンはまだ遭遇したオプションを解釈しない
3. pppd は次の条件に基づいて使用するシリアルデバイスを検出しようとする
   a. シリアルデバイスがコマンド行またはそれ以前に処理した構成ファイルで指定されている場合、pppd はそのデバイス名を使用する
   b. シリアルデバイスが指定されていない場合、pppd はコマンド行で notty、pty、または socket オプションを検索する。これらのオプションが指定されている場合、pppd はデバイス名が存在しないとみなす
   c. 上記以外の場合で、標準入力が tty に接続されていることを pppd が検出した場合は、tty の名前を使用する
   d. それでも pppd がシリアルデバイスを見つけられない場合は、接続を終了し、エラーを発生させる
4. pppd は次に /etc/ppp/options.*ttyname* ファイルが存在するかどうかをチェックする。 ファイルが見つかると、pppd はそのファイルを構文解析する
5. pppd はコマンド行のオプションを処理する
6. pppd はリンク制御プロトコル (LCP) のネゴシエーションを行い、接続を確立する
7. (省略可能) 認証が必要な場合、pppd は、/etc/ppp/pap-secrets または /etc/ppp/chap-secrets を読み取り、反対側のピアを認証する

pppd デーモンがコマンド行または他の構成ファイルで call *peer-name* オプションに遭遇すると、/etc/ppp/peers/*peer-name* ファイルが読み取られます。

PPP 構成ファイルにおける特権のしくみ

Solaris PPP 4.0 構成には特権の概念が含まれます。特権は、特に、同じオプションが複数の場所で呼び出されたときに、構成オプションの優先度を判定します。特権ソースから呼び出されたオプションは、非特権ソースから呼び出された同じオプションよりも優先されます。

ユーザー特権

唯一の特権ユーザーは、UID の値が 0 のスーパーユーザー (root) です。その他のすべてのユーザーは特権を与えられません。

ファイル特権

次のファイルは、所有者が誰であるかにかかわらず、特権を与えられる構成ファイルです。

- /etc/ppp/options
- /etc/ppp/options.*ttyname*
- /etc/ppp/peers/*peer-name*

$HOME/.ppprc は、ユーザーが所有するファイルです。$HOME/.ppprc およびコマンド行から読み取られたオプションは、pppd を起動しているユーザーがスーパーユーザーである場合にだけ特権が与えられます。

file オプションの引数は特権が与えられます。

オプション特権の意味

オプションの中には、呼び出したユーザーまたはソースが特権を与えられていないと動作しないものがあります。コマンド行で呼び出されたオプションは、pppd コマンドを実行中のユーザーの特権を割り当てられます。これらのオプションは、pppd を起動しているユーザーが root でなければ、特権が与えられません。

オプション	状態	意味
domain	特権がある	使用には特権が必要
linkname	特権がある	使用には特権が必要
noauth	特権がある	使用には特権が必要
nopam	特権がある	使用には特権が必要
pam	特権がある	使用には特権が必要
plugin	特権がある	使用には特権が必要
privgroup	特権がある	使用には特権が必要
allow-ip *addresses*	特権がある	使用には特権が必要
name *hostname*	特権がある	使用には特権が必要
plink	特権がある	使用には特権が必要

オプション	状態	意味
noplink	特権がある	使用には特権が必要
plumbed	特権がある	使用には特権が必要
proxyarp	noproxyarp が指定されている場合、特権がある	特権のない使用はこのオプションを優先指定できない
defaultroute	nodefaultroute が特権ファイルで、または特権ユーザーによって設定されている場合、特権がある	非特権ユーザーはこのオプションを優先指定できない
disconnect	特権ファイルで、または特権ユーザーによって設定されている場合、特権がある	非特権ユーザーはこのオプションを優先指定できない
bsdcomp	特権ファイルで、または特権ユーザーによって設定されている場合、特権がある	非特権ユーザーは特権ユーザーが指定したサイズより大きいコードサイズを指定できない
deflate	特権ファイルで、または特権ユーザーによって設定されている場合、特権がある	非特権ユーザーは特権ユーザーが指定したサイズより大きいコードサイズを指定できない
connect	特権ファイルで、または特権ユーザーによって設定されている場合、特権がある	非特権ユーザーはこのオプションを優先指定できない
init	特権ファイルで、または特権ユーザーによって設定されている場合、特権がある	非特権ユーザーはこのオプションを優先指定できない
pty	特権ファイルで、または特権ユーザーによって設定されている場合、特権がある	非特権ユーザーはこのオプションを優先指定できない
welcome	特権ファイルで、または特権ユーザーによって設定されている場合、特権がある	非特権ユーザーはこのオプションを優先指定できない
ttyname	特権ファイルで設定されている場合、特権がある	pppd を誰が起動したかに関係なく、スーパーユーザー特権で開かれる
	非特権ファイルで設定されている場合、特権がない	pppd を起動したユーザーの特権で開かれる

/etc/ppp/options 構成ファイル

ローカルマシン上のすべての PPP 通信にグローバルオプションを定義するには、/etc/ppp/options ファイルを使用します。/etc/ppp/options は特権ファイルです。pppd の規則ではありませんが、/etc/ppp/options はスーパーユーザーが所有してください。/etc/ppp/options で定義するオプションは、他のすべてのファイルおよびコマンド行内で定義される同じオプションより優先されます。

/etc/ppp/options で使用する可能性がある代表的なオプションを次に示します。

- **lock** – UUCP 形式のファイルロックを有効にする
- **noauth** – マシンが呼び出し元を認証しないことを示す

注 – Solaris PPP 4.0 ソフトウェアには、デフォルトの /etc/ppp/options ファイルは含まれていません。pppd の動作に、/etc/ppp/options ファイルは必要ありません。ただし、マシンに /etc/ppp/options ファイルがない場合、そのマシンで pppd を実行できるのは root だけであることに注意してください。

---

473 ページの「シリアル回線を介した通信を定義する方法」の説明に従って、テキストエディタを使用して /etc/ppp/options を作成する必要があります。マシンがグローバルオプションを必要としない場合は、空の /etc/ppp/options ファイルを作成できます。これで、root および一般ユーザーの両方がローカルマシン上で pppd を実行できます。

/etc/ppp/options.tmpl テンプレート

/etc/ppp/options.tmpl には、/etc/ppp/options ファイルに関する有用なコメントのほかに、グローバルな /etc/ppp/options ファイルに共通の次の 3 つのオプションが含まれます。

```
lock
nodefaultroute
noproxyarp
```

オプション	定義
lock	UUCP 形式のファイルロックを有効にする
nodefaultroute	デフォルトの送信経路を定義しないことを指定する
noproxyarp	proxyarp を許可しない

/etc/ppp/options.tmpl をグローバルオプションファイルとして使用するには、/etc/ppp/options.tmpl の名前を /etc/ppp/options に変更します。次に、サイトの必要に応じてファイルの内容を変更します。

/etc/ppp/options サンプルファイルの場所

表 **36–2** /etc/ppp/options ファイルの例

/etc/ppp/options の例	参照先
ダイアルアウトマシン用	473 ページの「シリアル回線を介した通信を定義する方法」
ダイアルインサーバー用	481 ページの「シリアル回線を介した通信を定義する方法 (ダイアルインサーバー)」
ダイアルインサーバー上での PAP サポート用	495 ページの「PPP 構成ファイルに PAP サポートを追加する方法 (ダイアルインサーバー) 」
ダイアルアウトマシン上での PAP サポート用	498 ページの「PPP 構成ファイルに PAP サポートを追加する方法 (ダイアルアウトマシン) 」
ダイアルインサーバー上での CHAP サポート用	503 ページの「PPP 構成ファイルに CHAP サポートを追加する方法 (ダイアルインサーバー) 」

/etc/ppp/options.*ttyname* 構成ファイル

シリアル回線上の通信の特性を /etc/ppp/options.*ttyname* ファイルで設定できます。/etc/ppp/options.*ttyname* は特権ファイルです。/etc/ppp/options および$HOME/.ppprc ファイル (存在する場合) を構文解析した後で pppd によって読み取られます。$HOME/.ppprc が存在しない場合は、pppd は /etc/ppp/options を構文解析した後 /etc/ppp/options.*ttyname* を読み取ります。

ttyname は、ダイアルアップリンク、専用回線リンクの両方で使用されます。*ttyname* は、モデムまたは ISDN TA が接続されている可能性があるマシン上の特定のシリアルポート (cua/a、cua/b など) を表します。

/etc/ppp/options.*ttyname* ファイルに名前を付けるときは、デバイス名にあるスラッシュ (/) をドット (.) に置き換えます。たとえば、デバイス cua/b 用の options ファイルの名前は /etc/ppp/options.cua.b になります。

注 – Solaris PPP 4.0 が正常に動作するうえで、/etc/ppp/options.*ttyname* ファイルは必要ありません。 サーバーが PPP 用のシリアル回線を 1 つだけ持ち、オプションはほとんど必要ない場合、オプションを別の構成ファイルまたはコマンド行で指定することができます。

ダイアルインサーバーでの /etc/ppp/options.*ttyname* の使用

ダイアルアップリンクでは、ダイアルインサーバー上のモデムが接続されているすべてのシリアルポートごとに、/etc/ppp/options.*ttyname* ファイルを個別に作成することもできます。通常のオプションは次のとおりです。

- ダイアルインサーバーが必要とする IP アドレス

  シリアルポート *ttyname* に着信する呼び出し元に特定の IP アドレスを使用させる必要がある場合は、このオプションを設定する。使用するアドレス空間により、予想される呼び出し元の数に比べて、PPP で使用可能な IP アドレスの数に制限がある場合がある。その場合は、ダイアルインサーバー上の PPP で使用されるシリアルインタフェースごとに IP アドレスを割り当てることを考える。この割り当ては、PPP に動的なアドレス指定を実装する

- `asyncmap` *map_value*

  `asyncmap` オプションは、特定のモデムまたは ISDN TA がシリアル回線上で受け取らない制御文字を割り当てる。`xonxoff` オプションを使用すると、`pppd` は自動的に 0xa0000 の asyncmap を設定する。

  map_value は、16 進数で入力し、問題のある制御文字を指定する

- `init "chat -U -f /etc/ppp/mychat"`

  `init` オプションは、`chat –U` コマンド内の情報を使用して、シリアル回線上で通信を開始するようにモデムに指示する。モデムは、`/etc/ppp/mychat` ファイル内の chat 文字列を使用する

- `pppd`(1m) のマニュアルページにリストされているセキュリティパラメータ

ダイアルアウトマシンでの `/etc/ppp/options.`*ttyname* の使用

ダイアルアウトマシンでは、モデムが接続されているシリアルポート用に `/etc/ppp/options.`*ttyname* ファイルを作成することも、あるいは `/etc/ppp/options.`*ttyname* を使用しないでおくこともできます。

注 – Solaris PPP 4.0 が正常に動作するうえで、`/etc/ppp/options.`*ttyname* ファイルは必要ありません。 ダイアルアウトマシンが PPP 用のシリアル回線を 1 つだけ持ち、オプションはほとんど必要ない場合、オプションを別の構成ファイルまたはコマンド行で指定することができます。

`options.ttya.tmpl` テンプレートファイル

`/etc/ppp/options.ttya.tmpl` ファイルには、`/etc/ppp/options.`*tty-name* ファイルに関して有用なコメントが含まれています。また、テンプレートには `/etc/ppp/options.`*tty-name* ファイルに共通の次の 3 つのオプションが含まれます。

```
38400
asyncmap 0xa0000
:192.168.1.1
```

オプション	定義
38400	ポート ttya でこのボーレートを使用する
asyncmap 0xa0000	ローカルマシンが接続に失敗したピアと通信できるように asyncmap 値 0xa0000 を割り当てる
:192.168.1.1	接続上で着信しているすべてのピアに IP アドレス 192.168.1.1 を割り当てる

サイトで /etc/ppp/options.ttya.tmpl を使用するには、/etc/ppp/options.tmpl の名前を /etc/ppp/options.*ttya-name* に変更します。*ttya-name* をモデムが接続しているシリアルポートの名前に置き換えます。次に、サイトの必要に応じてファイルの内容を変更します。

/etc/ppp/options.*ttyname* サンプルファイルの場所

表 **36–3** /etc/ppp/options.*ttyname* ファイルの例

/etc/ppp/options.*ttyname* の例	参照先
ダイアルアウトマシン用	473 ページの「シリアル回線を介した通信を定義する方法」
ダイアルインサーバー用	481 ページの「シリアル回線を介した通信を定義する方法 (ダイアルインサーバー)」

ユーザー独自のオプションの設定

この節では、ダイアルインサーバー上でユーザーを設定する方法について詳細に説明します。

ダイアルインサーバーでの $HOME/.ppprc の設定

$HOME/.ppprc ファイルは、独自の PPP オプションを設定するユーザーを対象としています。管理者が、ユーザーのために $HOME/.ppprc を設定することもできます。

$HOME/.ppprc 内のオプションは、ファイルを呼び出しているユーザーに特権がある場合だけ、特権を与えられます。

呼び出し元が pppd コマンドを使って呼び出しを開始した場合、pppd デーモンは、.ppprc ファイルを 2 番目に確認します。

ダイアルインサーバーで $HOME/*.ppprc* を設定する手順については、479 ページの「ダイアルインサーバーのユーザーを設定する」を参照してください。

ダイアルアウトマシンでの $HOME/.ppprc の設定

注 – $HOME/*.ppprc* は、ダイアルアウトマシン上で Solaris PPP 4.0 が正常に動作するのに必要ではありません。

ダイアルアウトマシンでは、特別な場合を除いて、$HOME/.ppprc は必要ありません。以下を行う場合は、1 つ以上の .ppprc ファイルを作成します。

- 通信のニーズが異なる複数のユーザーが同じマシンからリモートピアを呼び出すのを許可する場合。このような場合は、ダイアルアウトする必要がある各ユーザーのホームディレクトリに個別の .ppprc ファイルを作成する
- Van Jacobson 圧縮を無効にするなど、接続に固有の問題を制御するオプションを指定する必要がある場合。接続に関する問題の障害追跡については、James Carlson による「*PPP Design, Implementation, and Debugging*」および pppd(1M) のマニュアルページを参照。

.ppprc ファイルは、ダイアルインサーバーを構成するときにもっとも頻繁に使用されるため、.ppprc の構成手順については 479 ページの「ダイアルインサーバーのユーザーを構成する方法」を参照してください。

ダイアルインサーバーと通信するための情報の指定

ダイアルインサーバーと通信するには、サーバーに関する情報を収集し、いくつかのファイルを編集する必要があります。特に大切なのは、ダイアルアウトマシンが呼び出す必要があるすべてのダイアルインサーバーについて通信要件を設定する必要があることです。ダイアルインサーバーに関する ISP 電話番号などのオプションは、/etc/ppp/options.*ttyname* ファイルで指定できます。 ただし、ピア情報は、/etc/ppp/peers/*peer-name* ファイルで設定するのが最適です。

/etc/ppp/peers/*peer-name* ファイル

注 – /etc/ppp/peers/*peer-name* ファイルは、ダイアルアウトマシン上で Solaris PPP 4.0 が正常に動作するのに必要ではありません。

特定のピアと通信するための情報を指定するには、/etc/ppp/peers/*peer-name* ファイルを使用します。/etc/ppp/peers/*peer-name* を使用すると、一般ユーザーは、自分で設定することを許可されていない、あらかじめ選択された特権オプションを呼び出すことができます。

たとえば、非特権ユーザーの場合、noauth オプションが /etc/ppp/peers/*peer-name* ファイルで指定されていると、このオプションが優先されます。ユーザーが、認証資格を提供しない peerB への接続を設定する場合を考えます。スーパーユーザーは、noauth オプションを含む /etc/ppp/peers/peerB ファイルを作成できます。noauth は、ローカルマシンが peerB からの呼び出しを認証しないことを示します。

pppd デーモンは、次のオプションに遭遇すると、/etc/ppp/peers/*peer-name* を読み取ります。

call *peer-name*

ダイアルアウトマシンが通信する必要があるターゲットピアごとに /etc/ppp/peers/*peer-name* ファイルを作成できます。これは、スーパーユーザーの権限がなくても特定のダイアルアウト接続を呼び出すことを一般ユーザーに許可できる点で特に便利です。

/etc/ppp/peers/*peer-name* で指定する代表的なオプションを次に示します。

- user *user_name*

  PAP または CHAP 認証を行う場合に、ダイアルアウトマシンのログイン名として *user_name* をダイアルインサーバーに指定する

- remotename *peer-name*

  peer-name をダイアルインマシンの名前として使用する。remotename は、/etc/ppp/pap-secrets または /etc/ppp/chap-secrets ファイルを走査するときに、PAP または CHAP 認証と連携して使用される

- connect "chat *chat_script*..."

  chat スクリプト内の命令を使ってダイアルインサーバーへの通信を開く

- noauth

  通信開始時に、ピア *peer-name* の認証を行わない

- noipdefault

  ピアとのネゴシエートに使用する初期 IP アドレスを 0.0.0.0 に設定する。ほとんどの ISP への接続を設定するときに noipdefault を使用すると、ピア間で容易に IPCP ネゴシエーションを行うことができる

- `defaultroute`

  接続上で IP が確立されたときに、デフォルトの IPv4 経路指定をインストールする

特定のターゲットピアに適用する可能性がある上記以外のオプションについては、`pppd(1M)` のマニュアルページを参照してください。

/etc/ppp/peers/myisp.tmpl テンプレートファイル

`/etc/ppp/peers/myisp.tmpl` ファイルには、`/etc/ppp/peers/`*peer-name* ファイルに関して有用なコメントが含まれています。 また、テンプレートには、`/etc/ppp/peers/`*peer-name* ファイルで使用する可能性がある次の一般的なオプションが含まれます。

```
connect "/usr/bin/chat -f /etc/ppp/myisp-chat"
user myname
remotename myisp
noauth
noipdefault
defaultroute
updetach
noccp
```

オプション	定義
`connect "/usr/bin/chat -f /etc/ppp/myisp-chat"`	chat スクリプト `/etc/ppp/myisp-chat` を使ってピアを呼び出す
`user myname`	このアカウント名をローカルマシンに使用する。`myname` は、ピアの `/etc/ppp/pap-secrets` ファイル内でのこのマシンの名前
`remotename myisp`	`myisp` をローカルマシンの `/etc/ppp/pap-secrets` ファイル内のピア名として認識する。
`noauth`	認証資格を提供するためのピアの呼び出しを要求しない
`noipdefault`	ローカルマシンにデフォルトの IP アドレスを使用しない
`defaultroute`	ローカルマシンに割り当てられているデフォルトの経路指定を使用する
`updetach`	標準出力ではなく、PPP ログファイル内にエラーを記録する
`noccp`	CCP 圧縮を使用しない

サイトで /etc/ppp/peers/myisp.tmpl を使用するには、/etc/ppp/peers/myisp.tmpl の名前を /etc/ppp/peers/*peer-name* に変更します。*peer-name* は、呼び出されるピアの名前に置き換えます。次に、サイトの必要に応じてファイルの内容を変更します。

/etc/ppp/peers/*peer-name* サンプルファイルの場所

表 **36–4** /etc/ppp/peers/*peer-name* ファイルの例

/etc/ppp/peers/*peer-name* の例	参照先
ダイアルアウトマシン用	475 ページの「個々のピアとの接続を定義する方法」
専用回線上のローカルマシン用	488 ページの「専用回線上のマシンの設定方法」
ダイアルアウトマシン上での PAP 認証サポート用	498 ページの「PPP 構成ファイルに PAP サポートを追加する方法 (ダイアルアウトマシン) 」
ダイアルアウトマシン上での CHAP 認証サポート用	505 ページの「PPP 構成ファイルに CHAP サポートを追加する方法 (ダイアルアウトマシン) 」
クライアントシステムでの PPPoE サポート用	508 ページの「PPPoE クライアントの設定」

ダイアルアップリンク用のモデムの設定

この節では、モデムの設定について説明します。

モデム速度の設定

モデムの設定で重要なのは、モデムが動作する速度の指定です。Sun Microsystems のコンピュータで使用するモデムには、次のガイドラインを適用してください。

- 旧 SPARC システム – システムに添付されているハードウェアマニュアルを確認する。SPARCstation™ マシンの多くは、38400 bps を超えないモデム速度を要求する
- UltraSPARC™ マシン – モデム速度を 115200 bps に設定する。これは、最新のモデムで使用でき、ダイアルアップリンクに十分な速度である。デュアルチャネル ISDN TA を圧縮して使用する場合は、モデム速度を上げる必要がある。UltraSPARC での最大値は非同期接続で 460800 bps

ダイアルアウトマシンでは、/etc/ppp/peers/*peer-name* などの PPP 構成ファイルでモデム速度を設定するか、あるいは pppd のオプションとして速度を指定します。

ダイアルインサーバーでは、477 ページの「ダイアルインサーバーにデバイスを構成する」 で説明したように、`ttymon` 機能または `admintool` を使って速度を設定する必要があります。

ダイアルアップリンクでの会話の定義

ダイアルアウトマシンとそのリモートピアは、さまざまな命令をネゴシエーションしたり交換したりして PPP リンク上で通信します。ダイアルアウトマシンを構成するときは、ローカルおよびリモートモデムから要求される命令の内容を判定する必要があります。次に、その命令を含む chat スクリプトと呼ばれるファイルを作成します。この節では、モデムの設定および chat スクリプトの作成について説明します。

chat スクリプトの内容

ダイアルアウトマシンが接続する必要があるリモートピアは、通常、それぞれピア自身の chat スクリプトを必要とします。

注 – chat スクリプトは、通常、ダイアルアップリンクだけで使用されます。専用回線リンクは、起動時の設定が必要な非同期インタフェースを使用しないかぎり、chat スクリプトを使用しません。

---

chat スクリプトの内容は、モデムまたは ISDN TA の要件、およびリモートピアの要件によって決まります。スクリプトの内容は、通信の開始処理時にダイアルアウトマシンとリモートピアが交換する一連の送信予期文字列として表示されます。

予期文字列には、会話を開始するためにダイアルアウトホストマシンがリモートピアから受け取ると予想される文字が含まれます。送信文字列には、ダイアルアウトマシンが、予期文字列を受け取った後でリモートピアに送信する文字が含まれます。

chat スクリプト内の情報には、通常、以下が含まれます。

- モデムコマンド (しばしば *AT* コマンドと呼ばれる)。モデムが電話を通じてデータを伝送することを可能にする
- ターゲットピアの電話番号

  この電話番号は、ISP または企業サイトのダイアルインサーバー、あるいは個別のマシンが要求する番号の場合がある
- タイムアウト値 (必要な場合)
- リモートピアからの予想されるログインシーケンス
- ダイアルアウトマシンが送信するログインシーケンス

chat スクリプトの例

この節では、独自の chat スクリプトを作成する際の参考になる chat スクリプトの例を紹介します。モデムメーカーのガイドや ISP および他のターゲットホストからの情報には、モデムおよびターゲットピアの chat の要件が含まれています。また、数多くの PPP Web サイトで chat スクリプトのサンプルが提供されています。

基本のモデム chat スクリプト

以下は、独自の chat スクリプトを作成するためのテンプレートとして使用できる基本の chat スクリプトです。

```
ABORT   BUSY
ABORT   'NO CARRIER'
REPORT  CONNECT
TIMEOUT 10
"" AT&F1M0&M5S2=255
SAY     "Calling myserver\n"
TIMEOUT 60
OK      "ATDT1-123-555-1212"
ogin: pppuser
ssword: \q\U
% pppd
```

次の表では、chat スクリプトの内容を説明します。

スクリプトの内容	説明
`ABORT 'NO CARRIER'`	ダイアル時にモデムが `'NO CARRIER'` を報告した場合、伝送を中止する。 このメッセージは、通常、ダイアルまたはモデムのネゴシエーションが失敗したときに発生する
`REPORT CONNECT`	`CONNECT` 文字列をモデムから収集し、表示する
`TIMEOUT 10`	初期タイムアウトを 10 秒に設定する。モデムは即時に応答する必要がある
`"" AT&F1M0&M5S2=255`	`M0` – 接続中、スピーカーをオフに設定する
	`&M5` – モデムにエラー制御を要求させる
	S2=255 – TIES "+++" ブレークシーケンスを無効にする
`SAY "Calling myserver\n"`	ローカルマシン上に「Calling myserver (myserver を呼び出し中)」のメッセージを表示する
`TIMEOUT 60`	タイムアウトを 60 秒にリセットし、接続ネゴシエーションにより多くの時間を割り当てる
`OK "ATDT1-123-555-1212"`	電話番号 1-123-555-1212 を使ってリモートピアに発信する

スクリプトの内容	説明
ogin: pppuser	UNIX 方式のログインを使ってピアにログインする。ユーザー名 pppuser を指定
ssword: \q\U	\q – -v オプションを使ってデバッグする場合、ログをとらない \U – -U の後に続く文字列の内容を挿入する。文字列はコマンド行のここで指定する (通常パスワード)
% pppd	% シェルプロンプトを待ち、pppd コマンドを実行する

/etc/ppp/myisp-chat.tmpl chat スクリプトテンプレート

Solaris PPP 4.0 には、ユーザーが自分のサイトで使用するために変更できる /etc/ppp/myisp-chat.tmpl という chat スクリプトテンプレートが用意されています。/etc/ppp/myisp-chat.tmpl は、基本のモデム chat スクリプトと似ていますが、ログインシーケンスが含まれていません。

```
ABORT   BUSY
ABORT   'NO CARRIER'
REPORT  CONNECT
TIMEOUT 10
""      "AT&F1"
OK      "AT&C1&D2"
SAY     "Calling myisp\n"
TIMEOUT 60
OK      "ATDT1-123-555-1212"
CONNECT \c
```

スクリプトの内容	意味
ABORT BUSY	モデムが反対側のピアからこのメッセージを受け取った場合、伝送を中止する
ABORT 'NO CARRIER	ダイアル時にモデムが 'NO CARRIER' を報告した場合、伝送を中止する。このメッセージは、通常、ダイアルまたはモデムのネゴシエーションが失敗したときに発生する
REPORT CONNECT	CONNECT 文字列をモデムから収集し、表示する
TIMEOUT 10	初期タイムアウトを 10 秒に設定する。 モデムは即時に応答する必要がある
"" "AT&F1"	モデムを出荷時のデフォルトにリセット

スクリプトの内容	意味
OK "AT&C1&D2"	モデムをリセットする。その結果、&C1 では、モデムからの DCD がキャリアを追跡する。リモート側がなんらかの理由で電話を切った場合、DCD はドロップする &D2 では、DTR の High-Low 遷移により、モデムが停止する
SAY "Calling myisp\n"	ローカルマシン上に「Calling myisp (myisp を呼び出し中)」のメッセージを表示する
TIMEOUT 60	タイムアウトを 60 秒にリセットし、接続ネゴシエーションにより多くの時間を割り当てる
OK "ATDT1-123-555-1212"	電話番号 1-123-555-1212 を使ってリモートピアに発信する
CONNECT \c	反対側のピアのモデムからの CONNECT メッセージを待つ

ISP を呼び出すためのモデムの chat スクリプト

ダイアルアウトマシンから US Robotics Courier モデムを使用して ISP を呼び出すには、テンプレートとして次の chat スクリプトを使用します。

```
ABORT   BUSY
ABORT   'NO CARRIER'
REPORT  CONNECT
TIMEOUT 10
"" AT&F1M0&M5S2=255
SAY     "Calling myisp\n"
TIMEOUT 60
OK      "ATDT1-123-555-1212"
CONNECT \c
\r \d\c
SAY "Connected; running PPP\n"
```

次の表では、chat スクリプトの内容を説明します。

スクリプトの内容	説明
ABORT BUSY	モデムが反対側のピアからこのメッセージを受け取った場合、伝送を中止する
ABORT 'NO CARRIER'	モデムが反対側のピアからこのメッセージを受け取った場合、伝送を中止する
REPORT CONNECT	CONNECT 文字列をモデムから収集し、表示する
TIMEOUT 10	初期タイムアウトを 10 秒に設定する。 モデムは即時に応答する必要がある

スクリプトの内容	説明
"" AT&F1M0M0M0M0&M5S2=255	M0 – 接続中、スピーカーをオフに設定する &M5 – モデムにエラー制御を要求させる S2=255 – TIES "+++" ブレークシーケンスを無効にする
SAY "Calling myisp\n"	ローカルマシン上に「Calling myisp (myisp を呼び出し中)」のメッセージを表示する
TIMEOUT 60	タイムアウトを 60 秒にリセットし、接続ネゴシエーションにより多くの時間を割り当てる
OK "ATDT1-123-555-1212"	電話番号 1-123-555-1212 を使ってリモートピアに発信する
CONNECT \c	反対側のピアのモデムからの CONNECT メッセージを待つ
\r \d\c	CONNECT メッセージの最後まで待つ
SAY "Connected; running PPP\n"	ローカルマシン上に「Connected; running PPP (接続完了。PPP を実行中)」という通知メッセージを表示する

UNIX 方式ログイン用に拡張された基本の chat スクリプト

次の chat スクリプトは、Solaris のリモートピアまたは他の UNIX タイプのピアを呼び出すために基本のスクリプトを拡張したものです。この chat スクリプトは、474 ページの「ピアを呼び出すための命令群を作成する方法」で使用されています。

```
SAY "Calling the peer\n"
TIMEOUT 10
ABORT BUSY
ABORT 'NO CARRIER'
ABORT ERROR
REPORT CONNECT
"" AT&F1&M5S2=255
TIMEOUT 60
OK ATDT1-123-555-1234
CONNECT \c
SAY "Connected; logging in.\n"
TIMEOUT 5
ogin:--ogin: pppuser
TIMEOUT 20
ABORT 'ogin incorrect'
ssword: \qmypassword
"% " \c
SAY "Logged in.  Starting PPP on peer system.\n"
ABORT 'not found'
"" "exec pppd"
~ \c
```

次の表では、chat スクリプトのパラメータを説明します。

スクリプトの内容	説明
`TIMEOUT 10`	初期タイムアウトを 10 秒に設定する。 モデムは即時に応答する必要がある
`ABORT BUSY`	モデムが反対側のピアからこのメッセージを受け取った場合、伝送を中止する
`ABORT 'NO CARRIER'`	モデムが反対側のピアからこのメッセージを受け取った場合、伝送を中止する
`ABORT ERROR`	モデムが反対側のピアからこのメッセージを受け取った場合、伝送を中止する
`REPORT CONNECT`	`CONNECT` 文字列をモデムから収集し、表示する
`"" AT&F1&M5S2=255`	`&M5` – モデムにエラー制御を要求させる S2=255 – TIES "+++" ブレークシーケンスを無効にする
`TIMEOUT 60`	タイムアウトを 60 秒にリセットし、接続ネゴシエーションにより多くの時間を割り当てる
`OK ATDT1-123-555-1234`	電話番号 1-123-555-1212 を使ってリモートピアに発信する
`CONNECT \c`	反対側のピアのモデムからの `CONNECT` メッセージを待つ
`SAY "Connected; logging in.\n"`	ユーザーの状態を知らせるために、「Connected; logging in (接続完了。ログイン中)」という通知メッセージを表示する
`TIMEOUT 5`	タイムアウトを変更し、ログインプロンプトを迅速に表示できるようにする
`ogin:--ogin: pppuser`	ログインプロンプトを待つ。ログインプロンプトを受け取らなかった場合は、RETURN を送信して待機する。次にユーザー名 `pppuser` をピアに送信する。この後に続くシーケンスは、ほとんどの ISP から PAP ログインと呼ばれているが、PAP 認証とはまったく無関係である
`TIMEOUT 20`	タイムアウトを 20 秒に変更し、パスワードの検証により多くの時間をかけられるようにする
`ssword: \qmysecrethere`	ピアからのパスワードプロンプトを待つ。プロンプトを受け取ると、パスワード `\qmysecrethere` を送信する。`\q` は、パスワードがシステムログファイルに書き込まれるのを防ぐ
`"% " \c`	ピアからのシェルプロンプトを待つ。 chat スクリプトは C シェルを使用する。ユーザーが異なるシェルを使ってログインすることを希望する場合は、この値を変更する
`SAY "Logged in. Starting PPP on peer system.\n"`	「Logged in. Starting PPP on peer system (ログイン完了。ピアシステム上で PPP を開始中)」という通知メッセージを表示してユーザーに状態を通知する
`ABORT 'not found'`	シェルがエラーに遭遇した場合、伝送を中止する

スクリプトの内容	説明
`"" "exec pppd"`	ピア上で pppd を起動する
`~ \c`	PPP がピア上で開始するのを待つ

ISP は、`CONNECT \c` の直後に PPP を開始することをしばしば「PAP ログイン」といいます。しかし、実際には、PAP ログインは PAP 認証とは無関係です。

`ogin:--ogin: pppuser` 句は、ダイアルインサーバーから受け取ったログインプロンプトに対してユーザー名 (この例では pppuser) を送信するようにモデムに指示します。`pppuser` は、ダイアルインサーバー上のリモートユーザー `user1` 用に作成された専用の PPP ユーザーアカウント名です。ダイアルインサーバー上に PPP ユーザーアカウントを作成する方法については、479 ページの「ダイアルインサーバーのユーザーを構成する方法」を参照してください。

外部 ISDN TA 用 chat スクリプト

次は、ダイアルアウトマシンから ZyXEL omni.net. ISDN TA を使って呼び出すための chat スクリプトです。

```
SAY "Calling the peer\n"
TIMEOUT 10
ABORT BUSY
ABORT 'NO CARRIER'
ABORT ERROR
REPORT CONNECT
"" AT&FB40S83.7=1&K44&J3X7S61.3=1S0=0S2=255
OK ATDI18882638234
CONNECT \c
\r \d\c
SAY "Connected; running PPP\n"
```

次の表では、chat スクリプトのパラメータを説明します。

スクリプトの内容	説明
`SAY "Calling the peer"`	ダイアルアウトマシンの画面上にこのメッセージを表示する
`TIMEOUT 10`	初期タイムアウトを 10 秒に設定する
`ABORT BUSY`	モデムが反対側のピアからこのメッセージを受け取った場合、伝送を中止する
`ABORT 'NO CARRIER'`	モデムが反対側のピアからこのメッセージを受け取った場合、伝送を中止する
`ABORT ERROR`	モデムが反対側のピアからこのメッセージを受け取った場合、伝送を中止する

スクリプトの内容	説明
REPORT CONNECT	CONNECT 文字列をモデムから収集し、表示する
"" AT&FB40S83.7= 1&K44&J3X7S61.3=1 S0=0S2=255	この行内の文字は、以下を意味する ■ &F – 出荷時のデフォルトを使用する ■ B40 – 非同期 PPP 会話を行う ■ S83.7=1 – スピーチベアラのデータを使用する ■ &K44 – CCP 圧縮を有効にする ■ &J3 – MP を有効にする ■ X7 – DCE 側のレートを報告する ■ S61.3=1 – パケット断片化を使用する ■ S0=0 – 自動応答なし ■ S2=255 – TIES エスケープを無効にする
OK ATDI18882638234	ISDN 呼び出しを行う。マルチリンクでは、2 番目の呼び出しは、同じ電話番号に対して行われる。これは、通常、ほとんどの ISP の条件である。リモートピアが 2 番目の電話番号に異なる番号を要求する場合は、「+ *nnnn*」(*nnnn* は 2 番目の電話番号を表す) を付け加える
CONNECT \c	反対側のピアのモデムからの CONNECT メッセージを待つ
\r \d\c	CONNECT メッセージの最後まで待つ
SAY "Connected; running PPP\n"	ダイアルアウトマシンの画面上にこのメッセージを表示する

chat スクリプトのオプションの説明およびその他の詳細な情報については、chat（1M）のマニュアルページを参照してください。送信予期文字列の説明については、601 ページの「UUCP Chat-Script フィールド」を参照してください。

その他の chat スクリプト例の参照先

数多くの Web サイトで、chat スクリプトのサンプルとスクリプト作成のヒントが提供されています。

オーストラリア国立大学の Web サイトから利用できる PPP FAQ (Frequently Asked Questions) のページ (*URL*) も参考になります。

chat スクリプトの呼び出し

chat スクリプトを呼び出すには、connect オプションを使用します。PPP 構成ファイルまたはコマンド行で connect "chat ..." を使用できます。

chat スクリプトは実行可能ではありませんが、connect によって呼び出されるプログラムは実行可能でなければなりません。呼び出されるプログラムに chat ユーティリティを使用し、-f オプションを使用して chat スクリプトを外部ファイルに保存する場合、chat スクリプトファイルは実行可能ではありません。

chat(1m) で説明されている chat プログラムは、実際の chat スクリプトを実行します。pppd デーモンは、pppd が connect "chat ..." オプションに遭遇すると必ず、chat プログラムを起動します。

注 – Perl や Tcl などの外部プログラムを使って機能を拡張した chat スクリプトを作成することもできます。Solaris PPP 4.0 で chat ユーティリティが提供されているのは、ユーザーの便宜を図るためです。

▼ chat スクリプトを呼び出す方法 (手順)

1. **ASCII** ファイル形式で **chat** スクリプトを作成します。
2. 次の構文を使用して、任意の **PPP** 構成ファイル内で **chat** スクリプトを呼び出します。

   ```
   connect 'chat  -f /etc/ppp/chatfile'
   ```

   -f フラグは、ファイル名が後に続くことを示します。*/etc/ppp/chatfile* は、chat ファイルの名前を表します。
3. 外部 **chat** ファイルの読み取り権を pppd コマンドを実行するユーザーに与えます。

注意 – connect 'chat ...' オプションが特権ソースから呼び出された場合でも、chat プログラムは、常にユーザーの権限と連携して実行します。従って、-f オプションを使って読み取る個別の chat ファイルは、それを呼び出すユーザーが読み取り権を持っている必要があります。chat スクリプトにパスワードやその他の機密情報が含まれる場合、この特権はセキュリティの問題にかかわる可能性があります。

外部ファイル内の chat スクリプト

特定のピアで必要な chat スクリプトが長くて複雑な場合は、スクリプトを別ファイルとして作成することを考えます。外部 chat ファイルは、簡単に維持、作成できます。ハッシュ記号 (#) の後に続けて chat ファイルについてのコメントを追加できます。

外部ファイルに含まれる chat スクリプトの使用については、474 ページの「ピアを呼び出すための命令群を作成する方法」の手順を参照してください。

インライン chat スクリプト

次に示すように、chat スクリプトの全会話を 1 つの行に入れることができます。

```
connect 'chat "" "AT&F1" OK ATDT5551212 CONNECT "\c"'
```

chat キーワードの後から "\c"' までの句は、完全な chat スクリプトです。この形式は、pppd の引数として、PPP 構成ファイルまたはコマンド行で使用できます。

実行可能な chat ファイルの作成

実行可能なスクリプトの chat ファイルを作成して、ダイアルアップリンクが開始されたときに自動的に実行されるようにできます。これにより、接続開始時に、従来の chat スクリプトに含まれるコマンドのほかに、パリティー設定のための `stty` のような追加コマンドを実行できます。

この実行可能な chat スクリプトは、7 ビット長 / 偶数パリティーを要求する旧スタイルの UNIX システムにログインし、PPP 実行時に 8 ビット長 / パリティーなしに移行します。

```
#!/bin/sh
chat "" "AT&F1" OK "ATDT555-1212" CONNECT "\c"
stty evenp
chat ogin: pppuser ssword: "\q\U" % "exec pppd"
stty -evenp
```

▼ 実行可能な chat プログラムを作成する方法

1. テキストエディタを使用して、前述の例のような実行可能な **chat** プログラムを作成します。
2. **chat** プログラムを実行可能にします。

   # **chmod +x /etc/ppp/***chatprogram*

3. **chat** プログラムを呼び出します。

   `connect /etc/ppp/`*chatprogram*

   chat プログラムの場所は、`/etc/ppp` ファイルシステム内である必要はありません。任意の場所に保存できます。

接続時の呼び出し元の認証

この節では、PPP 認証プロトコルの動作と関連するデータベースについて説明します。

パスワード認証プロトコル (PAP)

PAP 認証は、UNIX の `login` プログラムと動作が多少似ていますが、ユーザーにシェルアクセスを許可しない点が異なります。PAP は、PPP 構成ファイルと `/etc/ppp/pap-secrets` ファイルの形式の PAP データベースを使って認証を設定

し、PAP セキュリティ資格を定義します。この資格には、ピア名 (PAP の用語では「ユーザー名」)、パスワード、ローカルマシンへの接続を許可されている呼び出し元に関する情報が含まれます。PAP のユーザー名とパスワードは、パスワードデータベース内の UNIX ユーザー名およびパスワードと同じものにすることも、違うものにすることもできます。

/etc/ppp/pap-secrets ファイル

PAP データベースは、/etc/ppp/pap-secrets ファイルに実装されています。 認証が成功するためには、PPP リンクの両側にある各マシンで、/etc/ppp/pap-secrets ファイル内に適切に設定された PAP 資格が必要です。 呼び出し元 (認証される側) は、/etc/ppp/pap-secrets ファイルまたは旧バージョンの +ua ファイルの user 列および password 列で資格を提供します。サーバー (認証する側) は、UNIX の passwd データベースまたは PAM 機能により /etc/ppp/pap-secrets 内の情報と対照してこの資格の妥当性を検証します。

/etc/ppp/pap-secrets ファイルの構文は、次のとおりです。

表 **36–5** /etc/ppp/pap-secrets の構文

呼び出し元	サーバー	パスワード	**IP** アドレス
myclient	ISP-server	mypassword	*

パラメータの意味は次のとおりです。

myclient	呼び出し元の PAP ユーザー名。 この名前は、呼び出し元の UNIX ユーザー名と同じ場合がある。特に、ダイアルインサーバーが PAP の login オプションを使用する場合は、同じ場合が多い。
ISP-server	リモートマシンの名前。ダイアルインサーバーである場合がしばしばある
mypassword	呼び出し元の PAP パスワード
IP address	呼び出し元に関連付けられている IP アドレス。 任意の IP アドレスを表すには、アスタリスク (*) を使用する

PAP パスワードの作成

PAP パスワードは、接続上をクリアテキストで (読み取り可能な ASCII 形式で) 送信されます。呼び出し元 (認証される側) では、PAP パスワードを次のどこかにクリアテキストで格納する必要があります。

- /etc/ppp/pap-secrets ファイル内
- 別の外部ファイル内

- pap-secrets @ 機能による名前付きパイプ内
- pppd のオプションとして、コマンド行上または PPP 構成ファイル内のどちらか
- +ua ファイルを介して

サーバー (認証する側) では、PAP パスワードは、次のどれかの方法で隠すことができます。

- pap-secrets ファイル内で papcrypt を指定し、crypt(3C) によってハッシュ化されたパスワードを使用する
- pppd に login オプションを指定し、パスワード列に二重引用符 ("") を入れることにより pap-secrets ファイルからパスワードを除外するこの場合、認証は UNIX の passwd データベースまたは pam (3PAM) メカニズムを利用して行われる。

PAP 認証時の動作

PAP 認証は、次の順序で発生します。

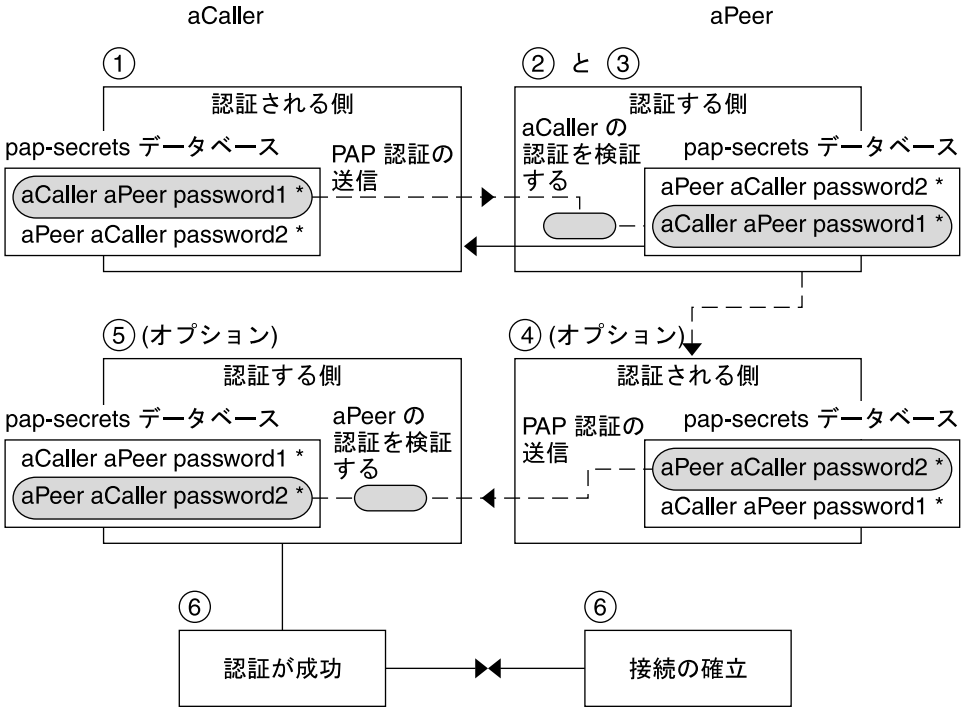


図 **36–1** PAP 認証処理

1. 呼び出し元 (認証される側) がリモートピア (認証する側) を呼び出し、接続ネゴシエーションの一環として PAP ユーザー名とパスワードを伝えます。

2. ピアは、/etc/ppp/pap-secrets ファイルで呼び出し元の識別情報を検証します。PAP の login オプションを使用する場合は、呼び出し元のユーザー名とパスワードの検証にパスワードデータベースが使用されます。
3. 認証が成功すると、ピアは呼び出し元との接続ネゴシエーションを継続します。認証に失敗すると、接続は切られます。
4. (オプション) 呼び出し元がリモートピアからの応答を認証する場合は、リモートピアが自身の PAP 資格を呼び出し元に送信する必要があります。したがって、リモートピアは認証される側になり、呼び出し側は認証する側になります。
5. (オプション) 最初の呼び出し元が自身の /etc/ppp/pap-secrets を読み取り、リモートピアの識別情報を検証します。

注 – 最初の呼び出し元がリモートピアに認証資格を要求する場合は、手順 1 と手順 4 が並行して行われます。

ピアが認証されると、ネゴシエーションが継続されます。認証されない場合は、接続が切られます。

6. 呼び出し元とピアのネゴシエーションは、接続の確立に成功するまで継続されます。

/etc/ppp/pap-secrets での login オプションの使用

PAP 資格を認証するための login オプションを PPP 構成ファイルに追加できます。たとえば /etc/ppp/options で login を指定した場合、pppd は呼び出し元の PAP 資格が Solaris のパスワードデータベース内に存在するかどうかを検証します。次の表に、login オプションを追加した /etc/ppp/pap-secrets ファイルの形式を示します。

表 **36–6** login オプションを追加した /etc/ppp/pap-secrets

呼び出し元	サーバー	パスワード	**IP** アドレス
joe	*	" "	*
sally	*	" "	*
sue	*	" "	*

パラメータの意味は次のとおりです。

呼び出し元	すべての承認された呼び出し元の名前

サーバー	アスタリスクは、任意のサーバー名が有効であることを示す。 `name` オプションは PPP 構成ファイルでは必須ではない
パスワード	二重引用符は、任意のパスワードが有効であることを示す。 この列にパスワードを入力した場合、ピアが提供するパスワードは、PAP パスワードと UNIX `passwd` データベースの両方に一致しなければならない
IP アドレス	アスタリスクは、任意の IP アドレスが許可されることを示す。

チャレンジハンドシェーク認証プロトコル (CHAP)

CHAP 認証は、チャレンジと応答という概念を使用します。つまり、ピア (認証する側) は識別情報を証明するために呼び出し元 (認証される側) にチャレンジします。チャレンジには、乱数、および認証する側によって生成された一意の ID が含まれます。呼び出し元は、ID、乱数、および呼び出し元の CHAP セキュリティ資格を使って適切な応答 (ハンドシェーク) を生成しピアに送信します。

CHAP セキュリティ資格には、CHAP ユーザー名と CHAP シークレットが含まれます。シークレットは、PPP リンクネゴシエーションを行う前に、あらかじめ呼び出し元とピアの両方が知っている任意の文字列です。CHAP セキュリティ資格は、CHAP データベース `/etc/ppp/chap-secrets` 内で設定します。

`/etc/ppp/chap-secrets` ファイル

CHAP データベースは、`/etc/ppp/chap-secrets` ファイルに実装されています。認証が成功するためには、PPP リンクの両側にある各マシンで、`/etc/ppp/chap-secrets` ファイル内に互いのマシンの CHAP 資格が必要です。

注 – PAP と異なり、共有シークレットは、両方のピアでクリアテキストでなければなりません。CHAP では、crypt、PAM、または PPP ログインオプションは使用できません。

`/etc/ppp/chap-secrets` ファイルの構文は、次のとおりです。

表 **36–7** /etc/ppp/chap-secrets の構文

呼び出し元	サーバー	**CHAP** シークレット	**IP** アドレス
myclient	myserver	secret5748	*

パラメータの意味は次のとおりです。

myclient	呼び出し元の CHAP ユーザー名。 呼び出し元の UNIX ユーザー名と同じ名前にすることも、違う名前にすることもできる
myserver	リモートマシンの名前。ダイアルインサーバーである場合がしばしばある
secret5748	呼び出し元の CHAP シークレット **注 –** PAP パスワードと異なり、CHAP シークレットは送信されない。ローカルマシンが応答を処理するときに使用される。
IP address	呼び出し元に関連付けられている IP アドレス。 任意の IP アドレスを表すには、アスタリスク (*) を使用する

CHAP 認証時の動作

CHAP 認証は、次の順序で発生します。

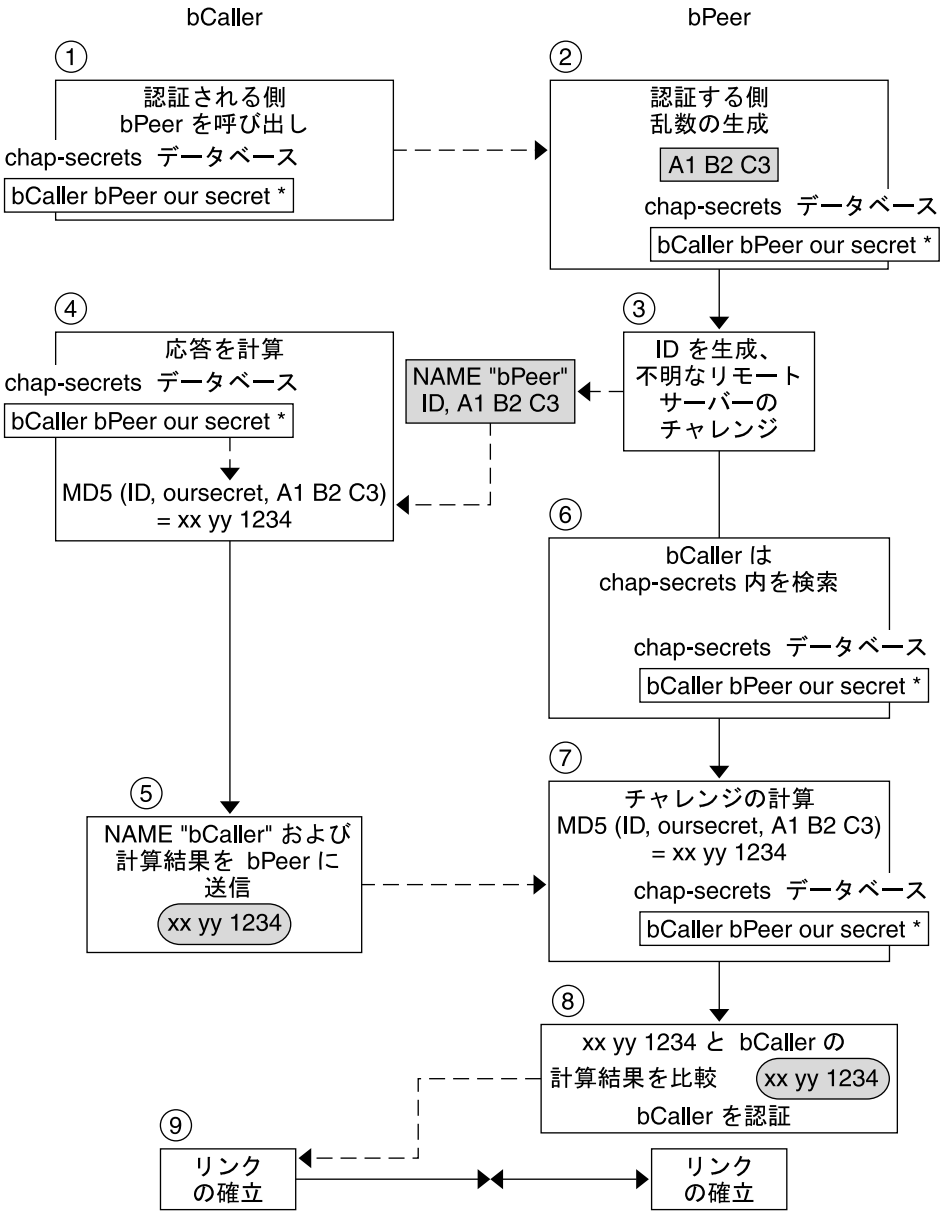


図 **36–2** CHAP 認証手順

1. 通信を開始しようとする 2 つのピアが、PPP リンクのネゴシエーション時に認証に使用するシークレットについて合意します。
2. 両方のマシンの管理者が、シークレット、CHAP ユーザー名、その他の CHAP 資格をそれぞれのマシンの `/etc/ppp/chap-secrets` データベースに追加します。

3. 呼び出し元 (認証される側) がリモートピア (認証する側) を呼び出します。
4. 認証する側が乱数と ID を生成し、それらを認証される側にチャレンジとして送信します。
5. 認証される側は、/etc/ppp/chap-secrets データベース内でピアの名前とシークレットを調べます。
6. 認証される側は、シークレットとピアの乱数チャレンジに MD5 計算アルゴリズムを適用することにより、応答を計算します。次に、認証される側は、認証する側に結果を応答として送信します。
7. 認証する側は、/etc/ppp/chap-secrets データベース内で認証される側の名前とシークレットを調べます。
8. 認証する側は、チャレンジとして生成された数値と /etc/ppp/chap-secrets 内の認証される側のシークレットに MD5 を適用することにより、自身の数値を計算します。
9. 認証する側は、呼び出し元からの応答と結果を比較します。2 つの数字が同じ場合、ピアは、呼び出し元の認証に成功し、接続ネゴシエーションが続けられます。認証されない場合は、接続が切られます。

呼び出し元の IP アドレス指定スキーマの作成

リモートユーザーごとに一意の IP アドレスを割り当てる代わりに、すべての着呼のために 1 つ以上の IP アドレスを作成することを考えます。専用 IP アドレスは、予想される呼び出し元の数が、ダイアルインサーバー上のシリアルポートとモデムの数を上回る場合、特に重要です。サイトの必要性に応じて、さまざまなシナリオを実現できます。さらに、シナリオは、相互に排他的ではありません。

呼び出し元への IP アドレスの動的割り当て

動的アドレス指定は、/etc/ppp/options.*ttyname* で定義されている IP アドレスを各呼び出し元に割り当てます。動的アドレス指定は、シリアルポート単位で発生します。特定のシリアル回線に呼が着信するたびに、呼び出しを処理するシリアルインタフェース用に /etc/ppp/options.*ttyname* ファイルで定義されている IP アドレスが呼び出し元に与えられます。

たとえば、ダイアルインサーバーに、着呼に対してダイアルアップサービスを提供するシリアルインタフェースが 4 つあると仮定します。

- シリアルポート term/a 用に、次のエントリがある /etc/ppp/options.term.a ファイルを作成します。

```
:10.1.1.1
```

- シリアルポート term/b 用に、次のエントリがある /etc/ppp/options.term.b ファイルを作成します。

```
:10.1.1.2
```

- シリアルポート term/c 用に、次のエントリがある /etc/ppp/options.term.c ファイルを作成します。

```
:10.1.1.3
```

- シリアルポート term/d 用に、次のエントリがある /etc/ppp/options.term.d ファイルを作成します。

```
:10.1.1.4
```

このアドレス指定スキーマでは、/dev/term/c のシリアルインタフェースに着信する呼び出しは、呼び出しを行っている間中、IP アドレス 10.1.1.3 が与えられます。最初の呼び出し元が回線を切った後、次にシリアルインタフェース /dev/term/c に着信する呼び出しも、IP アドレス 10.1.1.3 を与えられます。

動的アドレス指定には、次のような利点があります。

- PPP ネットワークの使用状況をシリアルポートまで追跡できる
- PPP 使用で割り当てる IP アドレスの数を最小限にできる
- IP フィルタリングをより簡単に管理できる

呼び出し元への IP アドレスの静的割り当て

サイトが PPP 認証を実装する場合は、個々の呼び出し元に特定の静的 IP アドレスを割り当てることができます。この場合、ダイアルアウトマシンがダイアルインサーバーを呼び出すたびに、呼び出し元は同じ IP アドレスを受け取ります。

静的アドレスは、pap-secrets または chap-secrets のどちらかのデータベースで実装します。以下は、静的 IP アドレスを定義した /etc/ppp/pap-secrets ファイルの例です。

呼び出し元	サーバー	パスワード	IP アドレス
joe	myserver	joepasswd	10.10.111.240
sally	myserver	sallypasswd	10.10.111.241
sue	myserver	suepasswd	10.10.111.242

以下は、静的 IP アドレスを定義した /etc/ppp/chap-secrets ファイルの例です。

呼び出し元	サーバー	CHAP シークレット	IP アドレス
account1	myserver	secret5748	10.10.111.244
account2	myserver	secret91011	10.10.111.245

sppp ユニット番号による IP アドレスの割り当て

PAP 認証または CHAP 認証を使用している場合は、sppp ユニット番号を使って IP アドレスを呼び出し元に割り当てることができます。次の表に、この方法の例を示します。

呼び出し元	サーバー	パスワード	IP アドレス
myclient	ISP-server	mypassword	10.10.111.240/28+

正符号 (+) は、ユニット番号が IP アドレスに追加されていることを示します。アドレス 10.10.111.240 から 10.10.111.255 までがリモートユーザーに割り当てられます。sppp0 は IP アドレス 10.10.111.240 を取得します。sppp1 は IP アドレス 10.10.111.241 を取得し、以下同様に続きます。

DSL サポート用の PPPoE トンネルの作成

PPPoE を使用することにより、1 台以上の DSL モデムを使用している複数のクライアントに PPP 超高速デジタルサービスを提供できます。PPPoE は、3 つの関係者、つまり企業、電話会社、サービスプロバイダを通して Ethernet トンネルを作成することにより、このサービスを実現します。

- PPPoE の動作の概要と説明については、449 ページの「PPPoE の概要」を参照してください。
- PPPoE トンネルの設定作業については、第 34 章を参照してください。

この節では、PPPoE コマンドおよびファイルについて詳しく説明します。概要を次の表に示します。

表 **36–8** PPPoE のコマンドと構成ファイル

ファイルまたはコマンド	説明	参照先
/etc/ppp/pppoe	PPPoE がシステムに設定したすべてのトンネルに対してデフォルトで適用される特性を含むファイル	568 ページの「/etc/ppp/pppoe ファイル」
/etc/ppp/pppoe.*device*	PPPoE がトンネルに使用する特定のインタフェースの特性を含むファイル	569 ページの「/etc/ppp/pppoe.*device* ファイル」
/etc/ppp/pppoe.if	PPPoE が設定したトンネルが動作する Ethernet インタフェースをリストしたファイル	565 ページの「/etc/ppp/pppoe.if ファイル」
/usr/sbin/sppptun	PPPoE トンネルに関係する Ethernet インタフェースを設定するためのコマンド	566 ページの「/usr/sbin/sppptun コマンド」
/usr/lib/inet/pppoed	PPPoE を使ってトンネルを設定するためのコマンドとオプション	567 ページの「/usr/lib/inet/pppoed デーモン」

PPPoE のインタフェースを設定するためのファイル

PPPoE トンネルの両端で使用されるインタフェースは、トンネルが PPP 通信をサポートする前に、あらかじめ設定しておく必要があります。設定には、/usr/sbin/sppptun および /etc/ppp/pppoe.if ファイルを使用します。これらのツールを使用して、すべての Solaris PPPoE クライアントおよびアクセスサーバー上の Ethernet インタフェースを設定する必要があります。

/etc/ppp/pppoe.if ファイル

/etc/ppp/pppoe.if ファイルは、ホスト上の PPPoE トンネルで使用されるすべての Ethernet インタフェースの名前をリストします。このファイルはシステムの起動時に処理され、ファイルにリストされているインタフェースは PPPoE トンネルで使用するために plumb されます。

/etc/ppp/pppoe.if は明示的に作成する必要があります。各行ごとにインタフェース名を 1 つずつ入力して PPPoE 用に設定します。

/etc/ppp/pppoe.if ファイルサンプル

次に、PPPoE トンネルに 3 つのインタフェースを提供するサーバーの /etc/ppp/pppoe.if ファイルの例を示します。

```
# cat /etc/ppp/pppoe.if
hme1
```

```
hme2
hme3
```

PPPoE クライアントは通常、/etc/ppp/pppoe.if にリストされているインタフェースを 1 つだけ使用します。

/usr/sbin/sppptun コマンド

/usr/sbin/sppptun コマンドを使用すると、PPPoE トンネルで使用する Ethernet インタフェースを手動で plumb したり unplumb したりできます。これに対して、/etc/ppp/pppoe.if はシステムの起動時だけ読み取られます。これらのインタフェースは、/etc/ppp/pppoe.if にリストされているインタフェースと一致する必要があります。

sppptun は、PPPoE トンネルで使用する Ethernet インタフェースを ifconfig コマンドと同様の方法で plumb します。ifconfig とは異なり、2 つの Ethernet プロトコル番号が必要なため、PPPoE をサポートするにはインタフェースを 2 回 plumb する必要があります。

sppptun の基本的な構文を次に示します。

```
# /usr/sbin/sppptun plumb pppoed device-name
      device-name:pppoed
# /usr/sbin/sppptun plumb pppoe device-name
      device-name:pppoe
```

この構文で、*device-name* は PPPoE に plumb されるデバイス名です。

上の 1 つめの sppptun コマンドを実行したときは、発見プロトコル pppoed がインタフェースに plumb されます。2 つめの sppptun を実行したときは、セッションプロトコル pppoe が plumb されます。sppptun は、plumb されたインタフェースの名前を表示します。必要な場合は、この名前を使ってインタフェースを unplumb します。

詳細は、sppptun(1M) のマニュアルページを参照してください。

インタフェースを管理するための sppptun コマンドサンプル

- 次の例は、/usr/sbin/sppptun を使用して PPPoE のインタフェースを手動で plumb します。

  例 **36–1** PPPoE をサポートするためにインタフェースを plumb するには

  ```
  # /usr/sbin/sppptun plumb pppoed hme0
  hme0:pppoed
   # /dev/sppptun plumb pppoe hme0
   hme0:pppoe
  ```

- 次の例は、PPPoE に plumb されたアクセスサーバー上のインタフェースを表示します。

例 **36–2** PPPoE アクセスサーバー上のすべてのインタフェースを表示するには

```
/usr/sbin/sppptun query
hme0:pppoe
hme0:pppoed
hme1:pppoe
hme1:pppoed
hme2:pppoe
hme2:pppoed
```

- 次の例は、インタフェースを unplumb します。

例 **36–3** PPPoE トンネルで使用しているインタフェースを unplumb するには

```
# sppptun unplumb hme0:pppoed
# sppptun unplumb hme0:pppoe
```

PPPoE アクセスサーバーのコマンドとファイル

DSL のサービスまたはサポートを顧客に提供するサービスプロバイダは、Solaris PPPoE を実行するアクセスサーバーを使用できます。PPPoE アクセスサーバーとクライアントは、従来のクライアントとサーバーの関係で機能します。この関係は、あるシステムが通信を開始し、別のシステムが応答するという点で、ダイアルアップリンクでのダイアルアウトマシンとダイアルインサーバーの関係に似ています。これに対して、PPP プロトコルにはクライアントとサーバーの関係という概念はなく、両方のマシンが同等のピアとみなされます。

PPPoE アクセスサーバーを設定するコマンドおよびファイルには、以下が含まれます。

- 566 ページの「`/usr/sbin/sppptun` コマンド」
- 567 ページの「`/usr/lib/inet/pppoed` デーモン」
- 568 ページの「`/etc/ppp/pppoe` ファイル」
- 569 ページの「`/etc/ppp/pppoe.`*device* ファイル」
- 573 ページの「`pppoe.so` プラグイン」

/usr/lib/inet/pppoed デーモン

`pppoed` デーモンは、将来の PPPoE クライアントからブロードキャストを受け取ります。さらに、`pppoed` は PPPoE トンネルのサーバー側とネゴシエーションし、PPP デーモン `pppd` をそのトンネル上で実行します。

`pppoed` サービスは、`/etc/ppp/pppoe` および `/etc/ppp/pppoe.`*device* ファイルで設定します。システム起動時に `/etc/ppp/pppoe` が存在する場合は、`pppoed` が自動的に実行します。コマンド行で `/usr/lib/inet/pppoed` と入力することにより、`pppoed` デーモンを明示的に実行することもできます。

/etc/ppp/pppoe ファイル

/etc/ppp/pppoe ファイルは、アクセスサーバーが提供するサービスと、PPP が PPPoE トンネル上でどのように実行するかを定義するオプションを説明します。 インタフェースごとに個別にサービスを定義することも、広域的にアクセスサーバー上のすべてのインタフェースに対してサービスを定義することもできます。アクセスサーバーは、将来の PPPoE クライアントからのブロードキャストに応答して、/etc/ppp/pppoe ファイル内の情報を送信します。

以下に、/etc/ppp/pppoe の基本的な構文を示します。

global-options
service service-name
    service-specific-options
    device interface-name

パラメータの意味は次のとおりです。

global-options	/etc/ppp/pppoe ファイルのデフォルトのオプションを設定する。このオプションには、pppoed または pppd で使用可能なオプションはすべて使用できる。オプションの完全なリストについては、pppoed(1M) および pppd(1M) のマニュアルページを参照 たとえば、グローバルオプションには、PPPoE トンネルで使用できる Ethernet インタフェースをリストする必要がある。/etc/ppp/pppoe でデバイスを定義しないと、インタフェースでサービスを提供できない devices をグローバルオプションとして定義するには、次の形式を使用する *device interface <,interface>* *interface* は、サービスが将来の PPPoE クライアントを待つインタフェースを指定する。複数のインタフェースがサービスに関連付けられている場合は、名前をコンマで区切って指定する
service service-name	*service-name* というサービスの定義を開始する。*service-name* には、提供されるサービスに適した任意の文字列を指定できる
service-specific-options	このサービスに固有の PPPoE および PPP のオプションを表示する
device *interface-name*	上記でリストしたサービスを利用できるインタフェースを指定する

/etc/ppp/pppoe のこの他のオプションについては、pppoed(1M) および pppd(1M) のマニュアルページを参照してください。

次に、典型的な /etc/ppp/pppoe ファイルの例を示します。

例 **36–4** 基本的な /etc/ppp/pppoe ファイル

```
device hme1,hme2,hme3
service internet
   pppd "name internet-server"
```

例 **36–4** 基本的な /etc/ppp/pppoe ファイル　(続き)

```
service intranet
   pppd "192.168.1.1:"
service debug
   device hme1
   pppd "debug name internet-server"
```

このファイルでは、以下が適用されています。

hme1,hme2,hme3	PPPoE トンネルに使用されるアクセスサーバー上の 3 つのインタフェース
service internet	将来のクライアントに対して internet というサービスを通知する。また、サービスを提供するプロバイダは internet の定義についても決定する。たとえば、プロバイダは、インターネットへのアクセスだけでなく、さまざまな IP サービスを意味する internet サービスを提供できる
pppd	呼び出し元が pppd を呼び出したときに使用されるコマンド行オプションを設定する。 "name internet-server" オプションは、ローカルマシン (アクセスサーバー) の名前を internet-server と付ける
service intranet	intranet という別のサービスを想定クライアントに通知する
pppd "192.168.1.1:"	呼び出し元が pppd を呼び出したときに使用されるコマンド行オプションを設定する。 呼び出し元が pppd を呼び出すと、ローカルマシン (アクセスサーバー) の IP アドレスとして 192.168.1.1 が設定される
service debug	PPPoE 用に定義されているインタフェースに 3 番目のサービス、デバッグを通知する
device hme1	PPPoE トンネルに対するデバッグを hme1 に限定する
pppd "debug name internet-server"	呼び出し元が pppd を起動したときに使用されるコマンド行オプション、この場合は PPP デバッグをローカルマシン internet-server に設定する

/etc/ppp/pppoe.*device* ファイル

/etc/ppp/pppoe.*device* ファイルは、PPPoE アクセスサーバーのインタフェース上で提供されるサービスと、PPP が PPPoE トンネル上でどのように実行するかを定義するオプションを説明します。 /etc/ppp/pppoe.*device* はオプションのファイルで、グローバルの /etc/ppp/pppoe とまったく同様に動作します。ただし、/etc/ppp/pppoe.*device* がインタフェース用に定義されている場合、そのインタフェースでは、このファイルのパラメータが、/etc/ppp/pppoe で定義されているグローバルパラメータより優先されます。

以下に、/etc/ppp/pppoe.*device* の基本的な構文を示します。

service *service-name*
service-specific-options
service *another-service-name*
service-specific-options

上記の構文と /etc/ppp/pppoe の構文の違いは、568 ページの「/etc/ppp/pppoe ファイル」 で示した device オプションを使用できない点だけです。

pppoe.so プラグイン

pppoe.so は PPPoE 共有オブジェクトファイルで、PPPoE のアクセスサーバーおよびクライアントによって呼び出されます。 このファイルは、MTU および MRU を 1492 に制限し、ドライバからのパケットにフィルタをかけ、pppoed とともに PPPoE トンネルをネゴシエートします。 アクセスサーバー側では、pppoe.so は pppd デーモンによって自動的に呼び出されます。

アクセスサーバー構成のための PPPoE および PPP ファイルの使用

この節では、あるアクセスサーバーを構成するために使用するすべてのファイルのサンプルを紹介します。このアクセスサーバーはマルチホームで、3 つのサブネット、green、orange、および purple が接続されています。 pppoed は、サーバー上で root として実行します。これはデフォルトの動作です。

PPPoE クライアントは、hme0 および hme1 インタフェースを通じて orange および purple ネットワークにアクセスできます。クライアントは、標準の UNIX ログインを使ってサーバーにログインします。サーバーは、クライアントを PAP を使って認証します。

green ネットワークは、クライアントに通知されません。クライアントが green にアクセスできるためには、直接「green-net」を指定し、CHAP 認証資格を提供しなければなりません。さらに、クライアント joe および mary だけが green ネットワークにアクセスできます。それには、彼らは静的 IP アドレスを使用する必要があります。

例 **36–5** アクセスサーバー用の /etc/ppp/pppoe ファイル

```
service orange-net
     device hme0,hme1
     pppd "require-pap login name orange-server orange-server:"
service purple-net
     device hme0,hme1
     pppd "require-pap login name purple-server purple-server:"
service green-net
     device hme1
     pppd "require-chap name green-server green-server:"
     nowildcard
```

このサンプルは、アクセスサーバーから使用できるサービスを説明します。1 番目の service 節は、orange ネットワークのサービスを説明します。

```
service orange-net
     device hme0,hme1
     pppd "require-pap login name orange-server orange-server:"
```

クライアントは、hme0 および hme1 インタフェース上で orange ネットワークにアクセスできます。pppd コマンドに指定されているオプションにより、サーバーは、想定クライアントからの PAP 資格を要求します。 また、pppd オプションはサーバーの名前を orange-server に設定します。この名前は pap-secrets ファイルで使用されます。

purple ネットワーク用の service 節は、ネットワーク名とサーバー名が異なる以外は、orange ネットワーク用の service 節と同じです。

次の service 節は、green ネットワークのサービスを説明します。

```
service green-net
     device hme1
     pppd "require-chap name green-server green-server:"
     nowildcard
```

この節は、クライアントのアクセスをインタフェース hme1 に限定しています。pppd コマンドに指定されているオプションにより、サーバーは、想定クライアントからの CHAP 資格を要求します。 また、pppd オプションはサーバー名を green-server に設定しています。この名前は chap-secrets ファイルで使用されます。nowildcard オプションは、green ネットワークの存在をクライアントに通知しないことを指定します。

このアクセスサーバーのシナリオでは、次のような /etc/ppp/options ファイルを設定する場合があります。

例 **36–6** アクセスサーバー用の /etc/ppp/options ファイル

```
     auth
     proxyarp
     nodefaultroute
     name no-service    # don't authenticate otherwise
```

name no-service オプションは、通常、PAP または CHAP 認証時に検索されるサーバー名を無効にします。サーバーのデフォルト名は、/usr/bin/hostname を使って得られます。前述の例の name オプションは、サーバー名を pap または chap-secrets ファイルでは見つかりそうにない名前 no-service に変更します。この処理により、任意のユーザーが pppd を実行したり、/etc/ppp/options で設定されている auth および name オプションを上書きするのを防ぐことができます。pppd は、no-service のサーバー名ではクライアントのシークレットを見つけることができないため、失敗します。

このアクセスサーバーのシナリオでは、次の /etc/hosts ファイルを使用します。

例 **36–7** アクセスサーバー用の /etc/hosts ファイル

```
     172.16.0.1    orange-server
     172.17.0.1    purple-server
```

例 **36–7** アクセスサーバー用の /etc/hosts ファイル　(続き)

```
172.18.0.1    green-server
172.18.0.2    joes-pc
172.18.0.3    marys-pc
```

次に、orange および purple ネットワークにアクセスしようとするクライアントの PAP 認証に使用する /etc/ppp/pap-secrets ファイルを示します。

例 **36–8** アクセスサーバー用の /etc/ppp/pap-secrets ファイル

```
* orange-server "" 172.16.0.2/16+
* purple-server "" 172.17.0.2/16+
```

次に、CHAP 認証に使用される /etc/ppp/chap-secrets ファイルを示します。joe および mary というクライアントだけがファイルにリストされていることに注意してください。

例 **36–9** アクセスサーバー用の /etc/ppp/chap-secrets ファイル

```
 joe green-server "joe's secret" joes-pc
mary green-server "mary's secret" marys-pc
```

PPPoE クライアントのコマンドとファイル

DSL モデム上で PPP を実行するには、マシンが PPPoE クライアントになる必要があります。PPPoE を実行するためにインタフェースを plumb し、次に pppoec ユーティリティを使ってアクセスサーバーの存在を「発見」する必要があります。その後、クライアントは DSL モデム上に PPPoE トンネルを作成し PPP を実行できます。

PPPoE クライアントは、従来のクライアント - サーバーモデルでアクセスサーバーに接続します。PPPoE トンネルはダイアルアップリンクではありませんが、ほぼ同じような方法で構成され、操作されます。

PPPoE クライアントを設定するコマンドおよびファイルには、以下が含まれます。

- 566 ページの「/usr/sbin/sppptun コマンド」
- 573 ページの「/usr/lib/inet/pppoec ユーティリティ」
- 573 ページの「pppoe.so プラグイン」
- 543 ページの「/etc/ppp/peers/*peer-name* ファイル」
- 538 ページの「/etc/ppp/options 構成ファイル」

/usr/lib/inet/pppoec ユーティリティ

/usr/lib/inet/pppoec ユーティリティは、PPPoE トンネルのクライアント側をネゴシエーションします。pppoec は、Solaris PPP 4.0 の chat ユーティリティに似ていますが、chat ユーティリティでは pppoec を直接起動しません。直接起動するのではなく、pppd の connect オプションの引数として /usr/lib/inet/pppoec を起動します。

pppoe.so プラグイン

pppoe.so は PPPoE 共有オブジェクトで、PPPoE によって読み込まれ、PPPoE 機能をアクセスサーバーとクライアントに提供します。 この共有オブジェクトは、MTU および MRU を 1492 に制限し、ドライバからのパケットにフィルタをかけ、実行時 PPPoE メッセージを処理します。

クライアント側では、ユーザーが plugin pppoe.so オプションを指定すると、pppd が pppoe.so を読み込みます。

アクセスサーバーピアを定義するための /etc/ppp/peers/*peer-name* ファイル

アクセスサーバーが pppoec によって発見されるように定義する場合は、 pppoec および pppd デーモンの両方に適用されるオプションを使用します。アクセスサーバーの /etc/ppp/peers/*peer-name* ファイルは次のパラメータを必要とします。

- sppptun – PPPoE トンネルが使用するシリアルデバイスの名前
- plugin pppoe.so – pppd に pppoe.so 共有オブジェクトを読み込むように指示する
- connect "/usr/lib/inet/pppoec *device*" – PPPoE に plumb されているインタフェース *device* 上で接続を開始し、pppoec ユーティリティを起動する

/etc/ppp/peers/ *peer-name* ファイル内の残りのパラメータは、サーバー上の *PPP* リンクに適用されます。ダイアルアウトマシン上の /etc/ppp/peers/*peer-name* と同じオプションを使用します。オプションの数を *PPP* リンクで必要な最小数に制限するようにしてください。

次の例は、509 ページの「PPPoE アクセスサーバーピアを定義する方法」で紹介されています。

例 **36–10** リモートアクセスサーバーを定義するための /etc/ppp/peers/*peer-name*

```
# vi /etc/ppp/peers/dslserve
sppptun
plugin pppoe.so
connect "/usr/lib/inet/pppoec hme0"
noccp
noauth
user Red
```

例 **36–10** リモートアクセスサーバーを定義するための /etc/ppp/peers/*peer-name*　（続き）

```
password redsecret
noipdefault
defaultroute
```

このファイルは、アクセスサーバー `dslserve` に PPPoE トンネルと PPP リンクを設定するときに使用するパラメータを定義します。オプションには、以下が含まれます。

オプション	説明
`sppptun`	`sppptun` をシリアルデバイスの名前として定義する
`plugin pppoe.so`	`pppd` に `pppoe.so` 共有オブジェクトを読み込むように指示する
`connect "/usr/lib/inet/pppoec hme0"`	`pppoec` を実行し、PPPoE トンネルおよび PPP リンク用のインタフェースとして `hme0` を指定する
`noccp`	接続上で CCP 圧縮をオフに設定する **注 –** なんらかの圧縮アルゴリズムを使用する場合、多くの ISP は独自の圧縮アルゴリズムだけを使用します。公開された CCP アルゴリズムをオフにすると、ネゴシエーションの時間を節約し、偶発的な相互運用性の問題を避けることができます。
`noauth`	`pppd` が認証資格をアクセスサーバーに要求するのを停止する。ほとんどの ISP は認証資格を顧客に提供しない
`user Red`	アクセスサーバーによる PAP 認証に必要なクライアントのユーザー名として `Red` の名前を設定する
`password redsecret`	PAP 認証のためにアクセスサーバーに提供されるパスワードとして `redsecret` を定義する
`noipdefault`	初期 IP アドレスとして 0.0.0.0 を割り当てる
`defaultroute`	IPCP ネゴシエーション後にデフォルトの IPv4 経路指定をインストールするよう `pppd` に指示する。接続がシステムのインターネットへの接続である場合、`/etc/ppp/peers/`*peer-name* 内に `defaultroute` を含める必要がある。PPPoE クライアントの場合これにあてはまる

第 37 章

非同期 Solaris PPP から Solaris PPP 4.0 への移行 (手順)

Solaris オペレーティングシステムの以前のバージョンでは、別の PPP 実装である非同期 Solaris PPP (asppp) が提供されていました。asppp を実行するピアを最新の PPP 4.0 に更新したい場合は、変換スクリプトを実行する必要があります。この章では、PPP 変換に関する次のトピックについて説明します。

この章では、サンプルの asppp 構成を使用して、PPP 変換を実施する方法について説明します。Solaris PPP 4.0 と asppp の相違点については、438 ページの「使用する Solaris PPP のバージョン」を参照してください。

asppp ファイルを変換する前に

変換スクリプト /usr/sbin/asppp2pppd を使用して、標準 asppp 構成を構成する次のファイルを変換できます。

- /etc/asppp.cf – 非同期 PPP 構成ファイル
- /etc/uucp/Systems – リモートピアの特性を記述する UUCP ファイル
- /etc/uucp/Devices – ローカルマシン上のモデムを記述する UUCP ファイル
- /etc/uucp/Dialers – /etc/uucp/Devices ファイルに記述されているモデムが使用するログインシーケンスが含まれる UUCP ファイル

asppp については、http://docs.sun.com に掲載されている「Solaris 8 System Administrator Collection」の『Solaris のシステム管理 (第 3 巻)』を参照してください。

例—/etc/asppp.cf 構成ファイル

579 ページの「asppp から Solaris PPP 4.0 に変換する方法」に示す手順は、次の /etc/asppp.cf ファイルを使用します。

```
#
ifconfig ipdptp0 plumb mojave gobi up

path
 inactivity_timeout 120     # Approx. 2 minutes
    interface ipdptp0
    peer_system_name Pgobi     # The name we log in with (also in
                                 # /etc/uucp/Systems
```

このファイルには次のパラメータが含まれています。

ifconfig ipdptp0 plumb mojave gobi up	ifconfig コマンドを実行し、ローカルマシン mojave の PPP インタフェース ipdptp0 からリモートピア gobi へのリンクを確立する
inactivity_timeout 120	2 分間アクティブでない回線を終了する
interface ipdptp0	ダイアルアウトマシン上のインタフェース ipdptp0 を非同期 PPP に構成する
peer_system_name Pgobi	リモートピアの名前 Pgobi を指定する

例—/etc/uucp/Systems ファイル

579 ページの「asppp から Solaris PPP 4.0 に変換する方法」に示す手順は、次の /etc/uucp/Systems ファイルを使用しています。

```
#ident  "@(#)Systems    1.5  92/07/14 SMI"  /* from SVR4 bnu:Systems 2.4 */
#

# .
# .
Pgobi Any ACU 38400 15551212 in:--in: mojave word: sand
```

このファイルには次のパラメータが含まれています。

Pgobi	Pgobi をリモートピアのホスト名として使用する
Any ACU	ダイアルアウトマシン mojave 上のモデムが、任意の時点で Pgobi 上のモデムとリンクを確立するようにする。Any ACU は「/etc/uucp/Devices ファイル内で ACU を探す」ことを意味する

38400	リンクの最大速度として 38400 を設定する
15551212	Pgobi の電話番号を指定する
in: -in: mojave word: sand	Pgobi が必要とするログインスクリプトを定義して、ダイアルアウトマシン mojave を認証する

例—/etc/uucp/Devices ファイル

579 ページの「asppp から Solaris PPP 4.0 に変換する方法」に示す手順は、次の /etc/uucp/Devices ファイルを使用します。

```
#ident  "@(#)Devices  1.6  92/07/14 SMI"  /* from SVR4 bnu:Devices 2.7 */

.
.
#

TCP,et - - Any TCP -
.
.
#
ACU cua/b - Any hayes
# 0-7 are on a Magma 8 port card
Direct cua/0 - Any direct
Direct cua/1 - Any direct
Direct cua/2 - Any direct
Direct cua/3 - Any direct
Direct cua/4 - Any direct
Direct cua/5 - Any direct
Direct cua/6 - Any direct
Direct cua/7 - Any direct
# a is the console port (aka "tip" line)
Direct cua/a - Any direct
# b is the aux port on the motherboard
Direct cua/b - Any direct
# c and d are high speed sync/async ports
Direct cua/c - Any direct
Direct cua/d - Any direct
```

このファイルは、シリアルポート cua/b に接続されている Hayes モデムをサポートします。

例—/etc/uucp/Dialers ファイル

579 ページの「asppp から Solaris PPP 4.0 に変換する方法」に示す手順は、次の /etc/uucp/Dialers ファイルを使用します。

```
#
#<この他にもSolaris UUCP でサポートされているモデムについての多くの情報があります。>
```

```
penril    =W-P    "" \d> Q\c : \d-> s\p9\c )-W\p\r\ds\p9\c-) y\c : \E\TP> 9\c OK
ventel    =&-%    "" \r\p\r\c $ k\c ONLINE!
vadic    =K-K    "" \005\p *-\005\p-*\005\p-* D\p BER? \E\T\e \r\c LINE
develcon ""    "" \pr\ps\c est:\007 \E\D\e \n\007
micom    ""    "" \s\c NAME? \D\r\c GO
direct
#
#
#
#   Hayes Smartmodem -- modem should be set with the configuration
#   switches as follows:
#
#       S1 - UP        S2 - UP        S3 - DOWN    S4 - UP
#       S5 - UP        S6 - DOWN    S7 - ?        S8 - DOWN
#
hayes    =,-,    "" \dA\pTE1V1X1Q0S2=255S12=255\r\c OK\r \EATDT\T\r\c CONNECT
```

<この他にもSolaris UUCP でサポートされているモデムについての多くの情報があります。>

このファイルには、あらゆるタイプのモデムの chat スクリプトが含まれます。/etc/uucp/Dialers ファイルでサポートされている Hayes モデムの chat スクリプトも含まれます。

asppp2pppd 変換スクリプトの実行 (作業)

/usr/sbin/asppp2pppd スクリプトは、/etc/asppp.cf に含まれる PPP 情報と PPP 関連の UUCP ファイルを、Solaris PPP 4.0 ファイル内の適切な場所にコピーします。

前提条件

次の作業に進む前に、以下のことを完了しておく必要があります。

- asppp と UUCP 構成ファイルがあるマシン上に Solaris 9 オペレーティング環境をインストールする
- PPP ファイルがあるマシン、たとえば mojave 上でスーパーユーザーになる

▼ asppp から Solaris PPP 4.0 に変換する方法

1. 変換スクリプトを実行します。

   ```
   # /usr/sbin/asppp2pppd
   ```

   変換処理が開始し、画面に次のようなメッセージが表示されます。

```
This script provides only a suggested translation for your existing aspppd
configuration.  You will need to evaluate for yourself whether the translation
is appropriate for your operating environment.
Continue [Yn]?
```

2. **Y** と入力してスクリプトの実行を継続します。画面に次のようなメッセージが表示されます。

   ```
   Chat cannot do echo checking; requests for this removed.
   Adding 'noauth' to /etc/ppp/options

   Preparing to write out translated configuration:
       1 chat file:
           1.  /etc/ppp/chat.Pgobi.hayes
       2 option files:
           2.  /etc/ppp/peers/Pgobi
           3.  /etc/ppp/options
       1 script file:
           4.  /etc/ppp/demand
   ```

   新しい Solaris PPP 4.0 ファイルが生成されました。

▼ 変換結果を表示する方法

変換処理の最後に、/usr/sbin/asppp2pppd 変換スクリプトによって作成された Solaris PPP 4.0 ファイルを表示できます。以下に示すオプションリストが表示されます。

```
Enter option number:
        1 - view contents of file on standard output
        2 - view contents of file using /usr/bin/less
        3 - edit contents of file using /usr/bin/vi
        4 - delete/undelete file from list
        5 - rename file in list
        6 - show file list again
        7 - escape to shell (or "!")
        8 - abort without saving anything
        9 - save all files and exit (default)
Option:
```

1. **1** を入力して、画面上にファイルの内容を表示します。

   表示するファイルの番号の入力を求めるプロンプトが表示されます。

   ```
   File number (1 .. 4):
   ```

この番号は、前述の手順 2 で示したように、変換処理中に表示された変換ファイルを示します。

2. **1** を入力して、**chat** ファイル /etc/ppp/chat.Pgobi.hayes を表示します。

```
File number (1 .. 4):  1
"" \d\dA\p\pTE1V1X1Q0S2=255S12=255\r\c
OK\r ATDT\T\r\c
CONNECT \c
in:--in: mojave
word: sand
```

chat スクリプトには、サンプルの /etc/uucp/Dialers ファイルの hayes 行に記述されているモデムの “chat” 情報が含まれています。また、/etc/ppp/chat.Pgobi.hayes にはサンプルの /etc/uucp/Systems に記述されている Pgobi のログインシーケンスが含まれています。したがって、現時点では、chat スクリプトは /etc/ppp/chat.Pgobi.hayes ファイルにあります。

3. **2** を入力して、ピアファイル /etc/ppp/peers/Pgobi を表示します。

```
File number (1 .. 4):  2
/dev/cua/b
38400
demand
idle 120
connect "/usr/bin/chat -f /etc/ppp/chat.Pgobi.hayes -T '15551212'"
user NeverAuthenticate
mojave:gobi
```

/etc/uucp/Devices ファイル内のシリアルポート情報 (/dev/cua/b) と、/etc/asppp.cf ファイル内のリンク速度、アイドル時間、認証情報、ピア名が表示されています。“demand” は “demand” スクリプトを意味します。このスクリプトは、ダイアルアウトマシンがピア Pgobi に接続を試みるときに呼び出されます。

4. **3** を入力して、ダイアルアウトマシン mojave 用に作成された /etc/ppp/options ファイルを表示します。

```
File number (1 .. 4):  3
#lock
noauth
```

/etc/ppp/options ファイル内の情報は /etc/asppp.cf ファイルから得られたものです。

5. **4** を入力して、demand スクリプトの内容を表示します。

```
File number (1 .. 4):  4
/usr/bin/pppd file /etc/ppp/peers/Pgobi
```

このスクリプトが実行されると、pppd コマンドが実行されます。このコマンドは、/etc/ppp/peers/Pgobi を読み込んで、mojave と Pgobi の間のリンクを確立します。

6. **9** を入力して、作成したファイルを保存し、変換スクリプトを終了します。

第 38 章

UUCP (概要)

この章では、 UNIX 間コピープログラム (UUCP) とデーモンについて説明します。この節の内容は次のとおりです。

UUCP を使用すると、コンピュータシステム間で相互にファイルの転送とメールの交換を行えます。また、UUCP を使用して Usenet のような大規模なネットワークにコンピュータを接続することもできます。

Solaris 環境では、HoneyDanBer UUCP とも呼ばれる基本ネットワークユーティリティ (BNU) バージョンの UUCP が提供されています。UUCP という用語はシステムを形成するすべてのファイルとユーティリティを意味するものであり、uucp プログラムはそのシステムの一部にすぎません。UUCP のユーティリティには、コンピュータ間でファイルをコピーするためのユーティリティ (uucp と uuto) から、リモートログインやリモートコマンド実行のためのユーティリティ (cu と uux) まで、さまざまなものがあります。

UUCP のハードウェア構成

UUCP は、次のハードウェア構成で利用できます。

直接リンク	2 つのマシンのシリアルポート間を RS-232 ケーブルで結ぶことにより、他のコンピュータとの間の直接リンクを作成できる。2 つのコンピュータが常時互いに通信を行い、両者の間の距離が 15m 以内の場合は、直接リンクを使用すると便利。この制限距離は、短距離モデムを使用することによりある程度延長できる

電話回線	高速モデムなどの自動呼び出し装置 (ACU) を使用すれば、通常の電話回線を介して他のコンピュータと通信できる。モデムは、UUCP が要求する電話番号をダイアルする。受信側のモデムは、着信に応答できなければならない
ネットワーク	UUCP は、TCP/IP またはその他のプロトコルファミリが機能するネットワークを介しても通信できる。コンピュータがネットワーク上でホストとして確立されていれば、そのネットワークに接続されている他のどのホストとも通信できる

この章では、UUCP ハードウェアをすでに設置、構成してあるものとして説明を進めます。モデムを設定する必要がある場合は、『*Solaris* のシステム管理 *(*基本編*)*』と、モデムに付属のマニュアルを参照してください。

UUCP ソフトウェア

Solaris インストールプログラムを実行するときに全体ディストリビューションを選択していれば、UUCP ソフトウェアは自動的に組み込まれています。あるいは、`pkgadd` を使用して UUCP を単独で追加することもできます。UUCP のプログラムは、デーモン、管理プログラム、およびユーザープログラムの 3 種類に分類されます。

UUCP デーモン

UUCP システムには、`uucico`、`uuxqt`、`uusched`、および `in.uucpd` の 4 つのデーモンがあります。 これらのデーモンは、UUCP のファイル転送とコマンド実行を処理します。これらのデーモンは、必要に応じて、シェルから手動で実行することもできます。

`uucico`	リンクに使用するデバイスを選択し、リモートコンピュータへのリンクを確立し、必要なログインシーケンスとアクセス権の検査を行う。また、データを転送し、ファイルを実行し、結果をログに記録し、転送の完了をメールによりユーザーに通知する。`uucico` は、UUCP ログインアカウント用の「ログインシェル」として働く。ローカル `uucico` デーモンはリモートマシンを呼び出して、セッションの間、リモート `uucico` デーモンと直接通信する 必要なファイルがすべて作成されたら、`uucp`、`uuto`、および `uux` プログラムが `uucico` デーモンを実行してリモートコンピュータに接続する。`uusched` と `Uutry` は、どちらも `uucico` を実行する。詳細は、`uucico(1M)` のマニュアルページを参照

uuxqt	リモート実行要求を処理する。このデーモンは、スプールディレクトリを検索して、リモートコンピュータから送られた実行ファイル (名前は常に X.*file*) を見つける。X.*file* が見つかると、uuxqt はそのファイルをオープンして、実行に必要なデータファイルのリストを取得する。次に、必要なデータファイルが使用可能でアクセスできるかどうかを確認する。ファイルが使用可能であれば、uuxqt は Permissions ファイルを調べて、要求されたコマンドを実行する権限があるかどうかを確認する。uuxqt デーモンは、cron により起動される uudemon.hour シェルスクリプトから実行される。詳細は、uuxqt(1M) のマニュアルページを参照
uusched	スプールディレクトリ内でキューに入っている作業をスケジュールする。uusched デーモンは、cron によって、ブート時に最初に実行される。詳細は、uusched(1M) のマニュアルページを参照。 uusched は uucico デーモンを起動する前に、リモートコンピュータを呼び出す順序をランダム化する
in.uucpd	ネットワークを介した UUCP 接続をサポートする。リモートホスト上の inetd は、UUCP 接続が確立されるたびに in.uucpd を呼び出す。次に、uucpd がログイン名を要求する。呼び出し側ホストの uucico は、これに対してログイン名を応答しなければならない。次に in.uucpd は、不要な場合を除いてパスワードを要求する。詳細は、in.uucpd(1M) のマニュアルページを参照

UUCP 管理プログラム

ほとんどの UUCP 管理プログラムは、/usr/lib/uucp に置かれています。基本データベースファイルの多くは、/etc/uucp に置かれています。ただし、uulog だけは例外で、これは /usr/bin に置かれています。uucp ログイン ID のホームディレクトリは /usr/lib/uucp です。su または login を使用して管理プログラムを実行するときには、uucp ユーザー ID を使用します。このユーザー ID は、プログラムとスプールデータファイルを所有しています。

uulog	指定したコンピュータのログファイルの内容を表示する。ログファイルは、このマシンが通信する各リモートコンピュータごとに作成される。ログファイルには、 uucp、uuto、uux の使用が記録される。詳細は、 uucp(1C) のマニュアルページを参照
uucleanup	スプールディレクトリをクリーンアップする。これは通常、cron によって起動される uudemon.cleanup シェルスクリプトから実行される。詳細は、uucleanup(1M) のマニュアルページを参照
Uutry	呼び出し処理機能をテストし、簡単なデバッグを行うことができる。uucico デーモンを呼び出して、このマシンと指定されたリモートコンピュータとの間の通信リンクを確立する。詳細は、Uutry(1M) のマニュアルページを参照

uucheck	UUCP のディレクトリ、プログラム、およびサポートファイルの有無を検査する。また、/etc/uucp/Permissions ファイルの所定の部分に、明らかな構文エラーがないかどうかも検査する。詳細は、uucheck(1M) のマニュアルページを参照

UUCP ユーザープログラム

UUCP のユーザープログラムは /usr/bin にあります。これらのプログラムを使用するのに、特別な権限は必要ありません。

cu	このマシンをリモートコンピュータに接続して、ユーザーが両方のマシンに同時にログインできるようにする。cu を使用すれば、接続したリンクを切断することなく、どちらのマシンでもファイルを転送したり、コマンドを実行したりできる。詳細は、cu(1C) のマニュアルページを参照
uucp	あるマシンから別のマシンへファイルをコピーする。uucp は作業ファイルとデータファイルを作成し、転送するジョブをキューに入れ、uucico デーモンを呼び出す。このデーモンは、リモートコンピュータへの接続を試みる。詳細は、uucp(1C) のマニュアルページを参照
uuto	ローカルマシンから、リモートマシン上の公共スプールディレクトリ /var/spool/uucppublic/receive にファイルをコピーする。uucp はリモートマシン上のアクセス可能な任意のディレクトリにファイルをコピーするのに対して、uuto は所定のスプールディレクトリにファイルを格納し、リモートユーザーに uupick を使用してそのファイルを取り出すよう指示する。詳細は、uuto(1C) のマニュアルページを参照
uupick	uuto を使用してコンピュータにファイルが転送されてきたときに、/var/spool/uucppublic/receive からファイルを取得する。詳細は、uuto(1C) のマニュアルページを参照
uux	リモートマシン上でコマンドを実行するために必要な作業ファイル、データファイル、および実行ファイルを作成する。詳細は、uux(1C) のマニュアルページを参照
uustat	要求された転送 (uucp、uuto、uux) の状態を表示する。また、キューに入っている転送を制御する手段も提供する。詳細は、uustat(1C) のマニュアルページを参照

UUCP データベースファイル

UUCP の構成の主要部分の 1 つに、UUCP データベースを形成するファイルの構成があります。これらのファイルは /etc/uucp ディレクトリにあります。マシン上で UUCP または asppp を設定するには、これらのファイルを編集する必要があります。使用できるファイルを次に示します。

Config	変数パラメータのリストが入っている。これらのパラメータは、ネットワークを構成するために手動で設定できる
Devconfig	ネットワーク通信を構成するために使用される
Devices	ネットワーク通信を構成するために使用される
Dialcodes	Systems ファイルのエントリの電話番号フィールド内で使用できるダイアルコード省略名が入っている。これは必須ではないが、UUCP の他に asppp でも使用できる
Dialers	リモートコンピュータとの接続を確立するときに、モデムとのネゴシエーションを行うために必要な文字列が入っている。これは、UUCP の他に asppp でも使用される
Grades	ジョブの処理順序と、ジョブの各処理順序に関連付けられたアクセス権を定義する。これらは、リモートコンピュータのキューにジョブを入れる際に、ユーザーが指定できる
Limits	このマシンで同時に実行できる uucico、uuxqt、および uusched の最大数を定義する
Permissions	このマシンにファイルを転送したり、コマンドを実行しようとしているリモートホストに与えられるアクセスのレベルを定義する
Poll	このシステムがポーリングするマシンと、ポーリングする時刻を定義する
Sysfiles	uucico と cu が、Systems、Devices、および Dialers ファイルとして、別のファイルや複数のファイルを使用する時に、その割り当てを行う
Sysname	TCP/IP ホスト名の他に、各マシンに固有の UUCP 名を定義できる
Systems	uucico デーモン、cu、および asppp が、リモートコンピュータへのリンクを確立するために必要とする情報が入っている。この情報には、リモートホスト名、リモートホストに対応する接続デバイス名、そのホストに接続できる日時、電話番号、ログイン ID、パスワードが含まれる

サポートデータベースの一部とみなすことのできるファイルが他にもいくつかありますが、それらは、リンクの確立とファイルの転送には直接関係しません。

UUCP データベースファイルの構成設定

UUCP データベースは、585 ページの「UUCP データベースファイル」に示したファイルから構成されます。ただし、基本的な UUCP 構成に関係する重要なファイルは次に示すものだけです。

- `/etc/uucp/Systems`
- `/etc/uucp/Devices`
- `/etc/uucp/Dialers`

`asppp` は UUCP データベースの一部を使用するので、`asppp` を構成する予定がある場合は、少なくともこれらのデータベースファイルだけは理解しておく必要があります。これらのデータベースを構成してしまえば、その後の UUCP の管理はきわめて簡単です。通常、`Systems` ファイルを最初に編集し、次に `Devices` ファイルを編集します。`/etc/uucp/Dialers` ファイルは、普通はデフォルトのままで使用できますが、デフォルトファイルに含まれていないダイアラを追加する予定がある場合は編集が必要になります。基本的な UUCP 構成と `asppp` 構成には、さらに次のファイルを加えることもできます。

- `/etc/uucp/Sysfiles`
- `/etc/uucp/Dialcodes`
- `/etc/uucp/Sysname`

これらのファイルは互いに関係しながら機能するので、1 つでも変更する場合は、全部のファイルの内容を理解しておくことが必要です。あるファイルのエントリに変更を加えた場合に、別のファイル内の関連エントリに対しても変更が必要になることがあります。585 ページの「UUCP データベースファイル」に挙げたその他のファイルは、上記のファイルほど緊密な相互関係を持っていません。

注 – `asppp` が使用するファイルはこの節で説明するものだけです。他の UUCP データベースファイルは使用しません。

第 39 章

UUCP の管理 (手順)

この章では、ご使用のマシンに合わせてデータベースファイルを変更したあと、UUCP 処理を起動する方法について説明します。この章には、Solaris 環境が動作するマシンで UUCP を構成し保守するための、手順と障害の解明についての情報が記載されています。

UUCP 管理 (作業マップ)

表 39–1に、この章で説明する手順の参照先と、各手順についての簡単な説明を示します。

表 **39–1** UUCP 管理 (作業マップ)

作業	説明	参照先
リモートマシンにユーザーシステムへのアクセスを許可する	`/etc/passwd` ファイルを編集し、ユーザーのシステムへのアクセスを許可するマシンを識別するようエントリを追加する	588 ページの「UUCP ログインの追加方法」
UUCP を起動する	UUCP の起動用に提供されているシェルスクリプトを使用する	589 ページの「UUCP の起動方法」

表 **39–1** UUCP 管理 (作業マップ)　(続き)

作業	説明	参照先
UUCP を TCP/IP ネットワーク上で有効にする	/etc/inetd.conf ファイルと /etc/uucp/Systems ファイルを編集し、TCP/IP 用の UUCP を起動する	591 ページの「TCP/IP 用 UUCP の起動方法」
UUCP に起こりがちな問題を解決する	モデムまたは ACU の異常を確認するための診断手順。 送信に関するデバッグを行うための診断手順	593 ページの「モデムまたは ACU の障害確認方法」 594 ページの「送信に関するデバッグ方法」

UUCP のログインの追加

リモートマシンからの UUCP (uucico) 着信要求が正しく取り扱われるように、各リモートマシンはローカルシステム上にログインを持っていなければなりません。

▼ UUCP ログインの追加方法

ユーザーのシステムへのアクセスをリモートマシンに許可するには、次の手順を行なって /etc/passwd ファイルにエントリを追加する必要があります。

1. /etc/passwd ファイルを編集し、システムにアクセスを許可するマシンを識別するためのエントリを追加します。

   通常、UUCP 接続でのシステムへのアクセスを許可するリモートマシンについて、次のようなエントリを /etc/passwd ファイルに入力します。

   ```
   Ugobi:*:5:5:gobi:/var/spool/uucppublic:/usr/lib/uucp/uucico
   ```

   リモートマシンのログイン名は慣例的に、そのマシン名の前に大文字の U を付けたものです。8 文字を超える名前は使用できないので、一部を短縮した名前や省略名を使用しなければならない場合もあります。

   例に示したエントリは、Ugobi からのログイン要求に /usr/lib/uucp/uucico が応答することを示しています。ホームディレクトリは /var/spool/uucppublic です。パスワードは /etc/shadow ファイルから取得されます。パスワードとログイン名は、リモートマシンの UUCP 管理者と協議して決める必要があります。リモート側の管理者は、ログイン名と暗号化されていないパスワードを含む正しいエントリを、リモートマシンの Systems ファイルに追加する必要があります。

2. 他のシステムの **UUCP** 管理者と、ローカルマシン名を調整します。

   同様に、ローカルマシン名とパスワードについて、UUCP を介して通信する相手方のすべてのマシンの UUCP 管理者と協議する必要があります。

UUCP の起動

UUCP には、次に示す 4 つのシェルスクリプトが付属しています。これらのスクリプトは、リモートマシンをポーリングし、転送を再スケジュールし、古いログファイルと成功しなかった転送を整理します。4 つのスクリプトは次のとおりです。

- uudemon.poll
- uudemon.hour
- uudemon.admin
- uudemon.cleanup

UUCP を円滑に運用するには、これらのスクリプトを定期的に実行する必要があります。Solaris の全体インストールを行なった場合は、これらのスクリプトを実行するための crontab ファイルが、インストールプロセスの一環として自動的に /usr/lib/uucp/uudemon.crontab の中に作成されます。全体インストールでない場合は、UUCP パッケージをインストールするときにこのファイルが作成されます。

UUCP シェルスクリプトは手動でも実行できます。次に示すのは、uudemon.crontab のプロトタイプです。 このファイルは、マシンの運用の都合に合わせて適宜変更できます。

```
#
#ident  "@(#)uudemon.crontab    1.5     97/12/09 SMI"
#
# This crontab is provided as a sample. For systems
# running UUCP edit the time schedule to suit, uncomment
# the following lines, and use crontab(1) to activate the
# new schedule.
#
#48 8,12,16 * * * /usr/lib/uucp/uudemon.admin
#20 3 * * * /usr/lib/uucp/uudemon.cleanup
#0 * * * * /usr/lib/uucp/uudemon.poll
#11,41 * * * * /usr/lib/uucp/uudemon.hour
```

注 – デフォルトでは、UUCP の操作は無効にされています。UUCP を有効にするには、タイムスケジュールを編集し、uudemon.crontab ファイルの適切な行のコメントを解除してください。

▼ UUCP の起動方法

uudemon.crontab ファイルは、次の手順に従って起動します。

1. スーパーユーザーになります。

2. /usr/lib/uucp/uudemon.crontab ファイルを編集し、必要に応じてエントリを変更します。

3. 次のコマンドを入力して、uudemon.crontab ファイルを起動します。

```
crontab < /usr/lib/uucp/uudemon.crontab
```

uudemon.poll シェルスクリプト

デフォルトの uudemon.poll シェルスクリプトは、1 時間に 1 回 /etc/uucp/Poll ファイルを読み取ります。Poll ファイル内のマシンのどれかに対するポーリングがスケジュールされると、作業ファイル (C.*sysnxxx*) が /var/spool/uucp/*nodename* ディレクトリに入れられます。*nodename* は、そのマシンの UUCP ノード名です。

このシェルスクリプトは、1 時間に 1 回ずつ uudemon.hour の前に実行されるようにスケジュールされているので、uudemon.hour が呼び出されたときには、作業ファイルが存在しています。

uudemon.hour シェルスクリプト

デフォルトの uudemon.hour シェルスクリプトは次のことを行います。

- uusched プログラムを呼び出し、スプールディレクトリを検索して未処理の作業ファイル (C.) を見つける。そして、それらの作業ファイルをリモートマシンに転送するためにスケジュールする
- uuxqt デーモンを呼び出し、スプールディレクトリを検索して、ローカルコンピュータに転送済みで、転送時に処理されなかった実行ファイル (X.) を見つける

デフォルトでは、uudemon.hour は 1 時間に 2 回実行されます。リモートマシンに対する呼び出しが頻繁に失敗すると予測される場合は、このスクリプトの実行頻度を増やすこともできます。

uudemon.admin シェルスクリプト

デフォルトの uudemon.admin シェルスクリプトは次のことを行います。

- p オプションと q オプション付きで uustat コマンドを実行する。q は、キューに入っている作業ファイル (C.)、データファイル (D.)、および実行ファイル (X.) の状態を報告する。p は、ロックファイル (/var/spool/locks) 中に列挙されているネットワークプロセス用のプロセス情報を表示する
- 結果の状態情報を、mail により uucp 管理ログインに送る

uudemon.cleanup シェルスクリプト

デフォルトの uudemon.cleanup シェルスクリプトは次のことを行います。

- /var/uucp/.Log ディレクトリから個々のマシンに関するログファイルを取り出し、それらをマージし、他の古いログ情報とともに /var/uucp/.Old ディレクトリに入れる
- 7 日以上経過している作業ファイル (C.)、7 日以上経過しているデータファイル (D.)、および2 日以上経過している実行ファイル (X.)を、スプールファイルから削除する
- 配送できなかったメールを送信元に戻す
- その日に収集した状態情報の要約を、メールにより UUCP 管理ログイン (uucp) に送る

TCP/IP を介した UUCP の実行

TCP/IP ネットワーク上で UUCP を実行するには、この節で説明するようにいくつかの変更が必要になります。

▼ TCP/IP 用 UUCP の起動方法

1. /etc/inetd.conf ファイルを編集し、次のエントリがコメント記号 **(#)** で始まっていないことを確認します。

   ```
   uucp stream tcp nowait root /usr/sbin/in.uucpd in.uucpd
   ```

2. /etc/uucp/Systems ファイルを編集し、対象エントリが次のフィールドを持っていることを確認します。

   System-Name Time TCP Port networkname Standard-Login-Chat

   典型的なエントリは次のようになります。

   ```
   rochester Any TCP - ur-seneca login: Umachine password: xxx
   ```

   networkname フィールドには、TCP/IP ホスト名を明示的に指定できます。これは、一部のサイトにとっては重要な点です。上の例に示したサイトの UUCP ノード名 rochester は、TCP/IP ホスト名 ur-seneca と違っています。さらに、rochester という TCP/IP ホスト名を持ち、UUCP を実行するまったく別のマシンが存在することもあり得ます。

   Systems ファイル内の Port フィールドには - を指定します。これは、エントリを uucp と指定するのと同じです。ほとんどの場合、*networkname* はシステム名と同じで、Port フィールドは - となります。これは、services データベースから標準 uucp ポートを使用することを意味します。in.uucpd デーモンは、認証のためにリモートマシンがログインとパスワードを送ることを想定しているので、getty や login と同様に、ログインとパスワードを要求します。

3. /etc/inet/services ファイルを編集し、次のように **UUCP** 用のポートを設定します。

```
uucp    540/tcp    uucpd        # uucp daemon
```

このエントリを変更する必要はありません。ただし、マシンがネームサービスとして NIS または NIS+ を実行する場合は、/etc/services の /etc/nsswitch.conf エントリを変更して、まず files、次に nis または nisplus が検査されるようにする必要があります。

UUCP のセキュリティと保守

UUCP の設定が終われば、その後の保守は簡単です。この節では、セキュリティ、保守、および障害追跡に関連する UUCP の作業について説明します。

UUCP のセキュリティの設定

デフォルトの /etc/uucp/Permissions ファイルは、UUCP リンクに関する最大限のセキュリティを提供します。デフォルトの Permissions ファイルには、エントリは入っていません。

定義する各リモートマシンについて、次に示す追加パラメータを設定できます。

- ローカルマシンからファイルを受け取る方法
- 読み取り権と書き込み権が与えられるディレクトリ
- リモート実行に使用できるコマンド

典型的な Permissions のエントリは次のようになります。

```
MACHINE=datsun LOGNAME=Udatsun VALIDATE=datsun
COMMANDS=rmail REQUEST=yes SENDFILES=yes
```

このエントリでは、システム内の任意の場所ではなく、通常の UUCP ディレクトリとの間でのファイルの送信と受信が可能となります。また、ログイン時にUUCP ユーザー名の認証が行われます。

日常の UUCP の保守

UUCP の保守に必要な作業の量はさほど多くはありません。589 ページの「UUCP の起動方法」で述べたように、crontab ファイルを正しい場所に配置してあることを確認する以外には、メールファイルと公共ディレクトリが大きくなるという点に注意する必要があります。

UUCP に関連する電子メール

UUCP のプログラムとスクリプトが生成する電子メールメッセージは、すべてユーザー ID uucp に送信されます。管理者がユーザー uucp として頻繁にログインしていないと、メールが蓄積されている (このためディスク空間を浪費している) ことに気付かない場合があります。この問題を解決するには、/etc/mail/aliases の中に別名を 1 つ作り、root か自分自身、そして他の UUCP 保守責任者に、電子メールを転送します。aliases ファイルを変更したあとで、newaliases コマンドを実行するのを忘れないようにしてください。

UUCP 公共ディレクトリ

ディレクトリ /var/spool/uucppublic は、UUCP がデフォルトでファイルをコピーできる場所として、すべてのシステムに対して提供されているディレクトリです。すべてのユーザーが、/var/spool/uucppublic への移動、その中のファイルの読み書きを行う権限を持っています。しかし、スティッキビットが設定されているため、このディレクトリのモードは 01777 です。したがって、ユーザーには、このディレクトリにコピーされ uucp に所有されているファイルを削除することはできません。このディレクトリからファイルを削除できるのは、root または uucp としてログインした UUCP 管理者だけです。このディレクトリ内に無秩序にファイルが蓄積するのを防ぐために、定期的にファイルを削除する必要があります。

このような保守作業がユーザーにとって面倒な場合は、セキュリティのために設定されているスティッキビットを削除するよりも、uuto と uupick を使用するよう各ユーザーに奨励してください。uuto と uupick の使い方については、uuto(1C) のマニュアルページを参照してください。このディレクトリのモードの制限の度合を強めて、特定のユーザーグループに使用を限定することもできます。だれかがディスク空間を使い切ってしまうことが望ましくないのであれば、そのディスクへの UUCP アクセスを拒否することもできます。

UUCP の障害追跡

ここでは、UUCP に関する一般的な問題を解決するための手順について説明します。

▼ モデムまたは ACU の障害確認方法

モデムや ACU で、適正に動作していないものがないかどうかを、いくつかの方法で検査できます。

1. 次のコマンドを実行し、接続障害の回数と理由を表示します。

```
# uustat -q
```

2. 特定の回線を介した呼び出しを行い、その試行に関するデバッグ情報を表示します。

   この回線は、/etc/uucp/Devices ファイル内で direct として定義されていなければなりません (回線が自動ダイアラに接続している場合は、コマンド行の終わりに電話番号を追加するか、デバイスを direct として設定する必要があります)。次のように入力します。

   ```
   # cu -d -l line
   ```

   line は /dev/cua/a です。

▼ 送信に関するデバッグ方法

特定のマシンに接続できない場合は、Uutry と uucp を使用して、そのマシンに対する通信を検査できます。

1. 接続を確認するには、次のように入力します。

   ```
   # /usr/lib/uucp/Uutry -r machine
   ```

   machine には、接続に問題のあるマシンのホスト名を指定します。このコマンドは次のことを行います。

   - デバッグ機能を指定して転送デーモン (uucico) を起動する。root としてログインしていれば、さらに多くのデバッグ情報が得られる
   - デバッグ出力を /tmp/*machine* に送る
   - 次のように入力すると、デバッグ出力を端末に表示する

     ```
     # tail -f
     ```

     出力を終了するには Control-c キーを押します。この出力を保存したい場合は、/tmp/*machine* から出力内容をコピーします。

2. Uutry を使用しても問題の原因が分からない場合は、ジョブをキューに入れてみます。

   ```
   # uucp -r file machine\!/dir/file
   ```

   file には転送したいファイル、*machine* には転送先のマシンを指定します。*/dir/file* には、相手のマシンのどこにファイルを転送するかを指定します。r オプションを指定すると、ジョブはキューに入りますが、転送は開始されません。

3. 次のコマンドを入力します。

   ```
   # Uutry
   ```

   それでも問題が解決できないときは、ご購入先へお問い合わせください。デバッグ出力を保存しておいてください。これは問題の診断に役立ちます。

   Uutry で -x *n* オプションを使用して、デバッグのレベルを増減することもできます。*n* はデバッグレベルを指定します。Uutry のデフォルトのデバッグレベルは 5 です。

デバッグレベル 3 では、接続がいつどのように確立されたかについての基本的な情報は提供されますが、転送自体について提供される情報は多くはありません。一方、デバッグレベル 9 では、転送処理に関するすべての情報が網羅されます。デバッグは転送の両端で行われるという点に注意してください。比較的大きいテキストについて 5 より高いレベルのデバッグを行いたい場合は、相手サイトの管理者に連絡して、デバッグを行う時期について同意を得てください。

UUCP /etc/uucp/Systems ファイルの検査

特定のマシンと接続しようとすると障害が発生する場合は、Systems ファイルの中の情報が最新のものであることを確認してください。マシンに関する次の情報が、最新でない可能性があります。

- 電話番号
- ログイン ID
- パスワード

UUCP エラーメッセージの検査

UUCP のエラーメッセージには、ASSERT と STATUS の 2 つの種類があります。

- プロセスが異常終了した場合は、ASSERT エラーメッセージが /var/uucp/.Admin/errors に記録されます。この種類のメッセージには、ファイル名、sccsid、回線番号、およびテキストが含まれています。この種類のメッセージが送られるのは、通常、システムに問題がある場合です。
- STATUS エラーメッセージは /var/uucp/.Status ディレクトリに格納されます。このディレクトリ内には、ローカルコンピュータが通信しようとした各リモートマシンについて、それぞれファイルが作られます。これらのファイルには、試行した通信と、その通信が成功したかどうかについての状態情報が入っています。

基本情報の検査

次のコマンドを使用して、基本的なネットワーク情報を検査できます。

- uuname コマンドは、ローカルマシンが接続できるマシンのリストを表示したい場合に使用します。
- uulog コマンドは、特定のホストのためのログディレクトリの内容を表示するために使用します。
- uucheck -v コマンドは、uucp が必要とするファイルとディレクトリが存在しているかどうかを検査するために使用します。また、Permissions ファイルも検査して、設定してあるアクセス権に関する情報を出力します。

第 40 章

UUCP (リファレンス)

この章では、UUCP を使用する場合のリファレンス情報について説明します。この章の内容は次のとおりです。

UUCP /etc/uucp/Systems ファイル

/etc/uucp/Systems ファイルには、uucico がリモートコンピュータとの通信リンクを確立するために必要な情報が入っています。/etc/uucp/Systems は、UUCP を構成するときに編集しなければならない最初のファイルです。

Systems ファイルの中の各エントリは、このホストが通信するリモートコンピュータを表します。1 つのホストについて複数のエントリがある場合もあります。付加的なエントリは、順番に試される代替通信パスを表します。さらに、UUCP のデフォルト状態では、/etc/uucp/Systems ファイルに含まれていないコンピュータがこのホストにログインできないようになっています。

Sysfiles ファイルを使用して、Systems ファイルとして使用されるファイルをいくつか定義できます。詳細は、615 ページの「UUCP /etc/uucp/Sysfiles ファイル」で Sysfiles ファイルの説明を参照してください。

Systems ファイルのエントリの形式は次のとおりです。

System-Name	*Time*	*Type*	*Speed*	*Phone*	*Chat Script*

例 40–1 に、Systems ファイルのフィールドの例を示します。

例 **40–1** /etc/uucp/Systems のフィールド

```
System-Name Time Type  Speed Phone   Chat Script
Arabian     Any  ACUEC 38400 111222  Login: Puucp ssword:beledi
```

UUCP System-Name フィールド

このフィールドには、リモートコンピュータのノード名が入ります。TCP/IP ネットワークでは、この名前は、マシンのホスト名でも、/etc/uucp/Sysname ファイルによって UUCP 通信用として特別に作成した名前でもかまいません。597 ページの「UUCP /etc/uucp/Systems ファイル」を参照してください。例 40–1 では、System-Name フィールドにはリモートホスト arabian に関するエントリが含まれています。

UUCP Time フィールド

このフィールドには、リモートコンピュータを呼び出すことのできる曜日と時刻を指定します。Time フィールドの形式は次のとおりです。

```
daytime[;retry]
```

day の部分には、次のエントリのいくつかを含むリストを指定できます。

表 **40–1** Day フィールド

Su Mo Tu We Th Fr Sa	個々の曜日
Wk	任意の平日
Any	任意の日
Never	このホストはこのリモートコンピュータの呼び出しをいっさい行わない。呼び出しはリモートコンピュータ側から行う必要がある。それを受けて、このホストは受動モードで稼動する

例 40–1 では、Time フィールドに Any が示されています。これは、ホスト arabian をいつでも呼び出せるということです。

time の部分には、24 時間表記で表した時間の範囲を指定します。たとえば、午前 8 時 00 分から午後 12 時 30 分までなら `0800-1230` とします。*time* の部分を指定しなかった場合は、どのような時刻にでも呼び出しができるものとみなされます。

0000 の前後にまたがる時間範囲も指定できます。たとえば、`0800-0600` は、午前 6 時から午前 8 時までの間を除くすべての時間帯で呼び出し可能であることを示します。

UUCP retry サブフィールド

`retry` サブフィールドには、試行が失敗してから次の再試行までの間に最小限必要な時間 (分単位) を指定できます。デフォルトの待ち時間は 60 分です。サブフィールド区切り文字はセミコロン (;) です。たとえば、`Any;9` は、呼び出しはいつでもできるが、失敗したときは次の再試行までに少なくとも 9 分は待たなければならないことを意味します。

retry エントリを指定しなかった場合は、待ち時間倍加アルゴリズムが使用されます。これは、UUCP がデフォルトの待ち時間から始めて、失敗した試行の回数が増えるほど待ち時間を長くしていくことを意味します。たとえば、最初の再試行待ち時間が 5 分であるとします。応答がない場合は、次の再試行は10 分後となりま す。次の再試行は 20 分後というようになり、最大再試行時間の 23 時間に達するまで増加します。*retry* を指定した場合は、常にその値が再試行待ち時間となります。指定がなければ待ち時間倍加アルゴリズムが使用されます。

UUCP Type フィールド

このフィールドには、リモートコンピュータとの通信リンクを確立するために使用するデバイスタイプを指定します。このフィールドで使用するキーワードは、`Devices` ファイル中のエントリの最初のフィールドと突き合わされます。

例 **40–2** Type フィールドと `/etc/uucp/Devices` ファイル

```
File Name System-Name  Time  Type    Speed  Phone    Chat Script

Systems   arabian      Any   ACUEC, g 38400  1112222   ogin: Puucp ssword:beledi
```

Type フィールドでは、さらに、システムとの接続に使用するプロトコルを定義できます。上記の例では、デバイスタイプ `ACUEC` に `g` プロトコルを組み合わせる方法を示しています。プロトコルの詳細は、609 ページの「UUCP `Devices` ファイル内のプロトコル定義」を参照してください。

UUCP Speed フィールド

このフィールド (Class フィールドとも呼ばれます) は、通信リンクの確立に使用するデバイスの転送速度を指定します。このフィールドには、ダイアラのクラスを区別するために、1 個の英字と速度を含めることができます (たとえば、C1200、D1200) (詳細は、606 ページの「UUCP Class フィールド」を参照してください)。

デバイスにはどのような速度でも使用できるものがあり、その場合はキーワード Any を使用できます。このフィールドは、Devices ファイルの対応するエントリの Class フィールドに一致していなければなりません。

例 **40–3** Speed フィールドと /etc/uucp/Devices ファイル

```
File Name System-Name Time  Type    Speed  Phone   Chat Script

Systems   eagle       Any   ACU, g  D1200  NY3251  ogin: nuucp ssword: Oakgrass
```

このフィールドに情報を入れる必要がない場合は、フィールドの数を合わせるためにダッシュ (-) を指定してください。

UUCP Phone フィールド

このフィールドには、自動ダイアラ (ポートセレクタ) に与えるリモートコンピュータの電話番号 (トークン) を指定できます。電話番号は、オプションの英字による省略名と数字部分で構成されます。省略名を使用する場合は、Dialcodes ファイル内に列挙されているものの 1 つでなければなりません。

例 **40–4** Phone フィールドの対応関係

```
File Name System-Name  Time  Type  Speed  Phone    Chat Script

Systems   nubian       Any   ACU   2400   NY5551212 ogin: Puucp ssword:Passuan
```

文字列 System-Name では、等号 (=) は、二次発信音を待ってから残りの数字をダイアルするという ACU への指示となります。文字列の中にダッシュ (-) があれば、4 秒間待ってから次の数字をダイアルするという指示になります。

コンピュータがポートセレクタに接続されている場合は、そのセレクタに接続している他のコンピュータにアクセスできます。この種のリモートマシン用の Systems ファイルエントリの Phone フィールドには、電話番号を入れません。代わりに、このフィールドにはスイッチに渡すトークンを指定します。このようにすれば、このホストがどのリモートマシンとの通信を望んでいるかを、ポートセレクタが判断できます。この場合は、システム名だけを指定するのが普通です。対応する Devices ファイルエントリでは、エントリの末尾に \D を指定して、このフィールドが Dialcode ファイルを使用して解釈されないようにしなければなりません。

UUCP Chat-Script フィールド

このフィールド (Login フィールドとも呼ばれます) には、チャットスクリプトと呼ばれる文字列が入ります。チャットスクリプトには、ローカルマシンとリモートマシンが対話の最初の時点で互いに受け渡ししなければならない文字が含まれています。チャットスクリプトの形式は次のとおりです。

expect send [expect send] ....

expect は、対話を開始するために、ローカルホストがリモートホストから受信することを想定している文字列です。*send* は、ローカルホストが、リモートホストからの *expect* 文字列を受信した後で送信する文字列です。チャットスクリプトには、複数の expect-send シーケンスを含めることもできます。

基本的なチャットスクリプトには次の情報が含まれます。

- ローカルホストがリモートマシンから受信することを想定しているログインプロンプト
- ログインするためにローカルホストがリモートマシンに送るログイン名
- ローカルホストがリモートマシンから受信することを想定しているパスワードプロンプト
- ローカルホストがリモートマシンに送るパスワード

expect フィールドは、次の形式のサブフィールドを持つことができます。

expect[-send-expect]...

-send は、その前の *expect* が正常に読み取れなかった場合に送られるものであり、*send* の後の *-expect* は、その次に送られてくると想定されている文字列です。

たとえば、`login--login` という文字列を指定した場合、ローカルホストの UUCP は `login` が送られてくることを想定します。リモートマシンから `login` を受信すると、UUCP は次のフィールドに進みます。`login` を受信しなかった場合は、キャリッジリターンを送信し、再度 `login` が送られてくるのを待ちます。ローカルコンピュータが、初期状態でどのような文字も想定していない場合は、*expect* フィールドで文字列 `""` (NULL 文字列) を指定します。*send* 文字列が `\c` で終わっている場合を除き、*send* フィールドの送信の後には必ずキャリッジリターンが伴うという点に注意してください。

次に示すのは、*expect-send* 文字列を使用する `Systems` ファイルエントリの例です。

```
System-Name Time Type Speed Phone   Chat Script
sonora Any ACUEC 9600 2223333 "" \r \r ogin:-BREAK-ogin: Puucpx ssword: xyzzy
```

この例は、ローカルホストの UUCP に、2 個のキャリッジリターンを送ってから `ogin:` (`Login:` を表す) を待つように指示しています。 `ogin:` を受信しなかった場合は、`BREAK` を送ります。`ogin:` を受信した場合は、ログイン名 `Puucpx` を送ります。`ssword:` (`Password:` を表す) を受け取ったら、パスワード `xyzzy` を送ります。

表 40–2 に、便利なエスケープ文字をいくつか紹介します。

表 **40–2** `Systems` ファイルのチャットスクリプトで使用されるエスケープ文字

エスケープ文字	意味
`\b`	バックスペース文字を送信または想定する
`\c`	文字列の末尾で使用すると、普通なら送信されるキャリッジリターンが抑止される。その他の場合は無視される
`\d`	後続の文字を送る前に 1 ～ 3 秒の遅延が生じる
`\E`	エコーチェックを開始する。これ以降は、1 文字送信するたびに、UUCP はその文字が受信されるまで待ち、その後、チェックを続行する
`\e`	エコーチェックをオフにする
`\H`	ハングアップを 1 回無視する。このオプションはコールバックモデム用に使用する
`\K`	BREAK 文字を送信する
`\M`	`CLOCAL` フラグをオンにする
`\m`	`CLOCAL` フラグをオフにする
`\n`	改行文字を送信または想定する
`\N`	NULL 文字 (ASCII NUL) を送信する
`\p`	約 1/4 秒間または 1/2 秒間、一時停止する
`\r`	キャリッジリターンを送信または想定する
`\s`	スペース文字を送信または想定する
`\t`	タブ文字を送信または想定する
`EOT`	EOT とそれに続く 2 個の改行文字を送信する
`BREAK`	ブレーク文字を送信する
`\`*ddd*	8 進数 (*ddd*) で表される文字を送信または想定する

チャットスクリプトを使用したダイアルバックの有効化

組織によっては、リモートコンピュータからの呼び出しを処理するダイアルインサーバーを設定する場合があります。たとえば、コールバックモデムを持つダイアルインサーバーを配備し、社員が自宅のコンピュータから呼び出せるようにすることができます。ダイアルインサーバーは、リモートマシンを識別すると、そのリモートマシンとのリンクを切断し、逆にそのリモートマシンを呼び出して、通信リンクが再確立されます。

Systems ファイルのチャットスクリプトで、コールバックが必要な箇所で \H オプションを使用することにより、コールバックの操作を簡素化することができます。ダイアルインサーバーのハングアップが予想される箇所で、expect 文字列の一部として \H を使用します。

たとえば、ダイアルインサーバーを呼び出すチャットスクリプトに、次のような文字列が含まれているとします。

```
INITIATED\Hogin:
```

ローカルホストの UUCP ダイアル機能は、ダイアルインサーバーから INITIATED という文字列を受け取ることを想定しています。INITIATED 文字列を受け取ると、ダイアル機能は、ダイアルインサーバーがハングアップするまで、その後受信するすべての文字をフラッシュします。またダイアル機能は、expect 文字列のその次の部分、つまり ogin: という文字列がダイアルインサーバーから送られてくるのを待ちます。ogin: を受け取ると、ダイアル機能はチャットスクリプトを先へ進めます。

上記のサンプルでは \H の前後に文字列が指定されていますが、これらはなくてもかまいません。

UUCP ハードウェアフロー制御

擬似送信文字列 STTY=value を用いることによっても、モデムの特性を設定できます。たとえば、STTY=crtscts を使用すると、ハードウェアフロー制御が可能になります。 STTY はすべての stty モードを受け入れます。詳細は、stty(1) と termio(7I) のマニュアルページを参照してください。

次の例は、Systems ファイルのエントリ内でハードウェアフロー制御を指定しています。

```
System-Name Time Type Speed Phone    Chat Script
unix Any ACU 2400 12015551212 "" \r login:-\r-login:-\r-login:
nuucp password: xxx "" \ STTY=crtscts
```

擬似送信文字列は、Dialers ファイルのエントリの中でも使用できます。

UUCP パリティの設定

場合によっては、呼び出そうとしているシステムがポートのパリティを検査し、パリティに誤りがあると回線を切断することがあります。そのため、パリティのリセットが必要になります。expect-send (予期-送信) の文字列ペアとして "" P_ZERO を使用すると、上位ビット (パリティビット) が 0 に設定されます。たとえば :

```
System-Name Time Type Speed Phone    Chat Script
unix Any ACU 2400 12015551212 "" P_ZERO "" \r login:-\r-login:-\r-login:
nuucp password: xxx
```

同様に、P_EVEN はパリティを偶数 (デフォルト) に設定し、P_ODD は奇数パリティを設定し、P_ONE はパリティビットを 1 に設定します。

パリティ設定は、チャットスクリプトのどこにでも挿入できます。この設定は、チャットスクリプト内の "" P_ZERO より後にあるすべての情報に適用されます。この設定は、Dialers ファイルのエントリの中でも使用できます。

UUCP /etc/uucp/Devices ファイル

/etc/uucp/Devices ファイルには、リモートコンピュータへのリンクを確立するために使用できるすべてのデバイスに関する情報が入っています。この種のデバイスには、ACU (が含まれます)、直接リンク、ネットワーク接続などがあります。

次に示す /etc/uucp/Devices のエントリは、ポート A に接続され 38,400 bps で動作する US Robotics V.32bis モデムを表しています。

```
Type   Line  Line2 Class Dialer-Token-Pairs
ACUEC  cua/a -     38400 usrv32bis-ec
```

各フィールドについて、次のセクションで説明します。

UUCP Type フィールド

このフィールドでデバイスが設定するリンクの Type を説明します。このフィールドには以下のセクションに示すキーワードのいずれかを入れることができます。

キーワード Direct

キーワード Direct は、主として cu 接続用のエントリ内で使用されます。このキーワードは、このリンクが他のコンピュータまたはポートセレクタへの直接リンクであることを示します。cu の -l オプションで参照したい各回線について、それぞれ独立したエントリを作成する必要があります。

キーワード ACU

キーワード ACU は、(cu、UUCP、asppp、または Solaris PPP 4.0 を介した) リモートコンピュータへのリンクを、モデムを介して確立することを示します。このモデムは、直接ローカルコンピュータに接続しているものでも、ポートセレクタを介して間接的に接続しているものでもかまいません。

ポートセレクタ

これは、ポートセレクタの名前で置き換えるものとして、Type フィールド内で使用される変数です。ポートセレクタは、ネットワークに接続されたデバイスで、呼び出し側モデムの名前を要求し、アクセスを許可します。/etc/uucp/Dialers ファイルに

入っている呼び出しスクリプトは、micom ポートセレクタと develcon ポートセレクタについてのものだけです。ユーザーは、Dialers ファイルに独自のポートセレクタエントリを追加できます。詳細は、610 ページの「UCCP /etc/uucp/Dialers ファイル」を参照してください。

Sys-Name

Type フィールド内のこの変数は、特定のマシンの名前で置き換えられます。これは、リンクがこのマシンへの直接リンクであることを示します。この命名スキーマは、この Devices エントリ内の行と、コンピュータ Sys-Name についての /etc/uucp/Systems ファイルエントリを対応付けるために使用されます。

Type フィールドと /etc/uucp/Systems ファイル

例 40–5 は、/etc/uucp/Devices のフィールドと、/etc/uucp/Systems のフィールドの対応を示しています。各列の見出しは Devices ファイルに対応するものです。

フィールドの書体を変えて示したように、Devices ファイルの Type フィールドで使用されているキーワードは、Systems ファイルエントリの 3 番目のフィールドと突き合わされます。Devices ファイルの Type フィールドには ACUEC というエントリが入っており、これは自動呼び出し装置、つまりこの例では V.32bis モデムを示しています。この値は、Systems ファイルの 3 番目のフィールドと突き合わされます。このフィールドにも ACUEC というエントリが入っています。詳細は、597 ページの「UUCP /etc/uucp/Systems ファイル」を参照してください。

例 **40–5** Type フィールドと /etc/uucp/Systems ファイルの対応関係

```
File Name Type   Line  Line2 Class Dialer-Token-Pairs

Devices   ACUEC  cua/a -     38400 usrv32bis-ec

System    nubian Any   ACUEC 38400 9998888 "" \d\d\r\n\c-ogin-\r\n\c-ogin.......
```

UUCP Line フィールド

このフィールドには、Devices エントリに対応付けられる回線 (ポート) のデバイス名が入ります。たとえば、特定のエントリに対応付けられているモデムが /dev/cua/a (シリアルポート A) に接続されている場合、このフィールドに入力する名前は cua/a です。Line フィールドでオプションのモデム制御フラグ M を使用すると、キャリアを待たないでデバイスをオープンすることを指定できます。たとえば:

```
cua/a,M
```

UUCP Line2 フィールド

このフィールドは、フィールドの数を合わせるために存在しているだけです。ここには常にダッシュ (-) を指定します。Line2 フィールドを使用するのは 801 型のダイアラですが、この種類は Solaris 環境ではサポートされていません。801 型以外のダイアラは通常はこの設定を使用しませんが、このフィールドにダッシュだけは入れておく必要があります。

UUCP Class フィールド

Type フィールドでキーワード `ACU` または `Direct` を使用した場合は、Class フィールドにはデバイスの速度が入ります。ただし、このフィールドには、ダイアラのクラス (Centrex または Dimension PBX) を区別するために、1 個の英字と速度値を含めることができます (たとえば、`C1200`、`D1200`)。

大規模な事業所では複数種の電話ネットワークを使用することが多いため、このような指定が必要になります。たとえば、1 つのネットワークは事業所内の内線通信専用に使用し、もう 1 つのネットワークは外線通信に使用するといった方式が考えられます。このような場合は、内線回線と外線回線とを区別する必要があります。

`Devices` ファイルの Class フィールドで使用するキーワードは、`Systems` ファイルの Speed フィールドと突き合わされます。

例 **40–6** UUCP Class フィールド

```
File Name  Type   Line  Line2  Class  Dialer-Token-Pairs

Devices    ACU    cua/a -      D2400  hayes
```

どのような速度でも使用できるデバイスでは、Class フィールドにキーワード `Any` を使用します。`Any` を使用した場合は、回線は、`Systems` ファイルの Speed フィールドで要求された任意の速度に適合します。このフィールドが `Any` で、`Systems` ファイルの Speed フィールドも `Any` である場合は、速度はデフォルトの 2400bps となります。

UUCP Dialer-Token-Pairs フィールド

Dialer-Token-Pairs (DTP) フィールドには、ダイアラの名前とそれに渡すトークンが入ります。DTP フィールドの構文は次のとおりです。

dialer token [dialer token]

dialer の部分は、モデムかポートモニターの名前あるいは直接リンクデバイスの場合は `direct` または `uudirect` です。ダイアラとトークンのペアはいくつでも指定できます。*dialer* の部分がない場合は、`Systems` ファイル内の関連エントリから取得されます。*token* 部は、dialer 部の直後に指定できます。

対応するダイアラによっては、最後のダイアラとトークンのペアはない場合もあります。ほとんどの場合は、最後のペアには *dialer* 部だけが含まれます。*token* 部は、対応する Systems ファイルエントリの Phone フィールドから取得されます。

dialer 部の有効エントリは、Dialers ファイル内で定義されているものか、いくつかの特殊ダイアラタイプのうちの 1 つとなります。これらの特殊ダイアラタイプはコンパイル時にソフトウェア中に組み込まれているので、Dialers ファイル内に該当エントリがなくても使用できます。表 40–3 に、特殊ダイアラタイプを示します。

表 **40–3** ダイアラとトークンのペア

TCP	TCP/IP ネットワーク
TLI	トランスポートレベルインタフェースネットワーク (STREAMS を使用しないもの)
TLIS	トランスポートレベルインタフェースネットワーク (STREAMS を使用するもの)

詳細は、609 ページの「UUCP Devices ファイル内のプロトコル定義」を参照してください。

Dialer-Token-Pairs フィールドの構造

DTP フィールドの構造は、エントリに対応するデバイスに応じて 4 通りに設定できます。

- 直接接続モデム

  コンピュータのポートにモデムが直接接続されている場合は、対応する Devices ファイルエントリの DTP フィールドに入るペアは 1 つだけです。このペアは、通常はモデムの名前です。この名前は、Devices ファイルの特定のエントリと、Dialers ファイル内のエントリとを対応付けるために使用されます。したがって、Dialer フィールドは、Dialers ファイルエントリの最初のフィールドに一致している必要があります。

例 **40–7** 直接接続モデム用 Dialers フィールド

```
Dialers   hayes =,-,  ""          \\dA\pTE1V1X1Q0S2=255S12=255\r\c
                                  \EATDT\T\r\c CONNECT
```

Devices ファイルエントリの DTP フィールドには、dialer 部 (hayes) だけが示されている点に注意してください。これは、ダイアラに渡す *token* (この例では電話番号) が、Systems ファイルエントリの Phone フィールドから取得されることを意味します (例 40–9 で説明するように、\T が暗黙で指定されます)。

- 直接リンク – 特定のコンピュータへの直接リンクの場合は、対応するエントリの DTP フィールドには、キーワード direct が入ります。これは、Direct、*Sys-Name* の両方の直接リンクエントリにもあてはまります (604 ページの「UUCP Type フィールド」を参照)。

- 同じポートセレクタ上のコンピュータ – 通信したいコンピュータが、ローカルコンピュータと同じポートセレクタスイッチ上にある場合は、ローカルコンピュータはまずそのスイッチにアクセスする必要があります。そのスイッチが、相手のコンピュータとの接続を確立します。この種のエントリでは、ペアは 1 つだけです。*dialer* 部が `Dialers` ファイルのエントリと突き合わされます。

例 **40–8** 同一ポートセレクタ上のコンピュータ用 UUCP Dialer フィールド

```
Dialers    develcon ,""   ""            \pr\ps\c est:\007 \E\D\e \007
```

token 部が空である点に注意してください。このように指定されている場合は、この部分が `Systems` ファイルから取得されることを示しています。このコンピュータ用の `Systems` ファイルエントリには、Phone フィールドにトークンが含まれています。このフィールドは、通常、コンピュータの電話番号用として確保されています。
597 ページの「UUCP `/etc/uucp/Systems` ファイル」を参照してください。この種類の DTP にはエスケープ文字 (`\D`) が含まれています。これは、Phone フィールドの内容が、`Dialcodes` ファイル内の有効エントリとして解釈されないことを保証します。

- ポートセレクタに接続しているモデム – ポートセレクタに高速モデムが接続されている場合は、ローカルコンピュータはまずポートセレクタスイッチにアクセスする必要があります。そして、そのスイッチがモデムとの接続を確立します。この種類のエントリには、ダイアラとトークンのペアが 2 つ必要です。各ペアの *dialer* 部 (エントリの 5 番目と 7 番目のフィールド) が、`Dialers` ファイル内のエントリと突き合わされます。

例 **40–9** ポートセレクタに接続されたモデム用 UUCP Dialer フィールド

```
Dialers    develcon ""    ""     \pr\ps\c  est:\007    \E\D\e      \007

Dialers    ventel   =&-%   t""   \r\p\r\c  $           <K\T%\r>\c  ONLINE!
```

最初のペアでは、`develcon` がダイアラで、`vent` が Develcon スイッチに渡されるトークンです。トークンは、コンピュータに接続するデバイス (たとえば Ventel モデム) をダイアラに指示しています。各スイッチごとに設定が異なることがあるので、このトークンは各ポートセレクタに固有のものにします。Ventel モデムが接続された後、第 2 のペアがアクセスされます。このペアでは、Ventel がダイアラで、トークンは `Systems` ファイルから取得されます。

DTP フィールドでは 2 つのエスケープ文字が使用できます。

- `\T` – Phone (*token*) フィールドを、`/etc/uucp/Dialcodes` ファイルを使用して解釈することを指定します。通常、モデム (Hayes、US Robotics など) に対応する各呼び出しスクリプトについて、`/etc/uucp/Dialers` ファイルにこのエスケープ文字を組み込みます。したがって、呼び出しスクリプトがアクセスされるまでは、解釈は行われません。
- `\D` – Phone (*token*) フィールドを、`/etc/uucp/Dialcodes` ファイルを使用して解釈しないことを指定します。`Devices` エントリの末尾にエスケープ文字が何も指定されていないときは、デフォルトで `\D` があるものと想定します。`\D` は、`/etc/uucp/Dialers` ファイルの中でも、ネットワークスイッチ (`develcon` と

micom) に関連したエントリで使用されます。

UUCP Devices ファイル内のプロトコル定義

/etc/uucp/Devices では、各デバイスに使用するプロトコルを定義できます。通常は、デフォルトを使用するか、または呼び出そうとしている特定のシステムに対してプロトコルを定義できるので、この指定は不要です。597 ページの「UUCP /etc/uucp/Systems ファイル」を参照してください。プロトコルを指定する場合は、次の形式を使用する必要があります。

Type,Protocol [parameters]

たとえば、TCP/IP プロトコルを指定するには、TCP,te を入力します。

表 40–4 に、Devices ファイルで使用できるプロトコルを示します。

表 40–4 /etc/uucp/Devices で使用されるプロトコル

プロトコル	説明
t	このプロトコルは、TCP/IP や、その他の信頼性のある接続を介した伝送に、最もよく使用される。t はエラーのない伝送を前提としている
g	UUCP のネイティブプロトコル。低速で信頼性があり、ノイズの多い電話回線を介した伝送に適している
e	このプロトコルは、(TCP/IP のようなバイトストリーム指向ではなく) メッセージ指向でエラーのないチャネルを介した伝送を前提としている
f	このプロトコルは X.25 接続を介した伝送に使用される。f は、データストリームのフロー制御に関係している。特に X.25/PAD リンクなどのように、完全に (またはほとんど) エラーがないことが保証されるリンクでの使用を意図している。検査合計はファイル全体についてのみ実施される。伝送が失敗した場合は、受信側は再伝送を要求する

次に、デバイスエントリ用のプロトコル指定の例を示します。

```
TCP,te - - Any TCP -
```

この例は、デバイス TCP について t プロトコルの使用を試みるように指示しています。相手側がそれを拒否した場合は、e プロトコルが使用されます。

e と t のどちらも、モデムを介した通信には適していません。モデムがエラーのない伝送を保証するものであったとしても、モデムと CPU との間でデータが失われる可能性があります。

UCCP /etc/uucp/Dialers ファイル

/etc/uucp/Dialers ファイルには、よく使用される多くのモデムに関するダイアリング指示が入っています。標準外のモデムの使用や、UUCP 環境のカスタマイズを予定している場合以外は、通常このファイルのエントリの変更や追加は必要ありません。しかし、このファイルの内容と、Systems ファイルや Devices ファイルとの関係は理解しておく必要があります。

このファイルの中のテキストは、回線をデータ転送に使用できるようにするために、最初に行わなければならない対話を指定します。チャットスクリプトと呼ばれるこの対話は、通常は送受信される一連の ASCII 文字列で、電話番号をダイアルするためによく使用されます。

604 ページの「UUCP /etc/uucp/Devices ファイル」の例に示したように、Devices ファイルの 5 番目のフィールドは、Dialers ファイルへのインデックスか、または特殊ダイアラタイプ (TCP、TLI、または TLIS) です。 uucico デーモンは、Devices ファイルの 5 番目のフィールドを、Dialers ファイルの各エントリの最初のフィールドと突き合わせます。さらに、Devices の 7 番目の位置から始まる奇数番号の各フィールドは、Dialers ファイルへのインデックスとして使用されます。これらが一致すると、その Dialers のエントリがダイアラ対話を行うために解釈されます。

Dialers ファイルの各エントリの形式は次のとおりです。

dialer	*substitutions*	*expect-send*

例 40–10 に、US Robotics V.32bis モデム用のエントリの例を示します。

例 **40–10** /etc/uucp/Dialers ファイルのエントリ

```
Dialer     Substitution  Expect-Send
usrv32bis-e   =,-,   ""       dA\pT&FE1V1X1Q0S2=255S12=255&A1&H1&M5&B2&W\r\c OK\r
                              \EATDT\T\r\c CONNECT\s14400/ARQ STTY=crtscts
```

Dialer フィールドは、Devices ファイルの中の 5 番目以降の奇数番号のフィールドと突き合わされます。Substitutions フィールドは変換文字列です。各文字ペアの最初の文字が 2 番目の文字に変換されます。通常この変換は = と - を、「発信音待ち」と「一時停止」用としてダイアラが必要とする文字に変換するために使用されます。

それ以降の expect-send の各フィールドは文字列です。

例 40–11 に、Dialers ファイルのエントリの例をいくつか示します。これは、Solaris インストールプログラムの一環として UUCP をインストールしたときに提供されるファイルです。

例 **40–11** /etc/uucp/Dialers の抜粋

```
penril    =W-P "" \d> Q\c : \d-> s\p9\c )-W\p\r\ds\p9\c-) y\c : \E\TP> 9\c OK

ventel    =&-%    "" \r\p\r\c $ <K\T%%\r>\c ONLINE!

vadic    =K-K    "" \005\p *-\005\p-*\005\p-* D\p BER? \E\T\e \r\c LINE

develcon    ""    "" \pr\ps\c est:\007

\E\D\e \n\007 micom    ""    "" \s\c NAME? \D\r\c GO

hayes    =,-,    "" \dA\pTE1V1X1Q0S2=255S12=255\r\c OK\r \EATDT\T\r\c CONNECT

#   Telebit TrailBlazer
tb1200    =W-,    "" \dA\pA\pA\pTE1V1X1Q0S2=255S12=255S50=2\r\c OK\r
\EATDT\T\r\c CONNECT\s1200
tb2400    =W-,    "" \dA\pA\pA\pTE1V1X1Q0S2=255S12=255S50=3\r\c OK\r
\EATDT\T\r\c CONNECT\s2400
tbfast    =W-,    "" \dA\pA\pA\pTE1V1X1Q0S2=255S12=255S50=255\r\c OK\r
\EATDT\T\r\c CONNECT\sFAST

# USrobotics, Codes, and DSI modems

dsi-ec  =,-,    "" \dA\pTE1V1X5Q0S2=255S12=255*E1*F3*M1*S1\r\c OK\r \EATDT\T\r\c
CONNECT\sEC STTY=crtscts,crtsxoff

dsi-nec =,-,    "" \dA\pTE1V1X5Q0S2=255S12=255*E0*F3*M1*S1\r\c OK\r \EATDT\T\r\c CONNECT
STTY=crtscts,crtsxoff

usrv32bis-ec =,-,  "" \dA\pT&FE1V1X1Q0S2=255S12=255&A1&H1&M5&B2&W\r\c OK\r \EATDT\T\r\c
CONNECT\s14400/ARQ STTY=crtscts,crtsxoff

usrv32-nec =,-, "" \dA\pT&FE1V1X1Q0S2=255S12=255&A0&H1&M0&B0&W\r\c OK\r \EATDT\T\r\c
CONNECT STTY=crtscts,crtsxoff

codex-fast =,-, "" \dA\pT&C1&D2*MF0*AA1&R1&S1*DE15*FL3S2=255S7=40S10=40*TT5&W\r\c OK\r
\EATDT\T\r\c CONNECT\s38400 STTY=crtscts,crtsxoff

tb9600-ec =W-,  "" \dA\pA\pA\pTE1V1X1Q0S2=255S12=255S50=6\r\c OK\r
\EATDT\T\r\cCONNECT\s9600 STTY=crtscts,crtsxoff

tb9600-nec =W-, "" \dA\pA\pA\pTE1V1X1Q0S2=255S12=255S50=6S180=0\r\c OK\r \EATDT\T\r\c
CONNECT\s9600 STTY=crtscts,crtsxoff
```

表 40–5 に、Dialers ファイルの send 文字列でよく使用されるエスケープ文字を示します。

表 **40–5** /etc/uucp/Dialers で使用するエスケープ文字

文字	説明
\b	バックスペース文字を送信または想定する
\c	改行、キャリッジリターンを抑止する
\d	遅延 (約 2 秒)
\D	Dialcodes 変換なしの電話番号またはトークン
\e	エコーチェックを使用しない
\E	エコーチェックを使用する (低速デバイス用)
\K	ブレーク文字を挿入する
\n	改行文字を送信する
\nnn	8 進数値を送信する。使用できるその他のエスケープ文字は、の597 ページの「UUCP /etc/uucp/Systems ファイル」を参照
\N	NULL 文字 (ASCII NUL) を送信または想定する
\p	一時停止 (約 12 ～ 14 秒)
\r	リターン
\s	スペース文字を送信または想定する
\T	Dialcodes 変換を伴う電話番号またはトークン

次に示すのは Dialers ファイルの penril エントリです。

```
penril =W-P "" \d> Q\c : \d-> s\p9\c )-W\p\r\ds\p9\c-) y\c : \E\TP> 9\c OK
```

最初に、電話番号引数の置換メカニズムが確立されます。その結果、= はすべて W (発信音待ち) で置き換えられ、- はすべて P (一時停止) で置き換えられるようになります。

上記の行の残りの部分に指定されているハンドシェークの働きは、次のとおりです。

- "" – 何も待たない(つまり次へ進む)
- \d – 2 秒間の遅延の後キャリッジリターンを送信する
- > – > を待つ
- Q\c – キャリッジリターンを付けずに Q を送信する
- : – : を待つ
- \d- – 2 秒間の遅延の後 - とキャリッジリターンを送信する
- > – > を待つ
- s\p9\c – s を送信し、一時停止し、9 を送信するが、キャリッジリターンは送信しない

-)-W\p\r\ds\p9\c-)-) を待つ。) が受信されない場合は、- 文字の間の文字列を処理する。つまり、W を送信し、一時停止し、キャリッジリターンを送信し、遅延し、s を送信し、一時停止し、9 を送信し、キャリッジリターンを送信しないで) を待つ
- y\c – キャリッジリターンを付けずに y を送信する
- :-: を待つ
- \E\TP – エコーチェックを有効にする。この時点以降は、1 文字送信すると、その文字が受信されるまでほかの作業を行わない。次に電話番号を送信する。\T は、引数として渡された電話番号をとることを意味する。\T は Dialcodes 変換と、このエントリのフィールド 2 で指定されたモデム機能変換を適用する。次に、P とキャリッジリターンを送信する
- >-> を待つ
- 9\c – 改行を付けずに 9 を送信する
- OK – 文字列 OK を待つ

UUCP ハードウェアフロー制御

擬似送信文字列 STTY=value を用いることによっても、モデムの特性を設定できます。たとえば、STTY=crtscts を使用すると、出力ハードウェアフロー制御が可能になります。STTY=crtsxoff を使用すると、入力ハードウェアフロー制御が可能になります。STTY=crtscts,crtsxoff を使用すると、入出力の両方のハードウェアフロー制御が可能になります。

STTY はすべての stty モードを受け入れます。詳細は、stty(1) と termio(7I) のマニュアルページを参照してください。

次の例は、Dialers ファイルエントリ内でハードウェアフロー制御を使用可能にしています。

```
dsi =,-, "" \dA\pTE1V1X5Q0S2=255S12=255*E1*F3*M1*S1\r\c OK\r \EATDT\T\r\c
CONNECT\sEC STTY=crtscts
```

この擬似送信文字列は、Systems ファイルのエントリの中でも使用できます。

UUCP パリティの設定

場合によっては、呼び出そうとしているシステムがポートのパリティを検査し、パリティに誤りがあると回線を切断することがあります。そのため、パリティのリセットが必要になります。expect-send の対を成す文字列として P_ZERO を使用すると、パリティが 0 に設定されます。

```
foo =,-, "" P_ZERO "" \dA\pTE1V1X1Q0S2=255S12=255\r\c OK\r\EATDT\T\r\c CONNECT
```

同様に、P_EVEN はパリティを偶数 (デフォルト) に、P_ODD はパリティを奇数に、P_ONE はパリティを 1 に設定します。この擬似送信文字列は、Systems ファイルのエントリの中でも使用できます。

その他の基本的な UUCP 構成ファイル

基本的な UUCP 構成を行うときに、Systems、 Devices、および Dialers の各ファイルに加えて、この節で紹介するファイルを使用できます。

UUCP /etc/uucp/Dialcodes ファイル

/etc/uucp/Dialcodes ファイルにより、/etc/uucp/Systems ファイルの Phone フィールドで使用するダイアルコードの省略名を定義できます。Dialcodes ファイルは、同じサイトにある複数のシステムが使用する基本的な電話番号について、付加的な情報を指定するために使用できます。

各エントリの書式は次のとおりです。

abbreviation dial-sequence

abbreviation は、Systems ファイルの Phone フィールドで使用される省略名で、*dial-sequence* は、個々の Systems ファイルのエントリがアクセスされたときにダイアラに渡されるダイアルシーケンスです。表 40–6 に、この 2 つのファイル間の対応関係を示します。

表 **40–6** Dialcodes ファイルと Systems ファイルの間の対応関係

	フィールド名					
Dialcodes	*Abbreviation*	Dial-Sequence				
Systems	System-Name	Time	Type	Speed	Phone	Chat Script

表 40–7 に示すのは、Dialcodes ファイルのエントリの例です。

表 **40–7** Dialcodes ファイルのエントリ

Abbreviation	**Dial-sequence**
NY	1=212
jt	9+847

最初の行の NY は、Systems ファイルの Phone フィールドで使用される省略名です。Systems ファイルのエントリは、たとえば次のようになります。

```
NY5551212
```

uucico は、Systems ファイルから NY を読み取ると、Dialcodes ファイルから NY を探し、それに該当するダイアルシーケンス 1=212 を取得します。これは、New York City への電話呼び出しに必要なダイアルシーケンスです。このシーケンスは、1 という番号と、一時停止して次の発信音を待つことを示す等号 (=) と、市外局番 212 で構成されています。uucico はこの情報をダイアラに送り、再び Systems ファイルに戻って残りの電話番号 5551212 を処理します。

jt 9=847- というエントリは、Systems ファイル内の jt7867 などのような Phone フィールドを取り扱います。uucico は、jt7867 を含むエントリを Systems ファイルから読み取り、ダイアラとトークンのペアの中のトークンが \T であれば、9=847-7867 というシーケンスをダイアラに送ります。

UUCP /etc/uucp/Sysfiles ファイル

/etc/uucp/Sysfiles ファイルでは、uucp と cu が Systems、Devices、Dialers ファイルとして使用する別のファイルを割り当てます 。cu の詳細は、cu (1C) のマニュアルページを参照してください。Sysfiles は次の目的に使用できます。

- 別の Systems ファイルにより、uucp のサービスとは異なるアドレスに対してログインサービスを要求できます。
- 別の Dialers ファイルにより、cu と uucp で異なるハンドシェークを割り当てることができます。
- 複数の Systems、Dialers、Devices ファイル。特に Systems ファイルはサイズが大きくなるので、いくつかの小さいファイルに分割しておくと便利です。

Sysfiles ファイルの形式は次のとおりです。

```
service=w systems=x:x dialers=y:y devices=z:z
```

w には、uucico、cu、またはその両方をコロンで区切って指定します。*x* には、Systems ファイルとして使用される 1 つまたは複数のファイルをコロンで区切って指定します。これらは指定された順序で読み込まれます。*y* は Dialers ファイルとして使用される 1 つまたは複数のファイルです。*z* は Devices ファイルとして使用される 1 つまたは複数のファイルです。

完全パスで指定しない限り、各ファイル名は /etc/uucp ディレクトリからの相対パスとみなされます。

次に示すのは、標準の /etc/uucp/Systems に加えて使用するローカル Systems ファイル (Local_Systems) を定義する /etc/uucp/Sysfiles の例です。

```
service=uucico:cu systems=Systems :Local_Systems
```

/etc/uucp/Sysfiles の中にこのエントリがある場合、uucico と cu はどちらも、まず標準 /etc/uucp/Systems ファイルを調べます。呼び出そうとしているシステムのエントリがそのファイル内にないか、またはそのファイル内の該当エントリの処理に失敗した場合は、両コマンドは /etc/uucp/Local_Systems を調べます。

上記のエントリの場合は、cu と uucico は、Dialers ファイルと Devices ファイルを共有します。

uucico サービス用と cu サービス用に別の Systems ファイルを定義した場合は、マシンは 2 つの異なる Systems のリストを持つことになります。uucico リストは uuname コマンドを使用して表示でき、cu リストは uuname -C コマンドを使用して表示できます。このファイルのもう1 つの例として、代替ファイルの方を先に調べ、デフォルトファイルは必要なときだけ調べる場合を次に示します。

```
service=uucico systems=Systems.cico:Systems
  dialers=Dialers.cico:Dialers \
devices=Devices.cico:Devices
  service=cu systems=Systems.cu:Systems \
dialers=Dialers.cu:Dialers \
  devices=Devices.cu:Devices
```

UUCP /etc/uucp/Sysname ファイル

UUCP を使用するすべてのマシンは、ノード名と呼ばれる識別名を持っている必要があります。この名前は、リモートマシンの /etc/uucp/Systems ファイルに、チャットスクリプトやその他の識別情報とともに格納されます。通常は、UUCP は、uname -n コマンドから返されるものと同じノード名を使用し、TCP/IP でもこの名前を使用します。

/etc/uucp/Sysname ファイルを作成することによって、TCP/IP ホスト名とは別の UUCP ノード名を指定できます。このファイルには、ローカルシステムの UUCP ノード名が入った 1 行のエントリが含まれています。

UUCP /etc/uucp/Permissions ファイル

/etc/uucp/Permissions ファイルは、ログイン、ファイルアクセス、およびコマンド実行に関するリモートコンピュータのアクセス権を指定します。リモートコンピュータがファイルを要求する権限と、ローカルマシンでキューに入れられたファイルを受け取る権限を制限するオプションがあります。また、リモートマシンがローカルコンピュータ上で実行できるコマンドを指定するオプションもあります。

UUCP 構造のエントリ

各エントリは 1 行の論理行で、行末にバックスラッシュ (\) がある場合は次の行と継続していることを示します。エントリは、スペースで区切られたオプションから 構成されます。各オプションは、次の形式の名前と値のペアです。

name=value

values はコロンで区切ってリストとすることもできます。オプション指定の中では、スペースは使用できないので注意してください。

コメント行はポンド記号 (#) で始まり、その行の改行文字までの全部分を占めます。空行は無視されます (複数行エントリの中の空行も同じです)。

`Permissions` ファイルのエントリの種類を次に示します。

- `LOGNAME` – リモートマシンがローカルマシンにログインする (呼び出す) ときに有効になるアクセス権を指定する。

注 – リモートマシンがローカルマシンを呼び出すとき、固有のログインと検証可能なパスワードを使用しない限り、そのリモートマシンの識別情報は正確なものとはなりません。

- `MACHINE` – ローカルマシンがリモートコンピュータにログインする (呼び出す) ときに有効になるアクセス権を指定する。

`LOGNAME` エントリには LOGNAME オプションが含まれ、`MACHINE` エントリには MACHINE オプションが含まれます。1 つのエントリに両方のオプションを含めることもできます。

UUCP の考慮事項

`Permissions` ファイルを使用して、リモートコンピュータに付与されているアクセスのレベルを制限するときは、以下のことを考慮に入れる必要があります。

- リモートコンピュータが、UUCP 通信を目的としてログインするために使用するすべてのログイン ID は、1 つの `LOGNAME` エントリだけに指定されていなければならない。
- 呼び出されたサイトの名前が `MACHINE` エントリにない場合、そのサイトには次に示すデフォルトのアクセス権または制約が適用される。
  - ローカルの送信要求と受信要求は実行される
  - リモートコンピュータは、ローカルコンピュータの `/var/spool/uucppublic` ディレクトリにファイルを送信できる
  - リモートコンピュータがローカルコンピュータで実行するために送信するコマンドは、デフォルトのコマンドのどれかでなければならない (通常は `rmail`)

UUCP `REQUEST` オプション

リモートコンピュータがローカルコンピュータを呼び出し、ファイルの受信を要求したときに、その要求を承認することも拒否することもできます。`REQUEST` オプションは、リモートコンピュータがローカルコンピュータからのファイル転送を要求できる

かどうかを指定します。REQUEST=yes は、リモートコンピュータがローカルコンピュータからのファイル転送を要求できることを指定します。REQUEST=no は、リモートコンピュータがローカルコンピュータからのファイルの受信を要求できないことを指定します。REQUEST=no は、REQUEST オプションを指定しなかった場合に使用されるデフォルト値です。REQUEST オプションは、LOGNAME エントリ (リモートコンピュータがローカルコンピュータを呼び出す場合) と、MACHINE エントリ (ローカルコンピュータがリモートコンピュータを呼び出す場合) のどちらにも使用できます。

UUCP SENDFILES オプション

リモートコンピュータがローカルコンピュータを呼び出す作業を完了した後で、ローカルコンピュータのキュー中のリモートコンピュータ用の作業を受け取ろうとすることがあります。SENDFILES オプションは、ローカルコンピュータが、リモートコンピュータ用にキューに入れた作業を送信できるかどうかを指定します。

文字列 SENDFILES=yes は、リモートコンピュータが LOGNAME オプションに指定されている名前の 1 つを使用してログインしていれば、ローカルコンピュータがキューに入れた作業を送信できることを指定します。/etc/uucp/Systems の Time フィールドに Never を入力してある場合は、この文字列の使用は必須です。その場合、ローカルマシンは受動モードに設定され、相手のリモートコンピュータへの呼び出しを開始することはできなくなります。詳細は、597 ページの「UUCP /etc/uucp/Systems ファイル」を参照してください。

文字列 SENDFILES=call は、ローカルコンピュータがリモートコンピュータを呼び出したときに限り、ローカルコンピュータのキュー中のファイルを送信することを指定します。call の値は SENDFILES オプションのデフォルト値です。MACHINE エントリはリモートコンピュータへの呼び出しを送る場合に適用されるものなので、このオプションが意味を持つのは LOGNAME エントリの中で使用した場合だけです。MACHINE エントリでこのオプションを使用しても無視されます。

UUCP MYNAME オプション

このオプションを使用すると、hostname コマンドから戻される TCP/IP ホスト名以外に、固有の UUCP ノード名をローカルシステムに与えることができます。たとえば、偶然に他のシステムと同じ名前をローカルホストに付けてしまった場合などに、Permissions ファイルの MYNAME オプションを指定できます。たとえば、自分の所属組織が widget という名前で認識されるようにする場合を考えます。すべてのモデムが gadget というホスト名を持つマシンに接続されている場合は、gadget の Permissions ファイルに次のようなエントリを含めることができます。

```
service=uucico systems=Systems.cico:Systems
  dialers=Dialers.cico:Dialers \
  devices=Devices.cico:Devices
service=cu systems=Systems.cu:Systems \
  dialers=Dialers.cu:Dialers \
```

```
devices=Devices.cu:Devices
```

これで、システム world は、あたかも widget にログインしているかのようにマシン gadget にログインできます。ローカルマシンから world マシンを呼び出したときにも、world が widget という別名で認識するようにしたい場合は、次のようなエントリを作成します。

```
MACHINE=world MYNAME=widget
```

MYNAME オプションにより、ローカルマシンが自分自身を呼ぶこともできるので、テストの目的にも利用できます。しかし、このオプションはマシンの実際の識別情報を隠す目的にも使用できてしまうので、622 ページの「UUCP VALIDATE オプション」で述べる VALIDATE オプションを使用するようにしてください。

UUCP READ オプションと WRITE オプション

これらのオプションは、uucico がファイルシステムのどの部分を読み書きできるかを指定します。READ オプションと WRITE オプションは、MACHINE エントリと LOGNAME エントリのどちらにも使用できます。

次の文字列に示すように、READ オプションと WRITE オプションのどちらも、デフォルトは uucppublic ディレクトリです。

```
READ=/var/spool/uucppublic WRITE=/var/spool/uucppublic
```

文字列 READ=/ と WRITE=/ は、Other 権を持つローカルユーザーがアクセスできるすべてのファイルにアクセスできる権限を指定します。

これらのエントリの値は、コロンで区切ったパス名のリストです。READ オプションはリモート側からのファイル要求のためのものであり、WRITE オプションはリモート側からのファイル送出のためのものです。値の 1 つは、入力ファイルまたは出力ファイルの完全パス名の接頭辞でなければなりません。公共ディレクトリの他に /usr/news にもファイルにも送出する権限を付与するには、WRITE オプションに次の値を指定します。

```
WRITE=/var/spool/uucppublic:/usr/news
```

パス名はデフォルトのリストに追加されるものではないので、READ オプションと WRITE オプションを使用するときはすべてのパス名を指定する必要があります。たとえば、WRITE オプションでパス名として /usr/news のみを指定した場合、公共ディレクトリにファイルを送出する権限は失われます。

リモートシステムがどのディレクトリに読み書きのアクセスができるかは、注意して決定しなければなりません。たとえば、/etc ディレクトリには多数の重要なシステムファイルが入っています。したがって、このディレクトリにファイルを送出する権限はリモートユーザーには付与しない方が賢明です。

UUCP NOREAD オプションと NOWRITE オプション

NOREAD オプションと NOWRITE オプションは、READ オプションと WRITE オプションおよびデフォルトに対する例外を指定します。次のエントリは、/etc ディレクトリ (およびこの下の各サブディレクトリ。このパス名は接頭辞であることを忘れないでください) の中のファイルを除くすべてのファイルの読み取りを許可しています。

```
READ=/ NOREAD=/etc WRITE=/var/spool/uucppublic
```

このエントリは、デフォルトの /var/spool/uucppublic ディレクトリへの書き込みだけを許可しています。NOWRITE も NOREAD オプションと同じ形で働きます。NOREAD オプションと NOWRITE オプションは、LOGNAME エントリと MACHINE エントリのどちらにも使用できます。

UUCP CALLBACK オプション

LOGNAME エントリの中で CALLBACK オプションを使用すると、呼び出し側システムがコールバックするまで、トランザクションを一切行わないことを指定できます。CALLBACK を設定する理由を次に示します。

- セキュリティ – マシンをコールバックすることで、それが正しいマシンであることを確認できる。
- 課金 – 大量のデータの伝送を行うときに、その長時間の呼び出しの料金を課すマシンを選択できる。

文字列 CALLBACK=yes は、ファイル転送を行う前に、ローカルコンピュータがリモートコンピュータをコールバックしなければならないということを指定します。

CALLBACK オプションのデフォルトは CALLBACK=no です。CALLBACK を yes に設定する場合は、呼び出し側に対応する MACHINE エントリの中で、以後の通信に影響を与えるアクセス権を指定する必要があります。これらのアクセス権は、LOGNAME の中で指定しないでください。また同様に、リモートマシンがローカルホストについて設定した LOGNAME エントリの中でも指定しないでください。

注 – 2 つのサイトが互いに CALLBACK オプションを設定すると、通信が開始されないので注意してください。

UUCP COMMANDS オプション

注意 – COMMANDS オプションは、システムのセキュリティを低下させる恐れがあります。このオプションは十分に注意して使用してください。

COMMANDS オプションは、リモートコンピュータがローカルコンピュータ上で実行できるコマンドを指定するために、MACHINE エントリの中で使用できます。uux プログラムは、リモート実行要求を生成し、それらの要求をリモートコンピュータに転送するためにキューに入れます。ファイルとコマンドはターゲットコンピュータに送られて、リモート実行されます。MACHINE エントリは、ローカルシステムが呼び出しを行う場合に限り適用されるという規則がありますが、このオプションは例外です。

COMMANDS は LOGNAME エントリの中では使えないという点に注意してください。MACHINE エントリの中の COMMANDS は、ローカルシステムがリモートシステムを呼び出すのか、リモートシステムがローカルシステムを呼び出すのかに関係なく、コマンド権限を定義します。

リモートコンピュータがローカルコンピュータ上で実行できるデフォルトのコマンドは、文字列 COMMANDS=rmail となります。MACHINE エントリの中で COMMANDS=rmail 文字列を使用した場合は、デフォルトのコマンドは無効化されます。たとえば、次のエントリは、COMMANDS のデフォルトを無効にして、owl、raven、hawk、dove という名前の各コンピュータが、rmail、rnews、lp の各コマンドをローカルコンピュータで実行できるようにします。

```
MACHINE=owl:raven:hawk:dove COMMANDS=rmail:rnews:lp
```

上記で指定した名前に加えて、コマンドの完全パス名も指定できます。たとえば、次のエントリは、 rmail コマンドがデフォルトの検索パスを使用することを指定しています。

```
COMMANDS=rmail:/usr/local/rnews:/usr/local/lp
```

UUCP のデフォルトの検索パスは、/bin と /usr/bin です。リモートコンピュータが、実行するコマンドとして rnews または /usr/local/rnews を指定した場合は、デフォルトのパスに関係なく /usr/local/rnews が実行されます。同様に、実行される lp コマンドは /usr/local/lp です。

リストに ALL という値を含めると、エントリに指定されたリモートコンピュータから、すべてのコマンドが実行できます。この値を使用した場合は、リモートコンピュータにローカルマシンへのフルアクセスを与えることになります。

注意 – これは、通常のユーザーが持っているよりもはるかに多くのアクセス権を与えることになります。この値を使用するのは、両方のマシンが同じサイトにあり、緊密に接続されていて、ユーザーが信頼できる場合に限定するようにしてください。

ALL が追加された文字列を次に示します。

```
COMMANDS=/usr/local/rnews:ALL:/usr/local/lp
```

この文字列は、次の 2 点を示しています。

- ALL の値は文字列の中のどこでも使用できる

- 要求された rnews、lp コマンドに完全パス名が指定されていない場合は、デフォルトではなく、それぞれについて指定されているパス名が使用される

COMMANDS オプションで cat や uucp などのように、潜在的な危険性のあるコマンドを指定するときは、VALIDATE オプションを使用するようにしてください。UUCP リモート実行デーモン (uuxqt) により実行する場合、ファイルを読み書きをするコマンドは、どれもローカルセキュリティにとって危険性のあるものとなります。

UUCP VALIDATE オプション

VALIDATE オプションは、マシンのセキュリティにとって危険性があると考えられるコマンドを指定するときに、COMMANDS オプションといっしょに使用します。VALIDATE は、コマンドアクセスを開放する方法としては ALL より安全ですが、COMMANDS オプションのセキュリティのレベルを補強するだけのものです。

VALIDATE は、呼び出し側マシンのホスト名と、そのマシンが使用しているログイン名とを相互にチェックするものであり、呼び出し側の識別情報について、ある程度の検証機能を備えています。この例では、widget または gadget 以外のマシンが Uwidget としてログインしようとすると、接続は拒否されます。

```
LOGNAME=Uwidget VALIDATE=widget:gadget
```

VALIDATE オプションを使用する場合、権限が与えられたコンピュータは UUCP トランザクション用に固有のログインとパスワードを持っていなければなりません。この認証処理では、このエントリに対応するログインとパスワードを保護することが重要な条件の 1 つです。部外者がこの情報を入手してしまうと、VALIDATE オプションはセキュリティに関する役割をまったく果たさなくなります。

UUCP トランザクションについて、特権を持つログインとパスワードをどのリモートコンピュータに付与するかについては、十分に検討する必要があります。ファイルアクセスとリモート実行の権限をリモートコンピュータに与えるということは、そのリモートコンピュータのすべてのユーザーに対して、ローカルコンピュータに対する通常のログインとパスワードを与えるのと同じことです。したがって、リモートコンピュータに信頼の置けないユーザーがいると判断した場合は、そのコンピュータには特権的なログインとパスワードは付与しないようにしてください。

次のような LOGNAME エントリは、あるリモートコンピュータが eagle、owl、hawk のどれかとしてローカルコンピュータにログインする場合は、そのコンピュータはログイン uucpfriend を使用する必要があることを指定します。

```
LOGNAME=uucpfriend VALIDATE=eagle:owl:hawk
```

部外者が uucpfriend を入手したとすれば、簡単に偽装することができます。

それでは、MACHINE エントリの中でだけ使用される COMMANDS オプションに対して、このエントリはどのような効果を持つのでしょうか。このエントリは、MACHINE エントリ (および COMMANDS オプション) を、特権ログインに対応する LOGNAME エントリにリンクします。このリンクが必要なのは、リモートコンピュータがログインしている時点では、実行デーモンはまだ動作していないためです。実際に、このリンク

はどのコンピュータが実行要求を送ったのかを認識しない非同期プロセスです。ここで問題になるのが、実行ファイルがどこから送られてきたのかを、ローカルコンピュータがどのようにして知るかという点です。

各リモートコンピュータは、ローカルマシン上にそれぞれ専用スプールディレクトリを持っています。これらのスプールディレクトリの書き込み権限は、UUCP プログラムだけに与えられています。リモートコンピュータからの実行ファイルは、ローカルコンピュータに転送された後に、このスプールディレクトリに入れられます。uuxqt デーモンが動作するときには、スプールディレクトリ名を使用して、Permissions ファイルから MACHINE エントリを見つけ、COMMANDS リストを取得します。Permissions ファイル内に該当するコンピュータ名が見つからない場合は、デフォルトのリストが使用されます。

次の例は、MACHINE エントリと LOGNAME エントリの関係を示しています。

```
MACHINE=eagle:owl:hawk REQUEST=yes \
COMMANDS=rmail:/usr/local/rnews \
READ=/ WRITE=/
LOGNAME=uucpz VALIDATE=eagle:owl:hawk \
REQUEST=yes SENDFILES=yes \
READ=/ WRITE=/
```

COMMANDS オプションの値は、リモートユーザーが、rmail と /usr/local/rnews を実行できることを示しています。

最初のエントリでは、リストされているコンピュータのどれかを呼び出したい場合に、実際には eagle、owl、hawk のどれかを呼び出すということを理解しておく必要があります。したがって、eagle、owl、および hawk のスプールディレクトリに置かれるファイルはすべて、それらのコンピュータのどれかが投入したことになります。あるリモートコンピュータがログインし、この 3 つのコンピュータのどれかであることを主張した場合、その実行ファイルもこの特権スプールディレクトリに入れられます。したがって、ローカルコンピュータでは、そのコンピュータが特権ログイン uucpz を持っていることを確認する必要があります。

UUCP OTHER 用の MACHINE エントリ

特定の MACHINE エントリに記述されていないリモートマシンについて、異なるオプション値を指定したい場合があります。これが必要になるのは、多数のコンピュータがローカルホストを呼び出し、コマンドセットがそのたびに異なるような場合です。次の例に示すように、このようなエントリでは、コンピュータ名として OTHER という名前を使用します。

```
MACHINE=OTHER \
COMMANDS=rmail:rnews:/usr/local/Photo:/usr/local/xp
```

他の MACHINE エントリに記述されていないコンピュータについても、MACHINE エントリに使用できるすべてのオプションを設定できます。

UUCP の MACHINE エントリと LOGNAME エントリの結合

共通オプションが同じである場合、MACHINE エントリと LOGNAME エントリを結合して、単一のエントリにすることができます。たとえば、次の 2 セットのエントリは、同じ REQUEST、READ、WRITE オプションを共有しています。

```
MACHINE=eagle:owl:hawk REQUEST=yes \
READ=/ WRITE=/

LOGNAME=uupz REQUEST=yes SENDFILES=yes \
READ=/ WRITE=/
```

この 2 つのエントリを結合したものを次に示します。

```
MACHINE=eagle:owl:hawk REQUEST=yes \
logname=uucpz SENDFILES-yes \
READ=/ WRITE=/
```

MACHINE エントリと LOGNAME エントリを結合することによって、Permissions ファイルは、効率的で管理しやすくなります。

UUCP の転送

一連のマシンを介してファイルを送信するときは、リレー (中継) マシンの COMMANDS オプションの中に uucp コマンドが含まれていなければなりません。次のコマンドを入力した場合、マシン willow がマシン oak に対して uucp プログラムの実行を許可する場合に限り、この転送操作は正常に機能します。

```
% uucp sample.txt oak\!willow\!pine\!/usr/spool/uucppublic
```

oak もローカルマシンに uucp のプログラムの実行を許可している必要があります。最終宛先マシンである pine は、転送動作を行わないため、uucp コマンドを許可する必要はありません。通常、マシンはこのように設定されていません。

UUCP /etc/uucp/Poll ファイル

/etc/uucp/Poll ファイルには、リモートコンピュータをポーリングするための情報が入っています。Poll ファイル内の各エントリには、呼び出すリモートコンピュータの名前と、それに続くタブ文字またはスペース、最後にそのコンピュータを呼び出す時刻が入ります。Poll ファイル内のエントリの形式は次のとおりです。

sys-name hour ...

たとえば次のようなエントリを指定したとします。

```
eagle 0 4 8 12 16 20
```

これは、コンピュータ eagle を 4 時間おきにポーリングします。

uudemon.poll スクリプトは Poll ファイルを処理しますが、実際にポーリングを行うわけではありません。単にスプールディレクトリ内にポーリング作業ファイル (名前は常に C.*file*) を設定するだけです。uudemon.poll スクリプトはスケジューラを起動し、スケジューラは、スプールディレクトリ内のすべての作業ファイルを調べます。

UUCP /etc/uucp/Config ファイル

/etc/uucp/Config ファイルを使用すると、いくつかのパラメータを手動で上書きできます。Config ファイルの各エントリの形式は次のとおりです。

parameter=value

構成可能な全パラメータ名のリストについては、システムに付属している Config ファイルを参照してください。

次の Config エントリは、デフォルトのプロトコル順序を Gge に設定し、G プロトコルのデフォルト値を、ウィンドウ数 7、パケットサイズ 512 バイトに変更します。

```
Protocol=G(7,512)ge
```

UUCP /etc/uucp/Grades ファイル

/etc/uucp/Grades ファイルには、リモートコンピュータへのジョブをキューに入れるときに指定できるジョブグレードが入っています。また、個々のジョブグレードに関するアクセス権も含まれています。このファイルのエントリは、ユーザーがジョブをキューに入れるときに使用する、管理者が定義したジョブグレードの定義を表しています。

Grades ファイルのエントリの形式は次のとおりです。

User-job-grade System-job-grade Job-size Permit-type ID-list

各エントリには、スペースで区切ったいくつかのフィールドがあります。エントリの最後のフィールドは、同じくスペースで区切ったいくつかのサブフィールドから構成されます。1 つのエントリが複数の物理行にわたる場合は、バックスラッシュを使用して、エントリを次の行に継続させることができます。コメント行はポンド記号 (#) で始まり、その行の全体を占めます。空の行は常に無視されます。

UUCP User-job-grade フィールド

このフィールドには、管理者が 64 文字以内で定義したユーザージョブのグレード名が入ります。

UUCP System-job-grade フィールド

このフィールドには、*User-job-grade* が対応付けされる 1 文字のジョブグレードが入ります。有効な文字は A ～ Z、a ～ z で、最も優先順位が高いのは A、最も優先順位が低いのは z です。

ユーザージョブグレードとシステムジョブグレードの関係

ユーザージョブグレードは複数のシステムジョブグレードに割り当てることができます。`Grades` ファイルは、ユーザージョブグレードのエントリを見つけるために先頭から検索されるという点に注意してください。したがって、最大ジョブサイズの制限値に応じて、複数のシステムジョブグレードのエントリが列挙されます。

ユーザージョブグレードの最大数には制限はありませんが、システムジョブグレードの許容最大数は 52 です。その理由は、1 つの *System-job-grade* には複数の *User-job-grade* を対応付けできるが、個々の *User-job-grade* はファイル内でそれぞれ単独の行でなければならないという点にあります。次に例を示します。

```
mail N Any User Any netnews N Any User Any
```

`Grades` ファイル内でこのような構成をした場合、2 つの *User-job-grade* が同じ *System-job-grade* を共有します。ジョブグレードに関するアクセス権は、*System-job-grade* ではなく *User-job-grade* に割り当てられるものなので、2 つの *User-job-grade* は同じ *System-job-grade* を共有しながら、それぞれ異なるアクセス権のセットを持つことができます。

デフォルトグレード

デフォルトのユーザージョブグレードとして、システムジョブグレードを割り当てることができます。そのためには、`Grades` ファイルの *User-job-grade* フィールドのユーザージョブグレードとしてキーワード default を使用し、そのデフォルトに割り当てるシステムジョブグレードを指定します。Restriction フィールドと ID フィールドは `Any` と定義して、どのようなユーザー、どのようなサイズのジョブでも、このグレードでキューに入れることができるようにします。次に例を示します。

```
default a Any User Any
```

デフォルトのユーザージョブグレードを定義しなかった場合は、組み込まれているデフォルトグレードである z が使用されます。Restriction フィールドのデフォルトは `Any` なので、デフォルトグレードのエントリが複数存在していても検査されません。

UUCP Job-size フィールド

このフィールドは、キューに入れることのできる最大ジョブサイズを指定します。*Job-size* はバイト数で表され、表 40–8 に示すオプションを使用できます。

表 **40–8** Job-size フィールド

nnnn	このジョブグレードの最大ジョブサイズを指定する整数
n `K`	K バイト数を表す 10 進数 (`K` はキロバイトの略号)
n `M`	M バイト数を表す 10 進数 (`M` はメガバイトの略号)
`Any`	最大ジョブサイズが指定されないことを指定するキーワード

次に例をいくつか示します。

- `5000` は 5000 バイトを表す
- `10K` は 10K バイトを表す
- `2M` は 2M バイトを表す

UUCP Permit-type フィールド

このフィールドには、ID リストをどのように解釈するかを指示するキーワードを指定します。表 40–9 に、キーワードとそれぞれの意味を示します。

表 **40–9** Permit-type フィールド

キーワード	**ID** リストの内容
`User`	このジョブグレードの使用を許可されているユーザーのログイン名
`Non-user`	このジョブグレードの使用を許可されていないユーザーのログイン名
`Group`	このジョブグレードの使用を許可されているメンバーのグループ名
`Non-group`	このジョブグレードの使用を許可されていないメンバーのグループ名

UUCP ID-list フィールド

このフィールドには、このジョブグレードへキューを入れることが許可、禁止されるログイン名またはグループ名のリストが入ります。名前のリストはそれぞれスペースで区切り、改行文字で終了します。このジョブグレードへキューを入れることを誰にでも許可する場合は、キーワード `Any` を使用します。

その他の UUCP 構成ファイル

この節では、UUCP の機能に影響を与えるファイルのうち、比較的変更頻度の低い 3 つのファイルについて説明します。

UUCP /etc/uucp/Devconfig ファイル

/etc/uucp/Devconfig ファイルを使用するとサービス別、つまり uucp 用と cu 用とに分けて、デバイスを構成できます。Devconfig のエントリは、個々のデバイスで使用される STREAMS モジュールを定義します。書式は以下の通りです。

service=*x* device= *y* push= *z*[:*z*...]

x は、cu か uucico、またはその両方のサービスをコロンで区切ったものです。*y* はネットワークの名前で、これは Devices ファイルのエントリに一致していなければなりません。*z* には、STREAMS モジュールの名前を、Stream にプッシュする順序で指定します。cu サービスと uucp サービスについて、それぞれ異なるモジュールとデバイスを定義できます。

次のエントリは STARLAN ネットワーク用のもので、このファイル内で最もよく使用されるものです。

```
service=cu       device=STARLAN     push=ntty:tirdwr
service=uucico   device=STARLAN     push=ntty:tirdwr
```

この例では、まず ntty、次に tirdwr がプッシュされます。

UUCP /etc/uucp/Limits ファイル

/etc/uucp/Limits ファイルは、uucp ネットワーク処理で同時に実行できる uucico、uuxqt、および uusched の最大数を制御します。ほとんどの場合は、デフォルトの値が最適であり、変更の必要はありません。変更したい場合は、任意のテキストエディタを使用してください。

Limits ファイルの形式は次のとおりです。

service=*x* max= *y*:

x は uucico、uuxqt、uusched のどれかで、*y* はそのサービスについての制限値です。フィールドは、小文字を使用して任意の順序で入力できます。

次に示すのは、Limits ファイルの中で一般的に使用される内容です。

```
service=uucico max=5
service=uuxqt max=5
service=uusched max=2
```

この例は、5 つの uucico、5 つの uuxqt、2 つの uusched をマシンで実行できることを示しています。

UUCP remote.unknown ファイル

通信機能の使用に影響を与えるファイルとして、もう 1 つ remote.unknown ファイルがあります。このファイルは、どの Systems ファイルにも含まれていないマシンが通信を開始したときに実行されるバイナリプログラムです。このプログラムはその通信をログに記録し、接続を切断します。

注意 – remote.unknown ファイルのアクセス権を変更して、このプログラムが実行できないようにすると、ローカルシステムはどのシステムからの接続も受け入れることになります。

このプログラムが実行されるのは、どの Systems ファイルにも含まれていないマシンが対話を開始した場合です。このプログラムは、その対話を記録し、接続を失敗させます。このファイルのアクセス権を変更して実行できないようにしてしまうと (chmod 000 remote.unknown)、ローカルシステムはすべての通信要求を受け入れることになります。妥当な理由がない限り、この変更は行わないようにしてください。

UUCP の管理ファイル

次に、UUCP 管理ファイルについて説明します。これらのファイルは、デバイスのロック、一時データの保管、リモート転送や実行に関する情報の保存などのために、スプールディレクトリ内に作成されます。

- 一時データファイル (TM) – これらのデータファイルは、他のコンピュータからファイルを受け取るときに、UUCP プロセスによりスプールディレクトリ /var/spool/uucp/*x* の下に作成されます。ディレクトリ *x* は、ファイルを送信しているリモートコンピュータと同じ名前です。一時データファイル名の形式は次のとおりです。

  TM.*pid.ddd*

  pid はプロセス ID、*ddd* は 0 から始まる 3 桁のシーケンス番号です。

  ファイル全体が受信されると、TM.*pid.ddd* ファイルは、伝送を発生させた C.*sysnxxxx* ファイル (以下で説明します) の中で指定されているパス名に移されます。処理が異常終了した場合は、TM.*pid.ddd* ファイルが *x* ディレクトリ内に残ることがあります。このファイルは、uucleanup を使用することにより自動的に削除されます。

- ロックファイル (LCK) – ロックファイルは、使用中の各デバイスごとに、/var/spool/locks ディレクトリ内に作成されます。ロックファイルは、対話の重複、複数の試行による同じ呼び出しデバイスの使用が発生するのを防ぎます。表 40–10 に、UUCP ロックファイルの種類を示します。

表 **40–10** UUCP ロックファイル

ファイル名	説明
LCK.. *sys*	*sys* はファイルを使用しているコンピュータ名を表す
LCK. *dev*	*dev* はファイルを使用しているデバイス名を表す
LCK .*LOG*	*LOG* はロックされている UUCP ログファイルを表す

通信リンクが予定外のときに切断された場合 (通常コンピュータがクラッシュしたとき)、これらのファイルがスプールディレクトリ内に残ることがあります。親プロセスが有効でなくなった後は、ロックファイルは無視 (削除) されます。ロックファイルには、ロックを引き起こしたプロセスのプロセス ID が入っています。

- 作業ファイル (C.) – 作業ファイルは、リモートコンピュータに送る作業 (ファイル転送またはリモートコマンド実行) がキューに入れられたときに、スプールディレクトリ内に作成されます。作業ファイル名の形式は次のとおりです。

  C.*sysnxxxx*

  sys はリモートコンピュータ名、*n* は作業のグレード (優先順位) を表す ASCII 文字、*xxxx* は、UUCP が割り当てる 4 桁のジョブシーケンス番号です。作業ファイルには次の情報が含まれています。

  - 送信または要求するファイルの完全パス名
  - 宛先、ユーザー名、またはファイル名を表す完全パス名
  - ユーザーのログイン名
  - オプションのリスト
  - スプールディレクトリ内の関連データファイルの名前。uucp -C オプションか uuto -p オプションが指定されている場合は、ダミー名 (D.0) が使用される
  - ソースファイルのモードビット
  - 転送完了の通知を受け取るリモートユーザーのログイン名

- データファイル (D.) – コマンド行でスプールディレクトリへのソースファイルのコピーを指定すると、データファイルが作成されます。データファイル名の形式は次のとおりです。

  D.*systmxxxxyyy* – *systm* はリモートコンピュータ名の最初の 5 文字で、*xxxx* は uucp が割り当てる 4 桁のジョブシーケンス番号です。4 桁のジョブシーケンス番号の後にサブシーケンス番号を続けることができます。 *yyy* は、1 つの作業(C.) ファイルについて複数の D. ファイルが作成された場合に使用されます。

- X. (実行ファイル) – 実行ファイルは、リモートコマンドの実行の前にスプールディレクトリ内に作成されます。実行ファイル名の形式は次のとおりです。

X.*sysnxxxx*

sys はリモートコンピュータ名で、*n* は作業のグレード (優先順位) を表す文字です。*xxxx* は、UUCP が割り当てる 4 桁のシーケンス番号です。実行ファイルには次の情報が入ります。

- 要求元のログイン名とコンピュータ名
- 実行に必要なファイル名
- コマンド文字列への標準入力として使用する入力
- コマンド実行の標準出力を受け取るコンピュータとファイルの名前
- コマンド文字列
- 終了ステータスの要求のためのオプション行

UUCP のエラーメッセージ

この節には、UUCP に関連したエラーメッセージを示します。

UUCP の ASSERT エラーメッセージ

表 40–11 に ASSERT エラーメッセージを示します。

表 40–11 ASSERT エラーメッセージ

エラーメッセージ	説明または処置
CAN'T OPEN	open() または fopen() が失敗した
CAN'T WRITE	write()、fwrite()、fprint()、または類似のコマンドが失敗した
CAN'T READ	read()、fgets()、または類似のコマンドが失敗した
CAN'T CREATE	creat() 呼び出しが失敗した
CAN'T ALLOCATE	動的割り当てが失敗した
CAN'T LOCK	LCK (ロック) ファイルを作成しようとしたが失敗した。場合によっては、これは重大なエラーとなる
CAN'T STAT	stat() 呼び出しが失敗した
CAN'T CHMOD	chmod() 呼び出しが失敗した
CAN'T LINK	link() 呼び出しが失敗した
CAN'T CHDIR	chdir() 呼び出しが失敗した

表 **40–11** ASSERT エラーメッセージ　*(続き)*

エラーメッセージ	説明または処置
CAN'T UNLINK	unlink() 呼び出しが失敗した
WRONG ROLE	内部ロジックの問題
CAN'T MOVE TO CORRUPTDIR	不良な C. ファイルまたは X. ファイルを、/var/spool/uucp/.Corrupt ディレクトリに移動しようとしたが失敗した。このディレクトリが存在しないか、モードまたは所有者が正しくない
CAN'T CLOSE	close() または fclose() 呼び出しが失敗した
FILE EXISTS	C. ファイルまたは D. ファイルを作成しようとしたが、そのファイルがすでに存在している。このエラーは、シーケンスファイルのアクセスに問題がある場合に生じる。これは通常、ソフトエラーを示す
NO uucp SERVICE NUMBER	TCP/IP 呼び出しを試みたが、/etc/services 内に UUCP に関するエントリがない
BAD UID	ユーザー ID がパスワードデータベース内にない。ネームサービス構成の検査が必要
BAD LOGIN_UID	前記と同じ
BAD LINE	Devices ファイル内に不良な行がある。引数が足りない行が 1 つ以上ある
SYSLST OVERFLOW	gename.c の内部テーブルがオーバーフローした。1 つのジョブが 30 を超えるシステムに接続しようとした
TOO MANY SAVED C FILES	前記と同じ
RETURN FROM fixline ioctl	失敗するはずのない ioctl(2) が失敗した。システムドライバに問題がある
BAD SPEED	Devices ファイルまたは Systems ファイルの中に不適正な回線速度がある (Class フィールドまたは Speed フィールド)
BAD OPTION	Permissions ファイルの中に不適正な行またはオプションがある。ただちに修正が必要
PKCGET READ	リモートマシンがハングアップした可能性がある。処置は不要
PKXSTART	リモートマシンが回復不可能な状態で異常終了した。通常このエラーは無視できる
TOO MANY LOCKS	内部的な問題がある。システムの購入先への問い合わせが必要
XMV ERROR	ファイル、またはディレクトリのどこかに問題が発生している。このプロセスが実行される前に、宛先のモードがチェックされるべきであるが実行されていないなど、スプールディレクトリに問題がある可能性がある
CAN'T FORK	fork と exec を実行しようとしたが失敗した。現行ジョブは失われず、後で再試行される (uuxqt)。処置は不要

UUCP の STATUS エラーメッセージ

表 40–12 に一般的な STATUS エラーメッセージを示します。

表 **40–12** UUCP の STATUS エラーメッセージ

エラーメッセージ	説明または処置
OK	状態は良好
NO DEVICES AVAILABLE	現在、この呼び出し用に使用可能なデバイスがない。該当のシステムについて Devices ファイル内に有効なデバイスがあるかどうかを確認してください。そのシステムの呼び出しに使用するデバイスが Systems ファイル内にあるかどうかを検査してください
WRONG TIME TO CALL	Systems ファイルに指定されている日時以外の時点で、システムに対する呼び出しが行われた
TALKING	会話中
LOGIN FAILED	特定のマシンのログインが失敗した。ログインまたはパスワードが正しくないか、番号が正しくないか、低速のマシンであるか、Dialer-Token-Pairs スクリプトによる処理が失敗した
CONVERSATION FAILED	起動に成功した後で対話が失敗した。一方の側がダウンしたか、プログラムが異常終了したか、回線 (リンク) が切断されたことが考えられる
DIAL FAILED	リモートマシンがまったく応答しない。ダイアラが不良であるか、電話番号が正しくない可能性がある
BAD LOGIN/MACHINE COMBINATION	あるマシンが、Permissions ファイルの条件を満たしていないログインとマシン名を使用して、ローカルマシンを呼び出そうとした。偽装の疑いがある
DEVICE LOCKED	使用しようとしている呼び出しデバイスは、現在ロックされ、他のプロセスに使用されている
ASSERT ERROR	ASSERT エラーが発生した。/var/uucp/.Admin/errors ファイルにエラーメッセージが入っているかどうかを検査し、631 ページの「UUCP の ASSERT エラーメッセージ」を参照
SYSTEM NOT IN Systems FILE	システムが Systems ファイルの中に記述されていない
CAN'T ACCESS DEVICE	アクセスしようとしたデバイスが存在しないか、またはモードが正しくない。Systems ファイルと Devices ファイルの中の該当のエントリを検査する
DEVICE FAILED	デバイスがオープンできない
WRONG MACHINE NAME	呼び出されたマシンは、予期したのとは異なる名前である
CALLBACK REQUIRED	呼び出されたマシンは、そのマシンがローカルマシンをコールバックする必要があることを示している

表 **40–12** UUCP の STATUS エラーメッセージ　(続き)

エラーメッセージ	説明または処置
REMOTE HAS A LCK FILE FOR ME	リモートマシンは、ローカルマシンに関連する LCK ファイルを持っている。そのリモートマシンがローカルマシンを呼び出そうとしている可能性がある。そのマシンの UUCP のバージョンが古い場合は、プロセスがローカルマシンに接続しようとして失敗し、LCK ファイルがそのまま残されたことが考えられる。リモートマシンの UUCP のバージョンが新しく、ローカルマシンと通信していない場合は、LCK を持っているプロセスはハングする
REMOTE DOES NOT KNOW ME	リモートマシンの Systems ファイルの中に、ローカルマシンのノード名がない
REMOTE REJECT AFTER LOGIN	ローカルマシンがログインのために使用したログインが、リモートマシンが予期している内容に一致していない
REMOTE REJECT, UNKNOWN MESSAGE	理由は不明だが、リモートマシンがローカルマシンとの通信を拒否した。リモートマシンが標準バージョンの UUCP を使用していない可能性がある
STARTUP FAILED	ログインは成功したが、初期ハンドシェークに失敗した
CALLER SCRIPT FAILED	通常、これは DIAL FAILED と同じ。しかしこのエラーが頻発する場合は、Dialers ファイル内の呼び出し側スクリプトに原因があることが考えられる。Uutry を使用して検査する

UUCP の数値エラーメッセージ

表 40–13 に、/usr/include/sysexits.h ファイルにより生成されるエラー状態メッセージの終了コード番号を示します。これらのすべてが現在 uucp で使用されているわけではありません。

表 **40–13** 番号による UUCP のエラーメッセージ

メッセージ番号	説明	意味
64	Base Value for Error Messages	エラーメッセージはこの番号から始まる
64	Command–Line Usage Error	コマンドの使い方に誤りがある。たとえば、引数の数が正しくない、誤ったフラグ、誤った構文など
65	Data Format Error	入力データになんらかの誤りがある。このデータ形式はユーザーデータだけに使用されるもので、システムファイルには使用されない
66	Cannot Open Input	入力ファイル (システムファイルでない) が存在しないか、または読み取れない。これには、メールプログラムに対する「No message」のようなエラーも含まれる
67	Address Unknown	指定されたユーザーが存在しない。このエラーは、メールアドレスやリモートログインに使用される

表 **40–13** 番号による UUCP のエラーメッセージ　*(続き)*

メッセージ番号	説明	意味
68	Host Name Unknown	ホストが存在しない。このエラーは、メールアドレスやネットワーク要求に使用される
69	Service Unavailable	サービスが使用不可。このエラーは、サポートプログラムまたはファイルが存在しない場合に起こることがある。このメッセージは、何かが正常に働かずその理由が分からない場合の包括的なメッセージでもある
70	Internal Software Error	内部ソフトウェアエラーが検出された。このエラーは、できるだけオペレーティングシステム関係以外のエラーに限定されるべきである
71	System Error	オペレーティングシステムエラーが検出された。このエラーは、「フォーク不可」や「パイプ作成不可」などの状態を示す。たとえば、`getuid` が `passwd` ファイル内に存在しないユーザーを戻した場合などが含まれる
72	Critical OS File Missing	`/etc/passwd` や `/var/admin/utmpx` などのシステムファイルのどれかが存在しないか、開くことができない。あるいは、構文エラーなどがある
73	Can't Create Output File	ユーザーが指定した出力ファイルが作成できない
74	Input/Output Error	あるファイルについて入出力を行なっているときにエラーが起こった
75	Temporary Failure. User is invited to retry	一時的な障害。実際のエラーではない何かを示す。たとえば `sendmail` では、これは、メールプログラムが接続を確立できなかったため、後で要求を再試行する必要があることなどを意味する
76	Remote Error in Protocol	プロトコルの交換中に、リモートシステムが「使用不可」を示す何かを戻した
77	Permission Denied	この操作を行うための適正なアクセス権がユーザーにない。これはファイルシステムの問題を示すものではなく (その場合は `NOINPUT` や `CANTCREAT` などが使用される)、より高いレベルのアクセス権が必要であることを意味する。たとえば、`kre` は、メールを送ることのできる学生を制限するためにこのメッセージを使用する
78	Configuration Error	システムの構成にエラーがある
79	Entry Not Found	エントリが見つからない
79	Maximum Listed Value	エラーメッセージの最大番号

第 41 章

リモートシステムの利用 (トピック)

次の各章で、Solaris 環境で FTP サーバーを管理し、リモートシステムにアクセスする方法について説明します。次の章が含まれます。

第 43 章	FTP サーバーを管理する詳細手順
第 44 章	リモートファイルにアクセスする詳細手順

第 42 章

リモートシステムの利用 (概要)

ここでは、リモートファイルの利用について説明します。

FTP サーバーとは

FTP サーバーは wu-ftpd に基づいています。 ワシントン大学 (セントルイス) で開発された wu-ftpd は、インターネット上での大量データの配布に幅広く使用され、大規模な FTP サイトではよく使われる規格です。ライセンス条項については、`/var/sadm/pkg/SUNWftpu/install/copyright` から利用できるドキュメントを参照してください。

リモートシステムとは

この章では、リモートシステムとは、物理ネットワークによってローカルシステムに接続され、TCP/IP 通信用に構成されたワークステーションまたはサーバーであると想定します。

Solaris 9 リリースのシステム上では、TCP/IP は起動時に自動的に構成されます。詳細については、『*Solaris* のシステム管理 *(IP* サービス*)*』を参照してください。

Solaris 9 の FTP サーバーの新機能

Solaris Solaris 9 の FTP サーバーは、Solaris 8 の FTP ソフトウェアとの互換性を保ちながら、新しい機能を提供してパフォーマンスの向上を図っています。

表 **42–1** Solaris 9 の FTP サーバーの新機能

機能	説明	参照先
タイプと場所によるユーザーの分類	タイプとアドレスに基づいて、ユーザーのクラスを定義できる	645 ページの「FTP サーバークラスの定義方法」
クラスごとの制限	`ftpaccess` ファイルに設定されている制限に基づいて、同時にログインできる特定クラスのユーザー数を制御する	646 ページの「ユーザーログインの制限を設定する方法」
システム全体およびディレクトリ関連のメッセージ	特定のイベントに対して指定されるメッセージを表示する	654 ページの「ユーザーに送信するメッセージの作成方法」
ディレクトリごとのアップロード権	ファイルおよびディレクトリの作成やアクセス権など、FTP サーバーへのアップロードを制御できる	658 ページの「FTP サーバーへのアップロードの制御方法」
ファイル名フィルタ	アップロードしたファイルの名前に使用できる文字とその順序を指定できる	658 ページの「FTP サーバーへのアップロードの制御方法」
仮想ホストのサポート	単一のマシンで複数のドメインをサポートするように FTP サーバーを構成できる	663 ページの「完全仮想ホスティングを有効にする方法」
コマンドのログ	実ユーザー、ゲストユーザー、匿名ユーザーの各 FTP ユーザーが実行したコマンドのログを記録できる	669 ページの「FTP ユーザーにより実行されたコマンドの検査」
転送処理のログ	実ユーザー、ゲストユーザー、匿名ユーザーの各 FTP ユーザーが実行した転送処理のログを記録できる	`ftpaccess(4)`、`xferlog (4)`、`in.ftpd(1M)` のマニュアルページ
必要時の圧縮とアーカイブ	必要に応じて、`ftpconversions` ファイルに指定されている変換方法で圧縮およびアーカイブができる	`ftpconversions(4)`、`ftpaccess (4)` のマニュアルページ

注 – Solaris 9 リリースでは、Solaris 8 の `/etc/default/ftpd` はサポートされていません。 Solaris 8 から Solaris 9 へのアップグレード中に、`BANNER` および `UMASK` の各エントリは、`wu-ftpd` 用の対応するエントリに変換されます。 ただし、いくつかの `BANNER` 行は、`ftpaccess` のメッセージ機能に合わせて手作業で変換する必要があります。 詳細は、`ftpaccess(4)` のマニュアルページを参照してください。

注 – Solaris 8 の FTP サーバーで提供されていたサブログイン機能は、Solaris 9 の FTP サーバーではサポートされていません。

第 43 章

FTP サーバーの管理 (手順)

この章では、次の表に示す FTP サーバーを設定し、管理するための作業について説明します。

表 **43–1** FTP サーバーの管理 (作業マップ)

作業	説明	参照先
FTP サーバーへのアクセスの構成	/etc/ftpd ディレクトリに置かれた ftpaccess 、ftpusers、ftphosts の各ファイルを使用して、FTP サーバーへのアクセスを確立または制限する	646 ページの「ユーザーログインの制限を設定する方法」 647 ページの「無効なログインの試行回数を制御する方法」 648 ページの「特定のユーザーの FTP サーバーへのアクセスを拒否する方法」 649 ページの「デフォルト FTP サーバーへのアクセスを制限する方法」 645 ページの「FTP サーバークラスの定義方法」
FTP サーバーのログインの設定	実ユーザー、ゲストユーザー、匿名ユーザーのログインアカウントを設定する	650 ページの「実 FTP ユーザーの設定方法」 651 ページの「ゲスト FTP ユーザーの設定方法」 652 ページの「匿名 FTP ユーザーの設定方法」 653 ページの「/etc/shells ファイルの作成方法」
メッセージファイルをカスタマイズする	/etc/ftpd/ftpaccess ファイルを編集して、特定のイベントに関連して FTP サーバーが FTP クライアントにメッセージを返すように構成する	654 ページの「メッセージファイルのカスタマイズ方法」 654 ページの「ユーザーに送信するメッセージの作成方法」 655 ページの「README オプションの構成方法」

表 **43–1** FTP サーバーの管理 (作業マップ)　(続き)

作業	説明	参照先
FTP サーバー上のファイルへのアクセスを構成する	/etc/ftpd/ftpaccess ファイルを使用して、特定のコマンドの実行、FTP サーバーからのファイルのダウンロード、FTP サーバーへのファイルのアップロードを許可するユーザーのクラスを指定する	274 ページの「ダイアルアップネットワークに対する DA 検出の構成方法」 658 ページの「FTP サーバー上のアップロードとダウンロードの制御」
限定された仮想ホスティングまたは完全な仮想ホスティングを有効化する	/etc/ftpd/ftpaccess ファイルを使用して、FTP サーバーが同一マシン上の複数ドメインをサポートするように構成する	662 ページの「限定仮想ホスティングを有効にする方法」 663 ページの「完全仮想ホスティングを有効にする方法」
FTP サーバーを起動する	nowait モードまたはスタンドアロンモードで FTP サーバーを起動するように、/etc/inet/inetd.conf ファイルを編集する	665 ページの「inetd.conf を使用して FTP サーバーを起動する方法」 666 ページの「スタンドアロン FTP サーバーの起動方法」
FTP サーバーを停止する	/etc/ftpd/ftpaccess ファイルを使用し、ftpshut を実行して FTP サーバーを停止する	666 ページの「FTP サーバーの停止」
一般的な FTP サーバーの問題を障害追跡する	syslogd を検査し、greeting text と log commands を使用して FTP サーバー上の問題をデバッグする	668 ページの「syslogd 内の FTP サーバーのメッセージを検査する方法」 668 ページの「greeting text を使用して ftpaccess を検査する方法」 669 ページの「FTP ユーザーにより実行されたコマンドの検査」

FTP サーバーへのアクセスの制御

/etc/ftpd ディレクトリに置かれた構成ファイルを使用して FTP サーバーへのアクセスを制御します。次に構成ファイルを示します。

- ftpusers には、FTP サーバーへのアクセスを拒否するユーザーが列挙されています。
- ftphosts は、複数のホストから FTP サーバー上の複数のアカウントへのログインを許可または拒否するために使用します。
- ftpaccess は、メインの FTP 構成ファイルです。-a オプション付きで呼び出された場合、FTP サーバーは /etc/ftpd/ftpaccess ファイルだけを読み取ります。ftpaccess を使用する場合、すべてのユーザーは FTP サーバーへのアクセス

を許可されたクラスのメンバーである必要があります。特定のクラスにのみ適用される ftpaccess 指令を複数指定することができます。

詳細は、ftpusers(4)、ftphosts(4)、ftpaccess(4) のマニュアルページを参照してください。

注 – FTP サーバーのすべての構成ファイルで、# 記号で始まる行はコメントとして扱われます。

▼ FTP サーバークラスの定義方法

ftpaccess を使用する場合、FTP サーバーにログインするには、ユーザーはクラスのメンバーである必要があります。class 指令を ftpaccess に追加するには、特定のホストからアクセスを許可されているユーザーの *class* 名と *typelist* を指定します。

1. スーパーユーザーになります。
2. ftpaccess ファイルに匿名ユーザー、ゲストユーザー、実ユーザーのエントリを追加します。

```
class class typelist addrglob [addrglob...]
```

class	FTP ユーザーの定義に使用するキーワード
class	class キーワードを使用して定義する名前。各ログインは、定義されているクラスのリストと比較される。ログインしたユーザーは、一致した最初のクラスのメンバーと見なされる
typelist	3 種類のユーザー、anonymous、guest、real に一致するキーワードからなる、カンマで区切られたリスト
addrglob	展開されたドメイン名または展開された数値アドレス。*addrglob* は、スラッシュ ('/') で始まるファイル名にすることもできる。address:netmask または address/cidr という形のアドレス展開を追加できる 次に、展開されたアドレスの例を示す。 ■ 数値の IPv4 アドレス : **10.1.2.3** ■ 展開されたドメイン名 ***.provider.com** ■ 展開された数値の IPv4 アドレス **10.1.2.*** ■ 数値の IPv4 アドレス:ネットマスク **10.1.2.0:255.255.255.0** ■ 数値の IPv4 アドレス/CIDR **10.1.2.0/24** ■ 数値の IPv6 アドレス : **2000::56:789:21ff:fe8f:ba98** ■ 数値の IPv6 アドレス/CIDR: **2000::56:789:21ff:fe8f:ba98/120**

例 — FTP サーバークラスの定義

```
class  local  real,guest,anonymous *.provider.com
class  remote real,guest,anonymous *
```

この例では、local クラスを、*.provider.com からログインする real、guest、または anonymous のいずれかの種類のユーザーとして定義します。最後の行では、remote を、*.provider.com 以外からログインするユーザーとして定義します。

▼ ユーザーログインの制限を設定する方法

ftpaccess ファイルに設定された指令により、特定のクラスのユーザーが同時にログインできる数を制限できます。各ログインの制限には、クラス名、UUCP スタイルの曜日リスト、制限を超過した場合に表示するメッセージファイルが含まれます。

ユーザーログインの制限を設定するには、次の手順を実行します。

1. スーパーユーザーになります。
2. 次のエントリを ftpaccess ファイルに追加します。

```
limit class n times [message_file]
```

limit	定義されたクラスの特定時刻の同時ログイン数を、指定したユーザー数に制限するキーワード
class	class キーワードを使用して定義する名前。各ログインは、定義されているクラスのリストと比較される。ログインしたユーザーは、一致した最初のクラスのメンバーと見なされる
n	ユーザー数
times	クラスが接続可能な曜日と 1 日の時間帯。任意の曜日を指定する場合は Any を指定する
message_file	ユーザーがアクセスを拒否された場合に表示されるメッセージファイル

例 — ユーザーログインの制限の設定

```
limit   anon     50  Wk0800-1800        /etc/ftpd/ftpmsg.deny
limit   anon    100  Any                 /etc/ftpd/ftpmsg.deny
limit   guest   100  Any                 /etc/ftpd/ftpmsg.deny
```

前述の例の最初の行では、毎週勤務時間中のクラス anon のユーザーの同時ログイン数が 50 に制限されています。2 行目では、勤務時間外の anon のユーザーの同時ログイン数を 100 に制限しています。最後の行では、常時 guest ユーザーの同時ログイン数が 100 に制限されています。日時パラメータの指定方法の詳細は、ftpaccess(4) のマニュアルページを参照してください。

前述の例では、そのほかに、指定したログイン制限数に達した場合に /etc/ftpd/ftpmsg.deny ファイルの内容が返されることを示しています。この場合、ftpmsg.deny は存在するものと仮定しています。/usr/sbin/ftpcount コマンドを使用して、特定の時刻にログインしている各クラスのユーザーの数とログイン制限を表示する方法については、ftpcount(1) のマニュアルページを参照してください。

ユーザーは、その時刻の指定ログイン制限数に達していなければ、FTP サーバーへのログインを許可されます。匿名ユーザーは、ユーザー ftp としてログインします。実ユーザーは、自分自身としてログインします。ゲストユーザーは、アクセス特権を制限する chroot 環境を持つ実ユーザーとしてログインします。

/usr/sbin/ftpwho コマンドを使用して、FTP サーバーにログインするユーザーの識別情報を検査する方法については、ftpwho(1) のマニュアルページを参照してください。

▼ 無効なログインの試行回数を制御する方法

必要な情報を誤入力するなどの理由で FTP サーバーへのログインが失敗すると、通常はログインが繰り返されます。ユーザーは、特定回数連続してログインを試行できます。その回数を超えるとメッセージが syslog ファイルに記録されます。その時点でユーザーとの接続は切断されます。ログイン失敗時の試行回数を制限するには、次の手順を実行します。

1. スーパーユーザーになります。
2. 次のエントリを ftpaccess ファイルに追加します。

```
loginfails n
```

loginfails	FTP 接続が切断されるまでのログインに失敗できる回数を割り当てるキーワード
n	ログインに失敗できる回数

例 — 無効なログイン試行回数の制御

```
loginfails 10
```

この例では、ユーザーがログイン試行に 10 回失敗すると FTP サーバーから接続を切断されることを示します。

▼ 特定のユーザーの FTP サーバーへのアクセスを拒否する方法

/etc/ftpd/ftpusers には、FTP サーバーへのログインを拒否するユーザーの名前が列挙されています。ログインが試行されると、FTP サーバーは /etc/ftpd/ftpusers ファイルを検査して、そのユーザーからのアクセスを拒否するかどうかを判定します。ユーザーの名前がそのファイルにない場合は、次に /etc/ftpusers ファイルを検査します。

/etc/ftpusers の中にユーザー名に一致するものがあった場合、使用を差し控えるべきファイルで一致するユーザー名が見つかったことを示す syslogd メッセージが書き込まれます。また、このメッセージでは、/etc/ftpusers の代わりに /etc/ftpd/ftpusers を使用することを推薦します。

注 – /etc/ftpusers ファイルのサポートは、本リリースでは推奨されていません。FTP サーバーをインストールする時に /etc/ftpusers ファイルがすでに存在している場合は、/etc/ftpd/ftpusers に移動されます。

詳細は、syslogd(1M)、in.ftpd(1M)、および ftpusers(4) のマニュアルページを参照してください。

1. スーパーユーザーになります。
2. **FTP** サーバーへのログインを拒否するユーザーのエントリを /etc/ftpd/ftpusers に追加します。

例 — FTP サーバーへのアクセスを拒否する

```
root
daemon
bin
sys
adm
lp
uccp
nuucp
listen
nobody
noaccess
nobody4
```

この例では、ftpusers ファイルの通常のエントリが列挙されています。ユーザー名は /etc/passwd ファイルのエントリに一致します。通常このリストには、スーパーユーザー root、その他の管理に使用するユーザー、システムアプリケーションを示すユーザーが含まれます。

root エントリは、セキュリティ手段の 1 つとして ftpusers に追加されています。デフォルトのセキュリティポリシーでは、root のリモートログインを拒否します。また、/etc/default/loginfile ファイルの CONSOLE エントリとして設定されているデフォルト値も同じポリシーに従っています。login(1) を参照してください。

▼ デフォルト FTP サーバーへのアクセスを制限する方法

これまでに説明した制御方法以外に、ftpaccess ファイルに明示的に文を追加して FTP サーバーへのアクセスを制限することができます。

1. スーパーユーザーになります。
2. 次のエントリを ftpaccess ファイルに追加します。

   a. デフォルトでは、すべてのユーザーはデフォルト **(**非仮想**) FTP** サーバーへのアクセスを許可されています。特定のユーザー **(**anonymous 以外**)** のアクセスを拒否するには、次のエントリを追加します。

   ```
   defaultserver deny username [username...]
   ```

defaultserver	アクセスの拒否または許可を設定する非仮想サーバーの識別に使用するキーワード
username	defaultserver へのアクセスを制限するユーザーのログイン名

   b. deny 行に列挙されていないユーザーのアクセスを許可するには、次の行を追加します。

   ```
   defaultserver allow username [username...]
   ```

   c. 匿名ユーザーのアクセスを拒否するには、次のエントリを追加します。

   ```
   defaultserver private
   ```

例 — デフォルト FTP サーバーへのアクセスの制限

```
defaultserver deny *
defaultserver allow username
```

この例では、FTP サーバーは、anon ユーザーと allow 行に列挙されているユーザー以外のユーザーのアクセスをすべて拒否するように設定されています。

また、ftphosts ファイルを使用して、複数のホストからの特定のログインアカウントのアクセスを拒否することができます。その他の情報は、ftphosts(4) のマニュアルページを参照してください。

FTP サーバーのログインの設定

FTP サーバーにアクセスするには、まずログインする必要があります。FTP サーバーは、実ユーザー、ゲストユーザー、匿名ユーザーの 3 種類のユーザーログインアカウントをサポートします。

- 実ユーザーには、FTP サーバーが動作するシステム上の端末セッションの確立を許可するアカウントがあります。ディレクトリとファイルのアクセス権の制約は受けますが、実ユーザーはディスク構造全体を参照できます。
- ゲストユーザーも、FTP サーバーにログインするためのアカウントを必要とします。各ゲストアカウントは、ユーザー名とパスワードを使用して設定されます。端末セッションを確立できないように、ゲストには有効なログインシェルは割り当てません。ログイン時に、FTP サーバーは `chroot(2)` 操作を実行して、ゲストが参照できるサーバーのディスク構造を制限します。

注 – FTP サーバーへのアクセスを許可できるように、実ユーザーとゲストユーザーのログインシェルを `/etc/shells` ファイルに列挙する必要があります。

- 匿名ユーザーは、`ftp` または `anonymous` をユーザー名として使って FTP サーバーにログインします。規約では、匿名ユーザーはパスワードの代わりに電子メールアドレスを入力します。

  ログイン時に、FTP サーバーは `chroot(2)` 操作を実行して、匿名ユーザーが参照できるサーバーのディスク構造を制限します。ゲストユーザーにはそれぞれ独立した領域を作成できますが、匿名ユーザーは全員が単一のファイル領域を共有します。

  実ユーザーとゲストユーザーは、個別のアカウントとパスワードを使用してログインしますが、本人以外はその存在を知りません。匿名ユーザーは、誰でも知っているアカウントでログインします。可能性として、このアカウントは誰でも使用できます。大規模なファイル配布システムのほとんどは匿名アカウントを使用して作成します。

▼ 実 FTP ユーザーの設定方法

実ユーザーの FTP サーバーへのアクセスを有効にするには、次の手順を実行します。

1. ユーザーに、端末セッションの確立に使用可能なユーザー名とパスワードで設定されたアカウントがあることを確認します。

   詳細は、『*Solaris* のシステム管理 *(*基本編*)*』の「ユーザーアカウントとグループの管理 (概要)」を参照してください。

2. 実ユーザーが `ftpaccess` ファイルのクラスのメンバーであることを確認します。

ftpaccess に定義されたユーザークラスについては、645 ページの「FTP サーバークラスの定義方法」 を参照してください。

3. ユーザーのログインシェルが /etc/shells ファイルに列挙されていることを確認します。

▼ ゲスト FTP ユーザーの設定方法

ftpconfig スクリプトを使用して、すべての必要なシステムファイルをホームディレクトリにコピーします。ゲストユーザーとゲストのホームディレクトリがすでに存在する場合は、ftpconfig スクリプトは現在のシステムファイルでホームディレクトリ下の領域を更新します。

詳細は、ftpconfig(1M) のマニュアルページを参照してください。

注 – 匿名ユーザー用のユーザー名は anonymous か ftp ですが、FTP ゲスト用のユーザー名は固定されていません。実ユーザー名として使用できる名前であれば、どのような名前でも選択できます。

ゲストユーザーの FTP サーバーへのアクセスを有効にするには、次の手順を実行します。

1. useradd スクリプトを使用して、ログインシェルが /bin/true、ホームディレクトリが */root_dir/./home_dir* であるようなゲストユーザーアカウントを作成します。

   詳細は、useradd(1M) のマニュアルページと『*Solaris* のシステム管理 *(*基本編*)*』の「ユーザーアカウントとグループの管理 (概要)」を参照してください。

注 – この手順では、**/home/guests/./guest1** は guest1 というユーザーのホームディレクトリの名前として使用されます。

```
# /usr/sbin/useradd -m -c "Guest FTP" -d \
  /home/guests/./guest1 -s /bin/true guest1
```

2. ゲストアカウントにパスワードを割り当てます。

3. guestuser エントリを ftpaccess ファイルに追加します。

```
guestuser guest1
```

注 – さらに、ftpaccess ファイルで guestgroup 機能を使用して、複数のゲストユーザーを指定することができます。ftpaccess ファイルで guest-root 機能を使用すると、ゲストユーザーのホームディレクトリパスで /./ を指定する必要がなくなります。

4. ftpaccess ファイルで、ゲストユーザーが class のメンバーであることを確認します。詳細は、**645** ページの「**FTP** サーバークラスの定義方法」 を参照してください。

5. ftpconfig スクリプトを使用して、chroot 領域の必要なファイルを作成します。

   ```
   /usr/sbin/ftpconfig -d /home/guests
   ```

6. /bin/true が /etc/shells ファイルに列挙されていることを確認します。**653** ページの「/etc/shells ファイルの作成方法」 を参照してください。

例 — ゲスト FTP サーバーの設定

この例では、FTP 領域は /home/guests ディレクトリに設定されます。

```
# /usr/sbin/ftpconfig -d /home/guests
Updating directory /home/guests
```

▼ 匿名 FTP ユーザーの設定方法

ftpconfig スクリプトは、anonymous アカウントを作成し、ホームディレクトリに必要なファイルを作成します。

詳細は、ftpconfig(1M) のマニュアルページを参照してください。

匿名ユーザーの FTP サーバーへのアクセスを有効にするには、次の手順を実行します。

1. ftpconfig スクリプトを使用して、匿名ユーザーアカウントを作成します。

   ```
   # /usr/sbin/ftpconfig anonymous-ftp-directory
   ```

2. ftpaccess ファイルで、匿名ユーザーが class に割り当てられていることを確認します。
   詳細は、645 ページの「FTP サーバークラスの定義方法」 を参照してください。

例 — 匿名 FTP ユーザーの設定

この例では、FTP 領域は /home/ftp ディレクトリに設定されます。

```
# /usr/sbin/ftpconfig /home/ftp
Creating user ftp
Updating directory /home/ftp
```

▼ /etc/shells ファイルの作成方法

1. スーパーユーザーになります。
2. /etc/shells ファイルを作成します。
3. /etc/shells を編集します。各行のシェルに完全パスを追加します。

例 — /etc/shells ファイルの作成

次に、FTP ゲストユーザー用の /bin/true が含まれる /etc/shells ファイルの例を示します。

```
/sbin/sh
/bin/csh
/bin/jsh
/bin/ksh
/bin/remsh
/bin/rksh
/bin/rsh
/bin/sh
/usr/bin/csh
/usr/bin/ksh
/usr/bin/bash
/usr/bin/tcsh
/usr/bin/zsh
/bin/true
```

メッセージファイルのカスタマイズ

FTP サーバーを構成して、特定のイベントに関連するメッセージを FTP クライアントに返すことができます。ユーザーが FTP サーバーにログインするときに表示される開始メッセージを設定することができます。また、ユーザーが別のディレクトリに移動する場合にメッセージを表示することもできます。

メッセージファイルには、プレーンテキストだけでなく、1 つまたは複数のマジッククッキーを設定できます。マジッククッキーは、% (パーセント記号) と、その後に続く 1 文字から構成されます。クッキーをマジックテキストに組み込む場合、メッセージファイルが呼び出された時点でクッキーに関連付けられた情報が画面に表示されます。

たとえば、メッセージテキストにクッキー %L が含まれているとします。

```
Welcome to %L!
```

メッセージが表示されるとき、マジッククッキー %L は、ftpaccess ファイルの hostname 文で定義されたサーバー名で置き換えられます。サポートされているメッセージクッキーの完全なリストは、ftpaccess(4) のマニュアルページを参照してください。

注 – ホスト名が ftpaccess ファイルに定義されていない場合、ローカルマシンのデフォルトホスト名が使用されます。

▼ メッセージファイルのカスタマイズ方法

1. スーパーユーザーになります。
2. メッセージファイルを編集して、適宜マジッククッキーを追加します。
   使用できるクッキーのリストは、ftpaccess(4) のマニュアルページを参照してください。

例 — メッセージファイルのカスタマイズ

次に、マジッククッキーを使用するメッセージファイルの例を示します。

```
Welcome to %L -- local time is %T.

You are number %N out of a maximum of %M.
All transfers are logged.

If your FTP client crashes or hangs shortly after login
please try
using a dash (-) as the first character of your password.
This will
turn off the informational messages that may be confusing
your FTP
client.

Please send any comments to %E.
```

▼ ユーザーに送信するメッセージの作成方法

ユーザーがログインした後、システム関連またはアプリケーション関連のメッセージが画面に表示されます。ftpaccess ファイルには、関連付けられた message 文をトリガするイベントが列挙されています。

1. スーパーユーザーになります。
2. 次のエントリを ftpaccess ファイルに追加します。

```
message message_file [when [class ...]]
```

`message`	ユーザーがログインしたとき、または作業ディレクトリを変更するコマンドを実行したときに表示されるメッセージファイルの指定に使用するキーワード
message_file	表示するメッセージファイルの名前
when	`login` または `cwd=`*dir* と設定されるパラメータ。 例を参照すること
class	`class` を指定すると、メッセージの表示を特定のクラスのメンバーに限定できる

例 - ユーザーに送信するメッセージの作成

```
message    /etc/ftpd/Welcome    login   anon guest
message    .message    cwd=*
```

この例では、クラス `anon` または `guest` のユーザーがログインするときに `/etc/ftpd/Welcome` ファイルが表示されることを示します。2 行目では、すべてのユーザーに対して現在の作業ディレクトリにある `.message` ファイルが表示されることを示します。

メッセージファイルは、ゲストユーザーおよび匿名ユーザーの `chroot` ディレクトリからの相対位置に作成します。

▼ README オプションの構成方法

ディレクトリに最初に移動したとき、`README` ファイルを表示することができます。README オプションを構成するには、次のエントリを `ftpaccess` ファイルに追加します。

1. スーパーユーザーになります。
2. 次のエントリを `ftpaccess` ファイルに追加します。

```
readme message_file [when [class...]]
```

`readme`	ユーザーがログインするか、作業ディレクトリを変更するときに確認するメッセージファイルの指定に使用されるキーワードメッセージファイルが存在する場合、ユーザーはその存在を示す通知と、ファイルの更新日付を受け取る

message_file	確認するメッセージファイルの名前
when	`login` または `cwd=`*dir* と設定されるパラメータ。 例を参照すること
class	`class` を指定すると、メッセージの表示を特定のクラスのメンバーに限定できる

注 – `greeting` キーワードと `banner` キーワードを使用して、ユーザーにメッセージを送信することもできます。`ftpaccess(4)` のマニュアルページを参照してください。

例 — README オプションの構成

```
readme  README*     login
readme  README*     cwd=*
```

この例では、ログイン時、またはディレクトリ変更時に、`README*` に一致するファイルをすべて表示することを示します。この例で使用されている設定に基づいたログイン例を次に示します。

```
% ftp earth
Connected to earth.
220 earth FTP server ready.
Name (earth:rimmer): ftp
331 Guest login ok, send your complete e-mail address as password.
Password:
230-
230-Welcome to earth -- local time is Thu Jul 15 16:13:24
1999.
230-
230-You are number 1 out of a maximum of 10.
230-All transfers are logged.
230-
230-If your FTP client crashes or hangs shortly after login
please try
230-using a dash (-) as the first character of your
password.  This will
230-turn off the informational messages that may be
confusing your FTP
230-client.
230-
230-Please send any comments to ftpadmin@earth.
230-
230 Guest login ok, access restrictions apply.
ftp> cd pub
250-Please read the file README
250-  it was last modified on Thu Jul 15 16:12:25 1999 - 0
days ago
```

```
250 CWD command successful.
ftp> get README /tmp/README
200 PORT command successful.
150 Opening ASCII mode data connection for README (0
bytes).
226 ASCII Transfer complete.
ftp> quit
221 Goodbye.
```

FTP サーバー上のファイルへのアクセスの制御

ここで説明する FTP サーバーのアクセス制御は、Solaris 9 オペレーティング環境で使用できる標準のファイルとディレクトのアクセス制御を補足するものです。Solaris の標準コマンドを使用して、ファイルへのアクセス、ファイルの変更、またはファイルのアップロードが可能なユーザーを制限します。chmod(1)、chown(1)、chgrp(1) のマニュアルページを参照してください。

▼ ファイルアクセスコマンドの制御方法

ftpaccess ファイル内のアクセス権機能を使用してどの種類のユーザーにどのコマンドの実行を許可するかを指定するには、次の手順を実行します。

1. スーパーユーザーになります。
2. 次のエントリを ftpaccess ファイルに追加します。

   command yes|no *typelist*

command	chmod、delete、overwrite、rename、または umask のいずれか
yes|no	ユーザーにコマンドの発行を許可または拒否する
typelist	"anonymous"、"guest"、"real" のキーワードを任意に組み合わせてカンマで区切って並べたリスト

例 — ファイルアクセスコマンドの制御

次に、FTP サーバー上のファイルアクセス機能に対して設定されているアクセス権の例を示します。

```
chmod no anonymous, guest
delete    no anonymous
overwrite no anonymous
rename    no anonymous
umask     no guest, anonymous
```

この例では、次のことが示されています。

- 匿名ユーザーは、ファイルの削除、上書き、名前変更を行うことができない
- ゲストユーザーと匿名ユーザーは両方とも、アクセスモードの変更、umask のリセットができない

FTP サーバー上のアップロードとダウンロードの制御

FTP サーバー上のディレクトリのアクセス権を設定することにより、FTP サーバーへのアップロードと FTP サーバーからのダウンロードを制御できます。デフォルトでは、匿名ユーザーはアップロードを実行できません。匿名ユーザーにアップロードを許可する場合は、充分な注意が必要です。

▼ FTP サーバーへのアップロードの制御方法

`ftpaccess` ファイルに指令を追加して、アップロードのアクセス権と、アップロード異常終了時に表示するエラーメッセージを指定します。

1. スーパーユーザーになります。
2. 次のエントリを `ftpaccess` ファイルに追加します。
   ユーザーにファイルのアップロードを許可するには、次のエントリを追加します。

```
upload [absolute|relative] [class=<classname>]... [-] root-dir \
dirglob yes|no owner group mode [dirs|nodirs] [<d_mode>]

path-filter typelist mesg allowed_charset {disallowed regexp...}
```

`upload`	ホームディレクトリ (`chroot()` の引数) として *root-dir* を持つユーザーに適用するキーワード。*root-dir* に "*" を指定して、任意のホームディレクトリを一致させることができる

`absolute|relative`	*root-dir* ディレクトリパスを、絶対パスとして解釈するか、または現在の `chroot` ディレクトリからの相対パスとして解釈するかを指定するパラメータ
`class`	任意個数の `class=<classname>` 制限の指定に使用するキーワード。制限が指定された場合、upload 節が有効になるのは、現在のユーザーが指定されたクラスのメンバーである場合に限定される
root-dir	ユーザーのルートディレクトリと匿名ユーザーのホームディレクトリ
dirglob	ディレクトリ名に一致するパターン。アスタリスクを任意の場所に使用することも、単独で使用して任意のディレクトリを表すこともできる
`yes|no`	FTP サーバーへのアップロードを許可または拒否する変数
owner	`dirnames` にアップロードされたファイルの所有者
group	`dirnames` にアップロードされたファイルに関連付けられているグループ
mode	アップロードされたファイルのアクセス権の指定に使用するパラメータ。デフォルトモード `0440` の場合、匿名アカウントのユーザーはアップロードされたファイルを読み取れない
`dirs|nodirs`	`dirnames` に列挙されたディレクトリに、ユーザーがサブディレクトリを作成することを許可または拒否するキーワード
`d_mode`	新しく作成したディレクトリのアクセス権を決定するオプションモード
`path-filter`	アップロードされたファイルの名前を制御するキーワード
typelist	"anonymous"、"guest"、"real" のキーワードを任意に組み合わせてカンマで区切って並べたリスト"
mesg	`regexp` 条件に一致しない場合に表示されるメッセージファイル
allowed_charset {disallowed regexp...}	ファイル名で使用できる、または使用できない、英数字

例 — FTP サーバーへのアップロードの制御

```
upload /export/home/ftp /incoming yes ftpadm ftpadmin 0440 nodirs
path-filter anonymous /etc/ftpd/filename.msg ^[-A-Za-z0-9._]*$ ^[.-]
```

この例では、次のことが示されています。

- /export/home/ftp への chroot を使用する FTP ユーザーアカウントは、/incoming ディレクトリにアップロードすることができる。アップロードされたファイルの所有者は、ユーザー ftpadm、グループ ftpadmin である。モードは nodirs キーワード付きで 0440 に設定され、匿名ユーザーによるサブディレクトリの作成を拒否する
- 匿名ユーザーの場合、ファイル名は A-Z、a-z、0-9、. (ピリオド)、- (ハイフン)、_ (下線) の任意の並びである。ファイル名を . (ピリオド) または - (ハイフン) で始めることはできない。ファイル名がこの条件を満足しない場合、/etc/ftpd/filename.msg メッセージファイルが FTP 管理者により作成済みであれば、そのファイルが表示される。このメッセージの後に、FTP サーバーのエラーメッセージが表示される

注 – 匿名ユーザーによるアップロードが許可されているディレクトリの所有者とアクセス権は、厳密に制御する必要があります。FTP 管理者は FTP サーバーにアップロードされるすべてのファイルの所有者である必要があります。匿名ユーザーにファイルのアップロードを許可する場合、FTP 管理者を作成する必要があります。ディレクトリの所有者はユーザー ftpadm、グループ ftpadm、アクセス権は 3773 である必要があります。

FTP サーバーにアップロードされるファイルのアクセスモードは 0440 である必要があります。モードを 0440 にすると、匿名アカウントのユーザーはアップロードされたファイルを読み取れません。この制限により、サーバーが第 3 者によってファイル配布の場所として使用されるのを防ぎます。

アップロードされたファイルを配布可能にするために、FTP 管理者はそれらのファイルを公共ディレクトリに移動することができます。

▼ FTP サーバーへのダウンロードの制御方法

1. スーパーユーザーになります。
2. 次のエントリを ftpaccess ファイルに追加して、ユーザーがファイルを読み取れないようにします。

   ```
   noretrieve [absolute|relative] [class=classname]... [-] filename ...
   ```

`noretrieve`	特定のファイル (複数可) の読み取りの拒否に使用するキーワード
`absolute|relative`	*root-dir* ディレクトリパスを、絶対パスとして解釈するか、または現在の `chroot` ディレクトリからの相対パスとして解釈するかを指定するパラメータ
`class`	`noretrieve` 制限を適用するユーザーの `class=<classname>` の指定に使用するキーワード
filename	ユーザーによる読み取りを拒否するファイルの名前

例 — FTP サーバーへのダウンロードの制御

```
noretrieve /etc/passwd
```

この例では、すべてのユーザーが `/etc/passwd` の読み取りを拒否されます。

仮想ホスティング

仮想ホスティングにより、FTP サーバーは同一マシン上の複数ドメインをサポートできます。各仮想ホストには、独立した論理インタフェースと IP アドレスが必要です。

FTP サーバーは、「限定」と「完全」の 2 種類の仮想ホスティングをサポートします。 限定仮想ホスティングでは、すべての仮想ホストが同じ構成ファイルを使用します。完全仮想ホスティングでは、各仮想ホストは個別の構成ファイルを使用できます。

注 – デフォルトでは、実ユーザーとゲストユーザーは、仮想ホストへのログインを拒否されます。次に示す `ftpaccess` 指令を設定すると、デフォルトを上書きできます。

特定のユーザーにアクセスを許可する場合
`virtual` *address* `allow` *username*
匿名ユーザーのアクセスを拒否する場合
`virtual` *address* `private` *username*

詳細は、`ftpaccess(4)` のマニュアルページを参照してください。

▼ 限定仮想ホスティングを有効にする方法

限定仮想ホスティングでは、仮想 FTP サーバーの部分的なサポートを提供します。限定仮想ホスティングのサポートを有効にするには、仮想ルートディレクトリを指定します。必要であれば、次に示す仮想ホストのパラメータを `ftpaccess` ファイルに設定することもできます。

- `banner`
- `logfile`
- `email`
- `hostname` `ftpaccess` ファイル内のすべての指令は、すべての仮想サーバーによりグローバルに共有されます。

1. スーパーユーザーになります。
2. 次のエントリを `ftpaccess` ファイルに追加します。

```
virtual address root|banner|logfile path
virtual address hostname|email string
```

`virtual`	仮想サーバー機能を有効にするために使用するキーワード
address	仮想サーバーの IP アドレス
`root`	仮想サーバーのルートディレクトリ
`banner`	仮想サーバーへの接続が確立したときに表示されるバナーファイル
`logfile`	仮想サーバーに対するファイル転送の記録
path	仮想サーバー上のディレクトリとファイルの位置の指定に使用する変数
`email`	メッセージファイルと `HELP` コマンドで使用される電子メールアドレス
`hostname`	グリーティングメッセージやステータスコマンドで表示されるホスト名
string	`email` パラメータまたは `hostname` パラメータの指定に使用する変数

注 – `hostname` を仮想サーバーの *address* として使用することは可能ですが、それよりも IPv4 アドレスの使用を強く推奨します。`hostname` に一致するホストが見つかるためには、FTP 接続を受信するときに DNS が使用可能である必要があります。IPv6 ホストの場合は、IPv6 アドレスよりもホスト名を使用します。

例 — 限定仮想ホスティングの有効化

```
virtual 10.1.2.3 root    /var/ftp/virtual/ftp-serv
virtual 10.1.2.3 banner  /var/ftp/virtual/ftp-serv/banner.msg
virtual 10.1.2.3 logfile /var/log/ftp/virtual/ftp-serv/xferlog
```

この例では、仮想 FTP サーバー上の `root` ディレクトリ、`banner`、`logfile` の位置を設定します。

注 – `ftpaddhost(1M)` スクリプトを `-l` オプション付きで使用して、限定仮想ホストを構成できます。

次の例では、`ftpaddhost` を `-l -b -x` オプションとともに実行して、テストバナーと、仮想ルート `/var/ftp/virtual/10.1.2.3` の下にあるログファイル `/var/ftp/virtual/10.1.2.3/xferlog` を使用する限定仮想ホスティングを構成します。

```
ftpaddhost -l -b -x /var/ftp/virtual/10.1.2.3/xferlog \
/var/ftp/virtual/10.1.2.3
```

▼ 完全仮想ホスティングを有効にする方法

完全仮想ホスティングでは、各仮想ドメインは個別の構成ファイルを使用できます。FTP サーバー上の仮想ホスティングの完全サポートを有効にするには、特定のドメインについて次に示す FTP 構成ファイルを作成または変更します。

- `ftpaccess`
- `ftpusers`
- `ftpgroups`
- `ftphosts`
- `ftpconversions`

詳細は、`ftpaccess(4)`、`ftpusers(4)`、`ftpgroups(4)`、`ftphosts(4)`、`ftpconversions(4)` のマニュアルページを参照してください。

注 – 構成ファイルの個別のバージョンが見つからない場合は、`/etc/ftpd` ディレクトリに置かれた構成ファイルのマスターバージョンを使用します。

---

1. スーパーユーザーになります。
2. 次のエントリを `/etc/ftpd/ftpservers` ファイルに追加します。

   address /config-file-dir

address	仮想サーバーの IP アドレス
config-file-dir	仮想ホスト用にカスタマイズされた構成ファイルが置かれるディレクトリ

注 – `hostname` を仮想サーバーの *address* として使用することは可能ですが、それよりも IPv4 アドレスの使用を強く推奨します。`hostname` に一致するホストが見つかるためには、FTP 接続を受信するときに DNS が使用可能である必要があります。IPv6 ホストの場合は、IPv6 アドレスよりもホスト名を使用します。

---

3. 仮想ホスト用にカスタマイズされた **FTP** サーバー構成ファイルを作成するには、`/etc/ftpd` ディレクトリにある構成ファイルのマスターバージョンを `/config-file-dir` ディレクトリにコピーします。

   詳細は、`ftpservers(4)` のマニュアルページを参照してください。

例 — 完全仮想ホスティングの有効化

```
#
# FTP Server virtual hosting configuration file
#

10.1.2.3 /net/inet/virtual/somedomain/
10.1.2.4 /net/inet/virtual/anotherdomain/
```

この例では、仮想サーバー上の 2 つの異なるドメインの IP アドレスを指定します。

注 – `ftpaddhost(1M)` スクリプトを `-c` オプション付きで使用して、完全仮想ホストを構成できます。

次の例では、`ftpaddhost` を `-l -b -x` オプションとともに実行して、テストバナーと、仮想ルート `/var/ftp/virtual/10.1.2.3` の下にあるログファイル `/var/ftp/virtual/10.1.2.3/xferlog` を使用する限定仮想ホスティングを構成します。

```
ftpaddhost -l -b -x /var/ftp/virtual/10.1.2.3/xferlog \
/var/ftp/virtual/10.1.2.3
```

FTP サーバーの自動起動

FTP サーバーを起動するには、次の 2 つの方法があります。

- nowait サーバー。inetd.conf ファイルから起動される
- スタンドアロンサーバー。コマンド行または起動スクリプトから起動される

inetd.conf を使用した FTP サーバーの起動

inetd.conf ファイルに nowait エントリを追加して FTP サーバーを起動することができます。サイトで処理する接続が多数になる場合、FTP デーモンをスタンドアロンモードで実行することもできます。詳細は、inetd.conf(4) のマニュアルページを参照してください。また、その他のコマンド行オプションについては、in.ftpd (1M) のマニュアルページを参照してください。

▼ inetd.conf を使用して FTP サーバーを起動する方法

1. スーパーユーザーになります。
2. nowait エントリを inetd.conf ファイルに追加します。

```
ftp  stream  tcp6  nowait  root  /usr/sbin/in.ftpd  in.ftpd -a
```

注 – -a オプションの指定により、ftpaccess ファイルの使用を有効にします。

---

3. inetd にシグナルを送信し、inetd.conf ファイルを再読み取りします。

```
# pkill -HUP inetd
```

スタンドアロン FTP サーバーの起動

FTP サーバーを、inetd.conf とは無関係に、スタンドアロンサーバーとして実行することもできます。

スタンドアロンサーバーの応答時間は常に可能な限り最短であり、FTP サービス専用の大規模サーバー向けです。スタンドアロンサーバーは一切再起動する必要がないので、専用サーバーに適した短い接続応答時間を実現します。スタンドアロンサーバーは、混雑していない時間帯も含めて、常に動作しており、永久に接続要求を待ちます。

▼ スタンドアロン FTP サーバーの起動方法

1. スーパーユーザーになります。
2. `inetd.conf` ファイルの **ftp** サービスの行の先頭に **#** 記号を挿入して、そのエントリをコメントアウトします。
3. `inetd` にシグナルを送信し、`inetd.conf` ファイルを再読み取りします。

   ```
   # pkill -HUP inetd
   ```

4. スタンドアロン **FTP** サーバーを起動します。

   ```
   # /usr/sbin/in.ftpd -a -S
   ```

   FTP サーバー起動スクリプトにこの行を追加します。システム起動スクリプトの作成については、『*Solaris* のシステム管理 (基本編)』の「実行制御スクリプト」を参照してください。

FTP サーバーの停止

`ftpshut(1M)` コマンドは、特定の時刻に FTP サーバーを停止します。

`ftpshut` を実行する場合、コマンド行オプションでシステム停止時刻を指定するファイルを作成します。この時刻になると、それ以上の新しい接続は受け付けられなくなり、既存の接続は切断されます。この停止時刻の情報に基づいて、サーバーが停止することがユーザーに通知されます。`ftpshut` により作成されるファイルの位置は、`ftpaccess` ファイルの `shutdown` 指令によって指定します。

▼ FTP サーバーの停止方法

`ftpshut` を実行し、`ftpaccess` ファイルに `shutdown` 指令を追加するには、次の手順を実行します。

1. スーパーユーザーになります。
2. 次のエントリを `ftpaccess` ファイルに追加します。

   `shutdown` *path*

`shutdown`	FTP サーバーの停止時刻が予定されているかどうかを定期的に検査するファイルへの *path* の指定に使用するキーワード
path	`ftpshut` コマンドが作成したファイルの位置

3. `ftpshut` コマンドを実行します。

```
ftpshut [ -V ] [ -l min] [ -d min] time [warning-message...]
```

`ftpshut`	FTP サーバーが停止することをユーザーに通知する手順を提供するコマンド
`-V`	著作権情報とバージョン情報を表示した後、接続を切断するように指定するオプション
`-l`	FTP サーバーへの新しい接続を拒否する時間の調整に使用されるフラグ
`-d`	FTP サーバーへの既存の接続を切断する時間の調整に使用されるフラグ
`time`	停止時刻として `now` を指定すると即時停止する。未来における停止時刻を指定するには、「*+number*」または「*HHMM*」のどちらかの形式で指定する
`[warning-message...]`	停止通知メッセージ

4. `ftprestart` コマンドを使用して、**FTP** サーバーを停止後に再起動します。

詳細は、`ftpshut(1M)`、`ftpaccess(4)`、`ftprestart(1M)` のマニュアルページを参照してください。

FTP サーバーのデバッグ

ここでは、FTP サーバーに関する問題をデバッグする方法についていくつか説明します。

▼ syslogd 内の FTP サーバーのメッセージを検査する方法

FTP サーバーは、/etc/syslog.conf ファイルでデーモンメッセージの出力先として指定された位置に、デバッグに役立つメッセージを書き込みます。FTP サーバーに問題が発生した場合、まずこのファイルで関連するメッセージを検査します。

FTP サーバーメッセージは、機能デーモンにより制御されます。FTP サーバーから /var/adm/message にメッセージを送信し、syslogd にその構成ファイルを再読み取りさせるには、次の手順を実行します。

1. 次のようなエントリを /etc/syslog.conf ファイルに追加します。

   ```
   daemon.info /var/adm/message
   ```

2. syslogd にシグナルを送信して、その構成ファイルを再読み取りさせます。

   ```
   # pkill -HUP syslogd
   ```

   この操作により、FTP サーバーから有益な情報を含むメッセージが /var/adm/messages に書き込まれます。

▼ greeting text を使用して ftpaccess を検査する方法

greeting text 機能を使用して、適切な内容の ftpaccess ファイルが使用されていることを検査するには、次の手順を実行します。

1. 次の指令を **ftpaccess** ファイルに追加します。

   ```
   greeting text message
   ```

2. **FTP** サーバーに接続します。

3. メッセージが表示されない場合、次の手順を実行します。

   a. ftpaccess ファイルが正しい位置に置かれていることを確認します。strings (1) コマンドを使用して、**FTP** サーバーバイナリからファイルの位置を取得します。

      ```
      # strings /usr/sbin/in.ftpd | grep "^/.*ftpaccess"
      ```

   b. 仮想ホスティングが構成されているかどうか ftpservers ファイルを検査します。

      詳細は、ftpaccess(4)、ftpservers(4)、strings(1)、syslog.conf (4)、pkill(1) のマニュアルページを参照してください。

▼ FTP ユーザーにより実行されたコマンドの検査

FTP ユーザーがどのコマンドを実行したかを確認するには、`ftpaccess` の `log commands` ログ機能を使用します。

1. 次の指令を `ftpaccess` ファイルに追加し、*typelist* で指定されたユーザーによるコマンドを個別に記録します。

   `log commands` *typelist*

2. `/etc/syslog.conf` で指定した場所に書き込まれたメッセージを検査します。

第 44 章

リモートシステムへのアクセス (手順)

本章では、リモートシステムにログインし、リモートシステムのファイルを操作するために必要なすべての作業について説明します。この章で説明する手順は次のとおりです。

この章では、次の表に示すリモートシステムにログインし、ファイルをコピーするための作業について説明します。

表 **44–1** リモートシステムへのアクセス (作業マップ)

作業	説明	参照先
リモートシステムにログインする (rlogin)	■ .rhosts ファイルを削除する ■ rlogin コマンドを使用してリモートシステムにアクセスする	677 ページの「.rhosts ファイルを検索して削除する方法」 678 ページの「リモートシステムが動作中かどうかを調べる方法」 678 ページの「リモートシステムにログインしているユーザーを検索する方法」 679 ページの「リモートシステムにログインする方法 (rlogin)」 680 ページの「リモートシステムからログアウトする方法 (exit)」
リモートシステムにログインする (ftp)	■ ftp 接続のオープンとクローズを行う ■ リモートファイルから、およびリモートシステムに、ファイルをコピーする	682 ページの「ftp によりリモートシステムへ接続する方法」 682 ページの「リモートシステムとの ftp 接続を終了する方法」 683 ページの「リモートシステムからファイルをコピーする方法 (ftp)」 685 ページの「ファイルをリモートシステムにコピーする方法 (ftp)」
rcp を使用してリモートファイルをコピーする	rcp コマンドを使用して、リモートシステムから、およびリモートシステムに、ファイルをコピーする	689 ページの「ローカルシステムとリモートシステム間でファイルをコピーする方法 (rcp)」

リモートシステムへのログイン (rlogin)

rlogin コマンドを使用すると、リモートシステムにログインできます。ログインした後は、リモートファイルシステム内で移動し、その内容を (リモートシステムによる承認にしたがって) 操作したり、ファイルをコピーしたり、リモートコマンドを実行したりできます。

ログイン先のシステムがリモートドメインに所属している場合は、システム名にドメイン名を追加してください。次の例では、SOLAR はリモートドメイン名です。

```
rlogin pluto.SOLAR
```

また、Control-d と入力すると、リモートログイン処理をいつでも中断できます。

リモートログイン (rlogin) の認証

rlogin 処理の認証 (ログインするユーザーの確認処理) は、リモートシステムまたはネットワーク環境で実行されます。

この 2 つの認証形式の主な違いは、要求される対話操作と、認証の確立方法にあります。リモートシステムがユーザーを認証しようとする場合に、/etc/hosts.equiv または .rhosts ファイルを設定していなければ、パスワードの入力を促すプロンプトが表示されます。ネットワークがユーザーを認証しようとする場合は、ユーザーはすでにネットワークに認識されているので、パスワードプロンプトは表示されません。

リモートシステムがユーザーを認証しようとする場合は、特に次のいずれかに該当する場合は、リモートシステム上のローカルファイル内の情報を使用した認証が行われます。

- ユーザーが属するシステム名とユーザー名がリモートシステム上の /etc/hosts.equiv ファイルに列挙されている

  または

- システム名とユーザー名が、リモートユーザーのホームディレクトリの下にある .rhosts ファイルに入っている場合

ネットワークによる認証は、次のどちらかの場合に利用されます。

- ローカルネットワーク情報サービスとオートマウンタを使用して設定された「信頼できるネットワーク環境」がある場合
- リモートシステムの /etc/nsswitch.conf ファイルが指定するネットワーク情報サービスがユーザーに関する情報を持っている場合

注 – 通常は、ネットワークによる認証がシステムによる認証より優先されます。

/etc/hosts.equiv ファイル

/etc/hosts.equiv ファイルには、リモートシステムの「信頼される (trusted) ホスト」が 1 行に 1 つずつ入っています。ユーザーがこのファイルに含まれるホストから (rlogin を使用して) リモートログインしようとする場合、リモートシステムがそのユーザーのパスワードエントリにアクセスできれば、ユーザーはパスワードを入力しなくてもログインできます。

典型的な hosts.equiv ファイルの構造は次のとおりです。

```
host1
host2 user_a
+@group1
-@group2
```

上記の host1 のように、ホスト名だけのエントリであれば、そのホストが信頼されているため、そのマシン上のユーザーも信頼できることを意味します。

この例の第 2 のエントリのようにユーザー名も含まれていると、その指定されたユーザーがアクセスしようとする場合にのみ、そのホストが信頼されます。

グループ名の先頭にプラス記号 (+) が付いている場合は、そのネットグループ内のすべてのマシンが信頼されていることを意味します。

グループ名の先頭にマイナス記号 (–) が付いている場合は、そのネットグループ内には信頼できるマシンがないことを意味します。

/etc/hosts.equiv ファイルを使用する場合のセキュリティの問題

/etc/hosts.equiv ファイルにはセキュリティ上の問題があります。/etc/hosts.equiv ファイルをシステム上で管理する場合は、ネットワーク内で信頼されるホストのみを含めるようにしてください。別のネットワークに所属するホストまたは公共領域にあるマシンを追加しないでください。たとえば、端末室に置かれているホストは追加しないでください。

信頼できないホストを使用すると、重大なセキュリティ上の問題が発生する可能性があります。/etc/hosts.equiv ファイルを正しく構成されたファイルと置き換えるか、ファイルを削除してください。

/etc/hosts.equiv ファイルに + のみの 1 行しか入っていない場合は、認識されているすべてのホストが信頼されることを示します。

.rhosts ファイル

.rhosts ファイルは、/etc/hosts.equiv ファイルに対応するユーザー用のファイルです。このファイルには、通常、ホストとユーザーの組み合わせのリストが入っています。このファイルにホストとユーザーの組み合わせが含まれている場合、そのユーザーには、パスワードを入力しなくても、そのホストからリモートログインする許可が与えられます。

.rhosts ファイルはユーザーのホームディレクトリの一番上のレベルに置かれていなければなりません。サブディレクトリに置かれている .rhosts ファイルは参照されません。

ユーザーは、各自のホームディレクトリ内で .rhosts ファイルを作成できます。.rhosts ファイルを使用することによって、/etc/hosts.equiv ファイルを使用しなくても、異なるシステムのユーザー自身のアカウント間で信頼できるアクセスを行うことができます。

.rhosts ファイルを使用する場合のセキュリティの問題

.rhosts ファイルにはセキュリティ上、重大な問題があります。/etc/hosts.equiv ファイルはシステム管理者の制御下にあり、効率よく管理できますが、誰でも .rhosts ファイルを作成して、システム管理者が知らないうちに自分が選んだユーザーにアクセス権を与えることができます。

すべてのユーザーのホームディレクトリが 1 台のサーバー上にあって、特定のユーザーだけがそのサーバーに対してスーパーユーザーのアクセス権を持っている場合、ユーザーが .rhosts ファイルを使用できないようにするためには、スーパーユーザーとして、空の .rhosts ファイルを各ユーザーのホームディレクトリに作成します。次に、このファイルのアクセス権を 000 に変更します。こうしておけば、スーパーユーザーでも、そのファイルを変更することが難しくなります。これにより、ユーザーが .rhosts を無責任に使用することによって生じるセキュリティ問題を防ぐことができます。ただし、ユーザーが自分のホームディレクトリへの実効パスを変更できる場合、この方法は何の解決にもなりません。

.rhosts ファイルを確実に管理する唯一の方法は、それを完全に使用できないようにすることです。詳細は、677 ページの「.rhosts ファイルを検索して削除する方法」を参照してください。システム管理者は、システムを頻繁にチェックして、このポリシーに対する違反を調べることができます。このポリシーに対する例外は、root アカウントです。ネットワークのバックアップや他のリモートサービスを実行するには、.rhosts ファイルが必要な場合があります。

リモートログインのリンク

システムが正しく構成されていれば、リモートログインをリンクできます。たとえば、earth 上のユーザーが jupiter にログインし、そこから pluto にログインします。

このユーザーは jupiter からログアウトして pluto に直接ログインすることもできますが、このリンク方法の方が便利です。

パスワードを入力せずにリモートログインをリンクするには、/etc/hosts.equiv または .rhosts を正しく設定しておかなければなりません。

直接リモートログインと間接リモートログイン

rlogin コマンドにより、リモートシステムに直接的または間接的にログインできます。

直接リモートログインは、デフォルトユーザー名、すなわち現在ローカルシステムにログインしている個人のユーザー名を使用します。これは、最も一般的なリモートログイン形式です。

間接リモートログインは、リモートログイン処理中に別のユーザー名を入力することによって行います。これは、一時的に借りているワークステーションから行うタイプのリモートログインです。たとえば、ユーザーが同僚のオフィスにいるときに自分のホームディレクトリに置かれているファイルを確認する必要がある場合、同僚のシステムからリモートで自分のシステムにログインすることができます。この場合、自分のユーザー名を入力して間接リモートログインを実行することになります。

表 44–2 は、直接ログインや間接ログインと認証方式の依存関係を示しています。

表 **44–2** ログイン方式と認証方式 (rlogin) の依存関係

ログイン方式	ユーザー名の提供	認証	パスワード
直接	システム	ネットワーク	なし
		システム	必要
間接	ユーザー	ネットワーク	なし
		システム	必要

リモートログイン後の処理

リモートシステムにログインするときに、rlogin コマンドはホームディレクトリを見つけようとします。ホームディレクトリが見つからなければ、リモートシステムのルートディレクトリ (/) が割り当てられます。たとえば、次のように表示されます。

```
No directory! Logging in with home=/
```

ただし、 rlogin コマンドがホームディレクトリを見つけると、.cshrc ファイルと .login ファイルを生成します。したがって、リモートログイン後は、プロンプトが標準ログインプロンプトになり、現在のディレクトリはローカルにログインするときと同じになります。

たとえば、通常のプロンプトにシステム名と作業用ディレクトリが表示される場合と、ログイン時の作業用ディレクトリがホームディレクトリの場合、ログインプロンプトは次のようになります。

```
earth(/home/smith):
```

リモートシステムにログインすると、同じようなプロンプトが表示され、rlogin コマンドをどのディレクトリから入力したかに関係なく、作業用ディレクトリがホームディレクトリになります。

```
earth(/home/smith): rlogin pluto
.
.
.
pluto(/home/smith):
```

唯一の違いは、プロンプトの先頭にローカルシステムではなくリモートシステムの名前が表示されることです。リモートファイルシステムは、ホームディレクトリと並んで存在します。

/home ディレクトリに移動して ls を実行すると、次のように表示されます。

```
earth(home/smith): cd ..
earth(/home): ls
smith  jones
```

▼ .rhosts ファイルを検索して削除する方法

1. スーパーユーザーになります。
2. find(1) コマンドを使用し、.rhosts ファイルを検索して削除します。

```
# find home-directories -name .rhosts -print -exec rm{}
```

home-directories	ユーザーのホームディレクトリがあるディレクトリへのパス。複数のパスを指定すると、複数のホームディレクトリを一度に検索できる
-name .rhosts	ここでは .rhosts を指定する
-print	現在のパス名を出力する
-exec rm {} \;	指定したファイル名に一致するファイルすべてに、rm コマンドを適用するように find コマンドに伝える

find コマンドは、指定したディレクトリから始めて .rhosts というファイルを検索します。ファイルが見つかると、find はファイルのパスを画面上に表示し、ファイルを削除します。

例 — .rhosts ファイルを検索して削除する

次の例では、/export/home ディレクトリ内で、すべてのユーザーのホームディレクトリ内の .rhosts ファイルを検索し削除します。

```
# find /export/home -name .rhosts -print | xargs -i -t rm{}
```

▼ リモートシステムが動作中かどうかを調べる方法

ping コマンドを使用して、リモートシステムが動作中かどうかを調べます。

```
$ ping system-name | ip-address
```

system-name	リモートシステム名
ip-address	リモートシステムの IP アドレス

ping コマンドは、次の 3 つのメッセージのどれかを返します。

状態メッセージ	意味
system-name is alive	このシステムにはネットワーク経由でアクセスできる
ping: unknown host *system-name*	未知のシステム名
ping:no answer from *system-name*	システムは認識されるが、現在は動作していない

ping を実行した対象のシステムが別のドメイン内にある場合は、出力メッセージにルーティング情報も含まれることがありますが、これは無視してかまいません。

ping コマンドのタイムアウトは 20 秒です。つまり、20 秒以内に応答がなければ、第 3 のメッセージを返します。*time-out* 値を秒単位で入力すると、ping の待ち時間を増減させることができます。

```
$ ping system-name | ip-address time-out
```

詳細は、ping(1M) のマニュアルページを参照してください。

▼ リモートシステムにログインしているユーザーを検索する方法

rusers(1) コマンドを使用して、リモートシステムにログインしているユーザーを検索します。

```
$ rusers [-l] remote-system-name
```

rusers	(オプションなし) システム名と、root など現在ログインしているユーザー名を表示する

`-l`	ユーザーのログインウィンドウ、ログイン日時、ログインしている時間、ユーザーのログイン元のリモートシステム名など、各ユーザーの詳細な情報を表示する

例 — リモートシステムにログインしているユーザーを検索する

次の例は、`rusers` の短い形式の出力を示しています。

```
$ rusers pluto
pluto    smith  jones
```

次の例では、`rusers` の長い形式の出力は、2 人のユーザーがリモートシステム `starbug` にログインしていることを示します。第 1 のユーザーは 9 月 10 日にシステムコンソールからログインし、ログイン時間は 137 時間 15 分でした。第 2 のユーザーはリモートシステム `mars` から 9 月 14 日にログインしました。

```
$ rusers -l starbug
root         starbug:console           Sep 10 16:13  137:15
rimmer       starbug:pts/0             Sep 14 14:37         (mars)
```

▼ リモートシステムにログインする方法 (`rlogin`)

`rlogin(1)` コマンドを使用してリモートシステムにログインします。

```
$ rlogin [-l user-name] system-name
```

`rlogin`	(オプションなし) 現在のユーザー名を使用して、リモートシステムに直接ログインする
`-l` *user-name*	ユーザー名を入力して、リモートシステムに間接的にログインする

ネットワークがユーザーを認証しようとする場合には、パスワードを求めるプロンプトは表示されません。リモートシステムがユーザーを認証しようとする場合は、パスワードの入力を求めるプロンプトが表示されます。

処理が成功すると、`rlogin` コマンドは、そのシステムへの前回のリモートログイン、リモートシステム上で動作中のオペレーティングシステムのバージョン、ホームディレクトリに未処理のメールがあるかどうかに関して、簡潔な情報を表示します。

例 — リモートシステムにログインする (`rlogin`)

次の例は、`pluto` へ直接リモートログインした出力結果を示しています。このユーザーはネットワークから認証されています。

```
$ rlogin starbug
Last login: Mon Jul 12 09:28:39 from venus
Sun Microsystems Inc.   SunOS 5.8       February 2000
```

```
starbug:
```

次の例は、pluto へ間接リモートログインした出力結果を示しています。この場合、ユーザーはリモートシステムから認証されています。

```
$ rlogin -l smith pluto
password: user-password
Last login: Mon Jul 12 11:51:58 from venus
Sun Microsystems Inc.   SunOS 5.8       February 2000
starbug:
```

▼ リモートシステムからログアウトする方法 (exit)

exit(1) コマンドを使用して、リモートシステムからログアウトします。

```
$ exit
```

例 — リモートシステムからログアウトする (exit)

次の例は、ユーザー smith がシステム pluto からログアウトする様子を示しています。

```
$ exit
pluto% logout
Connection closed.
earth%
```

リモートシステムへのログイン (ftp)

ftp コマンドは、インターネットのファイルトランスポートプロトコルへのユーザーインタフェースを提供します。このユーザーインタフェースはコマンドインタプリタと呼ばれ、リモートシステムにログインし、そのファイルシステムについて様々な処理を実行できるようにします。基本操作については、表 44–3 を参照してください。

rlogin と rcp とで、ftp が優れている最大のポイントは、ftp はリモートシステムで UNIX を実行する必要がないことです。ただし、リモートシステムを TCP/IP 通信ができるように構成する必要はあります。逆に、rlogin の優れている点は、ftp よりも豊富なファイル操作コマンドを使用できることです。

リモートログインの認証 (ftp)

ftp によるリモートログインの認証は、次のいずれかの方法により確立できます。

- パスワードエントリをリモートシステムの /etc/passwd か、または同等のネットワーク情報サービスマップまたはテーブルに追加する
- リモートシステム上で匿名 ftp アカウントを確立する

重要な ftp コマンド

表 **44–3** 重要な ftp コマンド

コマンド名	説明
ftp	ftp コマンドインタプリタにアクセスする
ftp *remote-system*	リモートシステムへの ftp 接続を確立する。詳細は、682 ページの「ftp によりリモートシステムへ接続する方法」を参照
open	コマンドインタプリタからリモートシステムにログインする
close	リモートシステムからログアウトしてコマンドインタプリタに戻る
bye	ftp コマンドインタプリタを終了する
help	すべての ftp コマンドを表示するか、コマンド名が指定されている場合は、コマンドの機能に関する簡単な説明を表示する
reset	リモートの ftp サーバーとコマンド応答シーケンスの同期をとり直す
ls	リモートの作業用ディレクトリの内容を表示する
pwd	リモートの作業用ディレクトリ名を表示する
cd	リモートの作業用ディレクトリを変更する
lcd	ローカルの作業用ディレクトリを変更する
mkdir	リモートシステム上でディレクトリを作成する
rmdir	リモートシステム上でディレクトリを削除する
get、mget	リモートの作業用ディレクトリからローカルの作業用ディレクトリに 1 つ以上のファイルをコピーする
put、mput	ローカルの作業用ディレクトリからリモートの作業用ディレクトリに 1 つ以上のファイルをコピーする
delete、mdelete	リモートの作業用ディレクトリから 1 つ以上のファイルを削除する

詳細は、ftp(1) のマニュアルページを参照してください。

▼ ftp によりリモートシステムへ接続する方法

1. ftp 認証を持っていることを確認します。
   680 ページの「リモートログインの認証 (ftp)」で説明しているように、ftp 認証を持っている必要があります。
2. ftp コマンドを使用してリモートシステムへ接続します。

   ```
   $ ftp remote-system
   ```
   接続に成功すると、確認メッセージとプロンプトが表示されます。
3. ユーザー名を入力します。

   ```
   Name (remote-system:user-name): user-name
   ```
4. プロンプトが表示されたら、パスワードを入力します。

   ```
   331 Password required for user-name:
   Password: password
   ```
   アクセス中のシステムで匿名 ftp アカウントが設定されている場合は、パスワードとして電子メールアドレスの入力を求めるプロンプトが表示されます。ftp インタフェースがパスワードを受け付けると、確認メッセージと (ftp>) プロンプトを表示します。

   これで、help など、ftp インタフェースから提供されるどのコマンドでも使用できます。主なコマンドについては、表 44–3 を参照してください。

例 — ftp によりリモートシステムへ接続する

次の ftp セッションは、リモートシステム pluto 上でユーザー smith によって確立されました。

```
$ ftp pluto
Connected to pluto.
220 pluto FTP server ready.
Name (pluto:smith): smith
331 Password required for smith:
Password: password
230 User smith logged in.
ftp>
```

▼ リモートシステムとの ftp 接続を終了する方法

bye コマンドを使用して、リモートシステムとの ftp 接続を終了します。

```
ftp> bye
221-You have  transferred 0 bytes in 0 files.
221-Total traffic for this sessions was 172 bytes in 0 transfers.
221-Thanks you for using the FTP service on spdev.
```

```
221 Goodbye.
```

接続を終了するメッセージに続いて、通常のシェルプロンプトが表示されます。

▼ リモートシステムからファイルをコピーする方法 (ftp)

1. リモートシステムからファイルをコピーしたい、ローカルシステム上のディレクトリに変更します。

   ```
   $ cd target-directory
   ```

2. ftp により接続します。
   682 ページの「ftp によりリモートシステムへ接続する方法」を参照してください。

3. コピー元ディレクトリに変更します。

   ```
   ftp> cd source-directory
   ```
   システムがオートマウンタを使用している場合、リモートシステムのユーザーのホームディレクトリは、/home の下にユーザーのホームディレクトリと並行して表示されます。

4. コピー元ファイルの読み取り権があることを確認します。

   ```
   ftp> ls -l
   ```

5. 転送タイプを binary に設定します。

   ```
   ftp> binary
   ```

6. ファイルを 1 つコピーするには、get コマンドを使用します。

   ```
   ftp> get filename
   ```

7. 一度に複数のファイルをコピーするには、mget コマンドを使用します。

   ```
   ftp> mget filename [filename ...]
   ```
   個々のファイル名を続けて入力するか、ワイルドカード文字を使用できます。mget コマンドでは、個々のファイルがコピーされ、そのたびに確認を求めるプロンプトが表示されます。

8. ftp による接続を終了します。

   ```
   ftp> bye
   ```

例 — リモートシステムからファイルをコピーする (ftp)

次の例では、ユーザー kryten は、システム pluto と ftp 接続し、get コマンドを使用して /tmp ディレクトリから自分のホームディレクトリにファイルを 1 つコピーします。

```
$ cd $HOME
ftp pluto
Connected to pluto.
220 pluto FTP server (SunOS 5.8) ready.
Name (pluto:kryten): kryten
331 Password required for kryten.
Password: xxx
230 User kryten logged in.
ftp> cd /tmp
250 CWD command successful.
ftp> ls
200 PORT command successful.
150 ASCII data connection for /bin/ls (129.152.221.238,34344)
(0 bytes).
dtdbcache_:0
filea
files
ps_data
speckeysd.lock
226 ASCII Transfer complete.
53 bytes received in 0.022 seconds (2.39 Kbytes/s)
ftp> get filea
200 PORT command successful.
150 ASCII data connection for filea (129.152.221.238,34331)
(0 bytes).
221 Goodbye.
```

次の例では、同じユーザー kryten が mget コマンドを使用して、/tmp ディレクトリから自分のホームディレクトリに複数のファイルをコピーします。kryten は、個々のファイルについてコピーするか、しないかの選択ができることに注意してください。

```
$ ftp> cd /tmp
250 CWD command successful.
ftp> ls files
200 PORT command successful.
150 ASCII data connection for /bin/ls (129.152.221.238,34345)
(0 bytes).
fileb
filec
filed
remote: files
21 bytes received in 0.015 seconds (1.36 Kbytes/s)
ftp> cd files
250 CWD command successful.
ftp> mget file*
mget fileb? y
200 PORT command successful.
150 ASCII data connection for fileb (129.152.221.238,34347)
(0 bytes).
226 ASCII Transfer complete.
mget filec? y
200 PORT command successful.
150 ASCII data connection for filec (129.152.221.238,34348)
(0 bytes).
```

```
226 ASCII Transfer complete.
mget filed? y
200 PORT command successful.
150 ASCII data connection for filed (129.152.221.238,34351)
(0 bytes).
226 ASCII Transfer complete.200 PORT command successful.
ftp> bye
221 Goodbye.
```

▼ ファイルをリモートシステムにコピーする方法 (ftp)

1. ローカルシステム上のコピー元ディレクトリに変更します。

   ftp コマンドを入力して接続するディレクトリは、ローカルの作業用ディレクトリ、つまりこの操作のコピー元ディレクトリになります。

2. ftp により接続します。

   682 ページの「ftp によりリモートシステムへ接続する方法」を参照してください。

3. コピー先ディレクトリに変更します。

   ```
   ftp> cd target-directory
   ```

   ローカルシステムでオートマウンタを使用中であれば、/home の下に自分のホームディレクトリと並行してリモートシステムのユーザーのホームディレクトリが表示されるので注意してください。

4. コピー先ディレクトリへの書き込み権があることを確認します。

   ```
   ftp> ls -l target-directory
   ```

5. 転送タイプを **binary** に設定します。

   ```
   ftp> binary
   ```

6. ファイルを **1** つコピーするには、put コマンドを使用します。

   ```
   ftp> put filename
   ```

7. 一度に複数のファイルをコピーするには、mput コマンドを使用します。

   ```
   ftp> mput filename [filename ...]
   ```

   個々のファイル名を続けて入力するか、ワイルドカード文字を使用できます。mput コマンドでは、個々のファイルがコピーされ、そのたびに確認を求めるプロンプトが表示されます。

8. ftp による接続を終了するには、bye と入力します。

   ```
   ftp> bye
   ```

例 — ファイルをリモートシステムにコピーする (ftp)

次の例では、ユーザー kryten はシステム pluto へ ftp により接続し、put コマンドを使用して自分のシステムからシステム pluto の /tmp ディレクトリにファイルをコピーします。

```
$ cd /tmp
ftp pluto
Connected to pluto.
220 pluto FTP server (SunOS 5.8) ready.
Name (pluto:kryten): kryten
331 Password required for kryten.
Password: xxx
230 User kryten logged in.
ftp> cd /tmp
250 CWD command successful.
ftp> put filef
200 PORT command successful.
150 ASCII data connection for filef (129.152.221.238,34356).
226 Transfer complete.
ftp> ls
200 PORT command successful.
150 ASCII data connection for /bin/ls (129.152.221.238,34357) (0 bytes).
dtdbcache_:0
filea
filef
files
ps_data
speckeysd.lock
226 ASCII Transfer complete.
60 bytes received in 0.058 seconds (1.01 Kbytes/s)
ftp> bye
221 Goodbye.
```

次の例では、同じユーザー kryten は mput コマンドを使用して自分のホームディレクトリから pluto の /tmp ディレクトリに複数のファイルをコピーします。kryten は、個々のファイルについてコピーするか、しないかを選択できることに注意してください。

```
$ cd $HOME/testdir
$ ls
test1   test2   test3
$ ftp pluto
Connected to pluto.
220 pluto FTP server (SunOS 5.8) ready.
Name (pluto:kryten): kryten
331 Password required for kryten.
Password: xxx
230 User kryten logged in.
ftp> cd /tmp
250 CWD command successful.
ftp> mput test*
mput test1? y
200 PORT command successful.
```

```
150 ASCII data connection for test1 (129.152.221.238,34365).
226 Transfer complete.
mput test2? y
200 PORT command successful.
150 ASCII data connection for test2 (129.152.221.238,34366).
226 Transfer complete.
mput test3? y
200 PORT command successful.
150 ASCII data connection for filef (129.152.221.238,34356).
226 Transfer complete.
ftp> bye
221 Goodbye.
```

rcp によるリモートコピー

rcp コマンドは、ローカルシステムとリモートシステム間、または 2 台のリモートシステム間でファイルやディレクトリをコピーします。このコマンドは、リモートシステムから (rlogin コマンドでログイン後に)、またはローカルシステムから (リモートシステムにログインせずに) 使用できます。

rcp を使用すると、次のリモートコピー操作を実行できます。

- 自分のシステムからリモートシステムにファイルやディレクトリをコピーする。
- リモートシステムからローカルシステムにファイルやディレクトリをコピーする。
- ローカルシステムを経由したリモートシステム間でファイルやディレクトリをコピーする。

オートマウンタを実行中の場合は、これらのリモート操作を cp コマンドで実行できます。しかし、cp の範囲は、オートマウンタにより作成された仮想ファイルシステムと、ユーザーのホームディレクトリから相対的に指定できる操作に制限されます。rcp はそのような制限を受けずに同じ操作を実行するので、ここでは rcp を使用する場合に限定して説明します。

コピー操作のセキュリティ上の注意事項

システム間でファイルやディレクトリをコピーするには、ログインしてファイルをコピーする許可を持っていなければなりません。

注意 – cp コマンドと rcp コマンドではともに、警告が表示されずにファイルが上書きされることがあります。コマンドを実行する前に、ファイル名が正しいかどうかを確認してください。

コピー元とコピー先の指定

C シェル内で rcp コマンドを使用すると、絶対パス名または相対パス名を使用して、コピー元 (コピーしたいファイルやディレクトリ) とコピー先 (ファイルやディレクトリをコピーする場所) を指定できます。

	絶対パス名	相対パス名
ローカルシステムから	mars:/home/jones/myfile.txt	~jones/myfile.txt
リモートログイン後	/home/jones/myfile.txt	~jones/myfile.txt

絶対パス名は、特定のシステムにマウントされているファイルやディレクトリを表します。上記の例で、第 1 の絶対パス名は mars システム上のファイル (MyFile.txt) を表します。相対パス名は、ファイルやディレクトリがある位置を、ユーザーのホームディレクトリからの相対パスで表します。上記の例で、相対パス名は絶対パスと同じ MyFile.txt を表しますが、jones のホームディレクトリを示すために「~」(チルド記号) を使用しています。

```
~ = mars:/home/jones
```

上記の 2 行目の例は、リモートログイン後の絶対パス名と相対パス名を示しています。相対パス名では明確な違いは見られません。しかし、リモートログイン操作により、jones のホームディレクトリがローカルシステム上にマウントされた (ローカルユーザーのホームディレクトリと並列に存在する) ので、絶対パス名ではシステム名 mars を指定する必要はありません。リモートログイン操作によって別のユーザーのホームディレクトリがどのようにマウントされるかについては、676 ページの「リモートログイン後の処理」を参照してください。

表 44–4 に、C シェルが認識する絶対パス名と相対パス名の例を示します。このサンプルでは、次の用語を使用します。

- 作業用ディレクトリ — rcp コマンドを入力するディレクトリ。リモート、ローカルのどちらの場合もあり
- 現在のユーザー — rcp コマンドを入力するユーザーの名前

表 **44–4** ディレクトリ名とファイル名に使用できる構文

ログイン先	構文	説明
ローカルシステム	.	ローカルの作業用ディレクトリ
	path/filename	ローカルの作業用ディレクトリ内の *path* と *filename*

表 44–4 ディレクトリ名とファイル名に使用できる構文　(続き)

ログイン先	構文	説明
	~	現在のユーザーのホームディレクトリ
	~/path/filename	現在のユーザーのホームディレクトリの下の *path* と *filename*
	~user	*user* のホームディレクトリ
	~user/path/filename	*user* のホームディレクトリの下の *path* と *filename*
	remote-system:path/filename	リモートの作業用ディレクトリ内の *path* と *filename*
リモートシステム	.	リモートの作業用ディレクトリ
	filename	リモートの作業用ディレクトリ内の *filename*
	path/filename	リモートの作業用ディレクトリ内の *path* と *filename*
	~	現在のユーザーのホームディレクトリ
	~/path/filename	現在のユーザーのホームディレクトリ内の *path* と *filename*
	~user	*user* のホームディレクトリ
	~user/path/filename	*user* のホームディレクトリの下の *path* と *filename*
	local-system:path/filename	ローカルの作業用ディレクトリ内の *path* と *filename*

▼ ローカルシステムとリモートシステム間でファイルをコピーする方法 (rcp)

1. コピーする許可を持っているかどうかを確認します。
   少なくとも、コピー元システム上で読み取り権を持ち、コピー先システム上で書き込み権を持っていなければなりません。
2. コピー元とコピー先の位置を決定します。
   コピー元またはコピー先のパスがわからない場合は、まず `rlogin` コマンドを使用してリモートシステムにログインし、位置が見つかるまでリモートシステム上を移動できます。手順については、679 ページの「リモートシステムにログインする方法 (rlogin)」を参照してください。 その後は、ログアウトしなくても次の手順を実行できます。
3. ファイルまたはディレクトリをコピーします。

```
$ rcp [-r] source-file | directory   target-file | directory
```

rcp	(オプションなし) コピー元からコピー先にファイルを 1 つコピーする
-r	コピー元からコピー先にディレクトリをコピーする

この構文は、リモートシステムとローカルシステムのどちらにログインするかに関係なく適用されます。表 44–4 で説明したとおり、ファイルやディレクトリのパス名のみをこの後で示す例のように変更します。

「~」と「.」を使用して、ローカルのファイル名やディレクトリ名のパス部分を指定できます。 ただし、「~」はリモートシステムではなく現在のユーザーに適用されることと、「.」はログイン先のシステムに適用されることに注意してください。 この 2 つの記号については、表 44–4 を参照してください。

例 — ローカルシステムとリモートシステム間でファイルをコピーする (rcp)

次に、rcp を使用してローカルシステムとリモートシステム間でファイルをコピーする使用例を示します。

例 **44–1** rcp を使用してリモートファイルをローカルシステムにコピーする

次の例では、rcp はファイル letter.doc をリモートシステム pluto の /home/jones ディレクトリから、ローカルシステム earth 上の作業用ディレクトリ (/home/smith) にコピーします。

```
earth(/home/smith): rcp pluto:/home/jones/letter.doc .
```

この例では、リモートログインをしないで rcp 操作を実行しています。コマンド行の最後にある「.」記号は、リモートシステムではなく、ローカルシステムを表します。

コピー先ディレクトリもローカルユーザーのホームディレクトリなので、「~」 記号で指定することもできます。

```
earth(home/smith): rcp pluto:/home/jones/letter.doc ~
```

例 **44–2** rlogin と rcp を使用してリモートファイルをローカルシステムにコピーする

次の例では、rlogin コマンドの実行後に rcp 操作が実行され、リモートシステムからローカルシステムにファイルをコピーしています。操作の流れは前述の例と同じですが、リモートログインによりパスが変更になります。

```
earth(/home/smith): rlogin pluto
.
.
.
pluto(/home/jones): rcp letter.doc ~
```

例 **44–2** rlogin と rcp を使用してリモートファイルをローカルシステムにコピーする (続き)

コマンド行の最後に「.」記号を使用するのは、この例では不適切です。 リモートログインが行われているので、「.」記号はリモートシステムを指し、実際には rcp に重複したファイルを作成させることになります。ただし、「~」は、リモートシステムにログインするときにも現在のユーザーのホームディレクトリを指します。

例 **44–3** rcp を使用してローカルファイルをリモートシステムにコピーする

次の例で、rcp はファイル notice.doc をローカルシステム earth のホームディレクトリ (/home/smith) からリモートシステム pluto の /home/jones ディレクトリにコピーします。

```
earth(/home/smith): rcp notice.doc pluto:/home/jones
```

リモートファイル名が指定されていないので、ファイル notice.doc は /home/jones ディレクトリに同じ名前でコピーされます。

次の例では、前の例と同じように rcp 操作が行われますが、rcp はローカルシステム上の別の作業用ディレクトリ (/tmp) で入力されます。現在のユーザーのホームディレクトリを指すために「~」記号が使われているので注意してください。

```
earth(/tmp): rcp ~/notice.doc pluto:/home/jones
```

例 **44–4** rlogin と rcp を使用してローカルファイルをリモートシステムにコピーする

次の例では、rlogin コマンドの実行後に rcp 操作が実行され、ローカルファイルをリモートディレクトリにコピーしています。操作の流れは前に示した例と同じですが、リモートログインによりパスが変更になります。

```
earth(/home/smith): rlogin pluto
.
.
.
pluto(/home/jones): rcp ~/notice.doc .
```

現在のユーザーのホームディレクトリはローカルシステム上にありますが、「~」記号によりそのディレクトリが表されます。ユーザーはリモートシステムにログインしているので、「.」記号はリモートシステム上の作業用ディレクトリを表します。 次の構文を使用しても同じ操作を実行します。

```
pluto(/home/jones): rcp earth:/home/smith/notice.doc /home/jones
```

第 45 章

ネットワークサービスの監視 (トピック)

第 46 章	ネットワークサービスの監視の手順

第 46 章

ネットワークパフォーマンスの監視 (手順)

この章ではネットワークのパフォーマンスを監視する方法について説明します。この章で説明する手順は次のとおりです。

ネットワークパフォーマンスの監視

表 46–1 に、ネットワークのパフォーマンスを監視するために使用できるコマンドを示します。

表 46–1 ネットワーク監視コマンド

コマンド名	説明
`ping`	ネットワーク上でホストの応答を調べる
`spray`	送信したパケットサイズの信頼性を検査する。パケットが遅延されていないか、落とされていないか判定できる
`snoop`	ネットワークからパケットを捕捉し、各クライアントから各サーバーへの呼び出しを追跡する
`netstat`	TCP/IP トラフィックに使用されるインタフェースや IP ルーティングテーブルなどに関するネットワーク状態と、`UDP`、`TCP`、`ICMP`、および `IGMP` についてのプロトコル別の統計情報を表示する

表 **46–1** ネットワーク監視コマンド　(続き)

コマンド名	説明
nfsstat	NFS の問題を解析するのに使用できる、サーバーおよびクライアントの統計情報の要約を表示する

▼ ネットワーク上でホストの応答を検査する方法

ping コマンドを使用して、ネットワーク上のホストの応答を検査します。

```
$ ping hostname
```

物理的な問題が発生していると思われる場合は、ping コマンドを使用して、ネットワーク上にあるホストの応答時間を調べることができます。あるホストからの応答が期待していたものと異なる場合は、そのホストについて調査します。物理的な問題が発生する理由を次に示します。

- ケーブルまたはコネクタの緩み
- 接地不良
- 終端処理の欠落
- 信号の反射

このコマンドの詳細については ping(1M) のマニュアルページを参照してください。

例 – ネットワーク上のホストの応答を検査する

最も簡単な ping コマンドの使い方は、ネットワーク上のホストへ 1 つのパケットを送信することです。ping コマンドが正しい応答を受信すると、"*host* is alive" というメッセージが表示されます。

```
$ ping elvis
elvis is alive
```

-s オプションを指定すると、ping は 1 秒ごとにデータグラムをホストへ送り、次に各応答と、その往復に要した時間を表示します。 次に例を示します。

```
$ ping -s pluto
64 bytes from pluto (123.456.78.90): icmp_seq=0. time=10. ms
64 bytes from pluto (123.456.78.90): icmp_seq=5. time=0. ms
64 bytes from pluto (123.456.78.90): icmp_seq=6. time=0. ms
^C
----pluto PING Statistics----
8 packets transmitted, 8 packets received, 0% packet loss

round-trip (ms) min/avg/max = 0/2/10
```

▼ ネットワーク上でホストへパケットを送信する方法

spray コマンドを使用すると、送信したパケットサイズの信頼性を検査できます。

```
$ spray [ -c count -d interval -l packet_size] hostname
```

-i *count*	送信するパケット数
-d *interval*	パケットの送信ごとに一時停止するマイクロ秒数。遅延を使用しないと、バッファーを使い果たす可能性がある
-l *packet_size*	パケットサイズ
hostname	パケットを送信するシステム

このコマンドの詳細は、spray(1M) のマニュアルページを参照してください。

例 – ネットワーク上のホストへパケットを送信する

次の例では、各パケットサイズが 2048 バイト (-l 2048) のパケット 100 個 (-c 100) を、ホストへ送信します。パケットは、各バースト間に 20 マイクロ秒の遅延時間 (-d 20) を入れて送信されます。

```
$ spray -c 100 -d 20 -l 2048 pluto
sending 100 packets of length 2048 to pluto ...
no packets dropped by pluto
279 packets/sec, 573043 bytes/sec
```

▼ ネットワークからパケットを捕捉する方法

ネットワークからパケットを捕捉し、各クライアントから各サーバーへの呼び出しを追跡するには、snoop コマンドを使用します。このコマンドは、ネットワークのパフォーマンスの問題を素早く解析するための、正確なタイムスタンプを提供します。詳細は、snoop(1M) のマニュアルページを参照してください。

```
# snoop
```

パケットが落とされるのは、バッファーの領域不足か、CPU の過負荷が原因となっている場合があります。

▼ ネットワークの状態を調べる方法

netstat コマンドを使用すると、ネットワークインタフェースやルーティングテーブルなどに関するネットワーク状態と、各種プロトコルについての統計情報を表示できます。

```
$ netstat [-i] [-r] [-s]
```

-i	TCP/IP インタフェースの状態を表示する
-r	IP ルーティングテーブルを表示する
-s	UDP、TCP、ICMP、および IGMP プロトコルについての統計情報を表示する

詳細は、netstat(1M) のマニュアルページを参照してください。

例 – ネットワークの状態を調べる

次の表示例は、netstat -i コマンドの出力を示したものです。このコマンドは、TCP/IP トラフィックに使用されるインタフェースの情報を表示します。

```
$ netstat -i
Name  Mtu  Net/Dest    Address      Ipkts  Ierrs Opkts  Oerrs Collis Queue
lo0   8232 software    localhost     1280   0     1280     0       0    0
le0   1500 loopback    venus      1628480   0   347070    16   39354    0
```

上記の表示例は、マシンが各インタフェース上で送受信したパケット数を示しています。有効なネットワークトラフィックが存在するマシンでは、Ipkts と Opkts が継続的に増加しています。

ネットワーク衝突率は、衝突カウント (Collis) を出力パケットの数 (Opkts) で割ることにより算出できます。上記の例では、衝突率は 11%です。ネットワーク全体の衝突率が 5 ～ 10 %を超える場合には、問題が発生している可能性があります。

入力パケットエラー率 (Ierrs/Ipkts) は、入力エラー数を合計入力パケット数で割ることにより算出できます。出力パケットエラー率 (Oerrs/Opkts) は、出力エラー数を合計出力パケット数で割ることにより算出できます。入力エラー率が高い場合 (0.25% を超えている場合)、ホストがパケットを落としている可能性があります。

次に、netstat -s コマンドの出力例を示します。このコマンドは、UDP、TCP、ICMP、および IGMP についてプロトコル別の統計情報を表示します。

```
UDP
    udpInDatagrams      =196543    udpInErrors         =     0
    udpOutDatagrams     =187820

TCP
 tcpRtoAlgorithm     =     4    tcpRtoMin           =   200
    tcpRtoMax           = 60000    tcpMaxConn          =    -1
    tcpActiveOpens      = 26952    tcpPassiveOpens     =   420
    tcpAttemptFails     =  1133    tcpEstabResets      =     9
    tcpCurrEstab        =    31    tcpOutSegs          =3957636
    tcpOutDataSegs      =2731494    tcpOutDataBytes     =1865269594
    tcpRetransSegs      = 36186    tcpRetransBytes     =3762520
    tcpOutAck           =1225849    tcpOutAckDelayed    =165044
```

```
    tcpOutUrg           =     7    tcpOutWinUpdate     =   315
    tcpOutWinProbe      =     0    tcpOutControl       = 56588
    tcpOutRsts          =   803    tcpOutFastRetrans   =   741
    tcpInSegs           =4587678
    tcpInAckSegs        =2087448    tcpInAckBytes       =1865292802
    tcpInDupAck         =109461    tcpInAckUnsent      =     0
    tcpInInorderSegs    =3877639    tcpInInorderBytes   =-598404107
    tcpInUnorderSegs    = 14756    tcpInUnorderBytes   =17985602
    tcpInDupSegs        =    34    tcpInDupBytes       = 32759
    tcpInPartDupSegs    =   212    tcpInPartDupBytes   =134800
    tcpInPastWinSegs    =     0    tcpInPastWinBytes   =     0
    tcpInWinProbe       =   456    tcpInWinUpdate      =     0
    tcpInClosed         =    99    tcpRttNoUpdate      =  6862
    tcpRttUpdate        =435097    tcpTimRetrans       = 15065
    tcpTimRetransDrop   =    67    tcpTimKeepalive     =   763
    tcpTimKeepaliveProbe=     1    tcpTimKeepaliveDrop =     0

IP
 ipForwarding        =     2    ipDefaultTTL        =   255
    ipInReceives        =11757234    ipInHdrErrors       =     0
    ipInAddrErrors      =     0    ipInCksumErrs       =     0
    ipForwDatagrams     =     0    ipForwProhibits     =     0
    ipInUnknownProtos   =     0    ipInDiscards        =     0
    ipInDelivers        =4784901    ipOutRequests       =4195180
    ipOutDiscards       =     0    ipOutNoRoutes       =     0
    ipReasmTimeout      =    60    ipReasmReqds        =  8723
    ipReasmOKs          =  7565    ipReasmFails        =  1158
    ipReasmDuplicates   =     7    ipReasmPartDups     =     0
    ipFragOKs           = 19938    ipFragFails         =     0
    ipFragCreates       =116953    ipRoutingDiscards   =     0
    tcpInErrs           =     0    udpNoPorts          =6426577
    udpInCksumErrs      =     0    udpInOverflows      =   473
    rawipInOverflows    =     0

ICMP
 icmpInMsgs          =490338    icmpInErrors        =     0
    icmpInCksumErrs     =     0    icmpInUnknowns      =     0
    icmpInDestUnreachs  =   618    icmpInTimeExcds     =   314
    icmpInParmProbs     =     0    icmpInSrcQuenchs    =     0
    icmpInRedirects     =   313    icmpInBadRedirects  =     5
    icmpInEchos         =   477    icmpInEchoReps      =    20
    icmpInTimestamps    =     0    icmpInTimestampReps =     0
    icmpInAddrMasks     =     0    icmpInAddrMaskReps  =     0
    icmpInFragNeeded    =     0    icmpOutMsgs         =   827
    icmpOutDrops        =   103    icmpOutErrors       =     0
    icmpOutDestUnreachs =    94    icmpOutTimeExcds    =   256
    icmpOutParmProbs    =     0    icmpOutSrcQuenchs   =     0
    icmpOutRedirects    =     0    icmpOutEchos        =     0
    icmpOutEchoReps     =   477    icmpOutTimestamps   =     0
    icmpOutTimestampReps=     0    icmpOutAddrMasks    =     0
    icmpOutAddrMaskReps =     0    icmpOutFragNeeded   =     0
    icmpInOverflows     =     0
IGMP:
        0 messages received
        0 messages received with too few bytes
```

```
        0 messages received with bad checksum
        0 membership queries received
        0 membership queries received with invalid field(s)
        0 membership reports received
        0 membership reports received with invalid field(s)
        0 membership reports received for groups to which we belong
        0 membership reports sent
```

次に、netstat -r コマンドの出力例を示します。このコマンドは、IP ルーティングテーブルを表示します。

```
Routing Table:
  Destination        Gateway           Flags  Ref   Use    Interface
------------------ -------------------- ----- ----- ------ ---------
localhost            localhost             UH       0   2817  lo0
earth-bb             pluto                 U        3  14293  le0
224.0.0.0            pluto                 U        3      0  le0
default              mars-gate             UG       0  14142
```

表 46–2 は、netstat -r コマンドが出力するレポート中のフィールドを説明します。

表 **46–2** netstat -r コマンドの出力

フィールド名		説明
Flags	U	ルートが正常に動作している
	G	ルートはゲートウェイを経由する
	H	ルートはホスト宛である
	D	ルートはリダイレクトを使用して動的に作成された
Ref		同じリンク層を共有している現在のルート数を示す
Use		送信されたパケット数を示す
Interface		ルートに使用されるネットワークインタフェースを表示する

▼ NFS サーバーとクライアントの統計情報を表示する方法

NFS 分散型ファイルサービスは、ローカルコマンドをリモートホストへの要求に変換する、リモート手続き呼び出し (RPC) 機能を使用します。リモート手続き呼び出しは同期型の呼び出しです。つまり、サーバーが呼び出しを完了してその結果を返すまで、クライアントアプリケーションはブロックまたは中断されます。NFS のパフォーマンスに影響を与える主要な要素の 1 つに再伝送率があります。

ファイルサーバーがクライアントの要求に応答できない場合、そのクライアントは、指定された回数だけ要求を再伝送して終了します。再伝送のたびにシステムにオーバーヘッドがかかり、ネットワークトラフィックが増加します。過度の再伝送はネットワークのパフォーマンスを低下させます。再伝送率が高い場合、次を調べてください。

- サーバーが過負荷になっており、要求の処理に時間がかかりすぎていないか
- Ethernet インタフェースがパケットを落としていないか
- ネットワークの輻輳によりパケットの伝送が低下していないか

表 46–3 に、クライアントとサーバーの統計情報を表示するための `nfsstat` コマンドのオプションとその説明を示します。

表 **46–3** クライアントとサーバーの統計情報を表示するためのコマンド

コマンド名	表示される情報
`nfsstat -c`	クライアントの統計情報
`nfsstat -s`	サーバーの統計情報
`netstat -m`	ファイルシステムごとのネットワーク統計情報

クライアントの統計情報を表示するには `nfsstat -c` を使用し、サーバーの統計情報を表示するには `nfsstat -s` を使用します。また、ファイルシステムごとのネットワークの統計情報を表示するには、`nfsstat -m` を使用します。詳細は、`nfsstat (1M)` のマニュアルページを参照してください。

例 – NFS サーバーとクライアントの統計情報を表示する

次の例は、クライアント `pluto` の RPC と NFS データを表示します。

```
$ nfsstat -c

        Client rpc:
Connection oriented:
calls    badcalls  badxids  timeouts newcreds  badverfs   timers
1595799  1511      59       297      0         0          0
cantconn nomem     interrupts
1198      0         7
Connectionless:
calls    badcalls  retrans  badxids  timeouts  newcreds   badverfs
80785    3135      25029    193      9543      0          0
timers   nomem     cantsend
17399    0         0

Client nfs:
calls    badcalls  clgets   cltoomany
1640097  3112      1640097  0
Version 2: (46366 calls)
null     getattr   setattr  root     lookup     readlink  read
```

```
0 0%      6589 14%  2202 4%  0 0%      11506 24%  0 0%       7654 16%
wrcache   write     create   remove    rename     link       symlink
0 0%      13297 28% 1081 2%  0 0%      0 0%       0 0%       0 0%
mkdir     rmdir     readdir  statfs
24 0%     0 0%      906 1%   3107 6%
Version 3: (1585571 calls)
null      getattr   setattr  lookup    access     readlink   read
0 0%      508406 32% 10209 0% 263441 16% 400845 25% 3065 0%   117959 7%
write     create    mkdir    symlink   mknod      remove     rmdir
69201 4%  7615 0%   42 0%    16 0%     0 0%       7875 0%    51 0%
rename    link      readdir  readdir+  fsstat     fsinfo     pathconf
929 0%    597 0%    3986 0%  185145 11% 942 0%    300 0%     583 0%
commit
4364 0%

Client nfs_acl:
Version 2: (3105 calls)
null      getacl    setacl   getattr   access
0 0%      0 0%      0 0%     3105 100% 0 0%
Version 3: (5055 calls)
null      getacl    setacl
0 0%      5055 100% 0 0%
```

表 46–4 に、nfsstat -c コマンドの出力とその説明を示します。

表 **46–4** nfsstat -c コマンドの出力とその説明

フィールド	説明
calls	送信された合計呼び出し数
badcalls	RPC によって拒否された合計呼び出し数
retrans	再伝送の合計数。このクライアントの場合、再伝送回数は 1% 未満 (6888 回の呼び出しのうち、10 回のタイムアウト)。再伝送は一時的な異常により発生する可能性がある。 1% 以上の再伝送率の場合は、問題が発生している可能性がある
badxid	1 つの NFS 要求に対して重複する承認を受信した回数
timeout	タイムアウトした呼び出しの回数
wait	利用可能なクライアントハンドルがないために呼び出しが待機した回数
newcred	認証情報を書き換えなければならなかった回数
timers	タイムアウト値が、呼び出しに対して指定されたタイムアウト値以上であった回数
readlink	シンボリックリンクに対して読み取りが行われた回数。この値が大きすぎる (10% を超える) 場合は、シンボリックリンクが多すぎる可能性がある

次に、nfsstat -m コマンドの出力例を示します。

```
pluto$ nfsstat -m
/usr/man from pluto:/export/svr4/man
```

```
Flags: vers=2,proto=udp,auth=unix,hard,intr,dynamic,
        rsize=8192, wsize=8192,retrans=5
 Lookups: srtt=13 (32ms), dev=10 (50ms), cur=6 (120ms)
 All:     srtt=13 (32ms), dev=10 (50ms), cur=6 (120ms)
```

表 46–5 に、ミリ秒単位で表示される nfsstat -m コマンドの出力を示します。

表 **46–5** nfsstat -m コマンドの出力

フィールド	説明
srtt	平準化された平均往復時間
dev	平均偏差
cur	現在の「予測」応答時間

ネットワークのハードウェアに問題の原因があると思われる場合は、ケーブルおよびコネクタを綿密にチェックしてください。

付録 A

『Solaris のシステム管理 (資源管理とネットワークサービス)』の更新情報

Solaris 9 9/02 Update リリースでの更新情報

第 7 章では、IPQoS フローアカウンティングモジュールを拡張アカウンティングと組み合わせて使用して、ネットワークフローの統計情報を取得する方法についての説明が追加されました。

用語集

`asppp`	Solaris 2.4 から Solaris 8 リリースまでの Solaris オペレーティング環境に含まれる PPP のバージョンの 1 つ。`asppp` は非同期 PPP 通信のみサポートする。
CHAP シークレット	識別目的で使用される ASCII またはバイナリ文字列。PPP リンク上の両ピアにより認識される。CHAP シークレットはシステムの `/etc/ppp/chap-secrets` ファイル内に平文のまま保存されるが、PPP リンク上には、たとえ暗号化された形であっても、決して送信されることはない。CHAP プロトコルは、呼び出し元が使用する CHAP シークレットのハッシュと、受け取り側の `/etc/ppp/chap-secrets` ファイルに設定されている呼び出し元の CHAP シークレットエントリのハッシュが一致することを検証する。
CSU/DSU	CSU デバイスと DSU デバイスを組み合わせた同期通信装置。専用回線 PPP リンク上で使用する。CSU/DSU はピアからの信号を専用回線に変換する。CSU/DSU の多くはリンクを確立するためのチャットスクリプトを必要としない。CSU/DSU は専用回線プロバイダにより構成されることが多い。 チャネルサービス装置 (CSU) と 加入者線終端装置 (DSU) も参照。
ISDN 端末アダプタ **(TA)**	信号変換装置。ISDN ネットワーク上でダイアルアップ PPP リンクにモデムと同等のインタフェースを提供する。ISDN TA を構成する場合、標準モデムを構成する場合と同じ Solaris PPP 4.0 構成ファイルを使用する。
Microsoft CHAP (MS-CHAP)	独自の PPP 用 Microsoft 認証プロトコル。Solaris PPP 4.0 では、クライアントモードとサーバーモードの両方において、このプロトコルのバージョン 1 と 2 をサポートする。
PPPoE (PPP over Ethernet)	RedBack Networks 独自のプロトコル。このプロトコルを使用して、ホストがイーサネットリンク上で PPP セッションを実行できる。PPPoE は通常デジタル加入者回線 (DSL) サービスで使用される。

SLP デーモン **(slpd)** SLP の Solaris 実装で DA または SA サーバーとして動作するデーモンプロセス。ホスト上でのサービス処理は、通知を個々に保持するのではなく、`slpd` を使用してサービス通知を登録する。SLP デーモンが SA サーバーとして構成される場合、各プロセスには、`slpd` と通信する SA クライアントライブラリが含まれる。SLP デーモンはすべての登録と登録解除を DA に転送する。デーモンは有効期限が切れたサービス通知を時間切れとし、アクティブまたはパッシブな DA 検出を実行して、利用可能な DA のテーブルを保守する。これらの仕組みを通して、DA の情報が UA クライアントに提供される。UA クライアントは DA 情報についてのみホスト上で `slpd` を使用する。SLP デーモンは Solaris 9 オペレーティング環境の一部としてホスト上にインストールされる。オプションで `slpd` を DA として構成できる。

アカウンティングの拡張 Solaris オペレーティング環境で、タスクまたはプロセスに基づくの資源消費量を柔軟に記録できる方法。

圧縮制御プロトコル **(CCP)** PPP のサブプロトコル。リンク上でのデータ圧縮の使用についてネゴシエーションを行う。ヘッダー圧縮とは異なり、CCP はリンク上に送信されたパケット内のすべてのデータを圧縮する。

インターネットプロトコル制御プロトコル **(IPCP)** PPP のサブプロトコル。リンク上のピアの IP アドレスについてネゴシエーションを行う。また、リンクのヘッダー圧縮をネゴシエーションし、ネットワーク層プロトコルを使用可能にする。

インターネットプロトコルバージョン **6** 制御プロトコル **(IPV6CP)** インターネットプロトコル制御プロトコル (IPCP) を参照。

加入者線終端装置 **(DSU)** 専用回線 PPP リンク上で使用する同期通信装置。DSU は通信回線上で使用されるデータフレーミング形式間の変換を行い、標準データ通信インタフェースを提供する。

チャネルサービス装置 (CSU) と CSU/DSU も参照。

コールバックコントロールプロトコル **(CBCP)** Microsoft 独自の PPP 拡張機能。コールバックセッションのネゴシエーションに使用する。Solaris PPP 4.0 ではこのプロトコルのクライアント側 (最初の呼び出し側) のみサポートする。

サービス **URL** サービスのネットワークロケーションを通知するために使用される URL。URL は、サービスの種類、ホスト名、サービスホストのネットワークアドレスから構成される。URL には、ポート番号や、サービスを使用するために必要なその他の情報が使用される場合もある。

サービスエージェント **(SA)** ネットワークサービスのサービス通知を保守する SLP エージェント。DA が使用できない場合は、SA が UA からのサービス要求のマルチキャストに答える。DA が使用できる場合は、SA はそのスコープをサポートする DA にサービスを登録、あるいはオプションで登録解除する。

サービス通知 サービスを定義する SA により配布される情報。サービス通知は、サービスを説明する、URL および、属性と値の対のリストの集合。す

	べてのサービス通知には有効期限がある。期限が切れると、サービス通知は再登録されない限り無効になる。
資源	資源管理において、アプリケーションの動作を変更するために操作可能な計算機システムの一部。
資源管理	利用可能なシステム資源のアプリケーションによる使用方法を制御可能にする機能。
資源制御	資源管理において、タスクおよびプロジェクトのエンティティまで拡張されたプロセスごとの資源制限値。
信頼できる呼び出し元	PPP において、ダイアルインサーバーがアクセスを許可するリモートピア。リモートピアのセキュリティ資格をダイアルインサーバーの PAP または CHAP シークレットデータベースに追加することによりアクセスを許可する。
専用回線 **PPP** リンク	ホストと、プロバイダからリースした同期ネットワーク媒体に接続された CSU/DSU からなる PPP 接続。専用回線媒体の一般的な例として OC3、T1 がある。管理は簡単だが、専用回線リンクはダイアルアップ PPP リンクよりも費用がかかることから、広くは使われていない。
ダイアルアウトマシン	ダイアルアップリンクを確立するための呼び出しを開始するピア。構成後は、ダイアルアウトマシンは任意の台数のダイアルインサーバーを呼び出すことができる。一般に、ダイアルアップリンクを確立するには、ダイアルアウトマシンが認証資格を提供する必要がある。
ダイアルアップ **PPP** リンク	電話回線または ISDN が提供する媒体など、通信媒体の一方の端にピアとモデムが使用されている PPP 接続。「ダイアルアップ」という用語は、ローカルモデムがリモートピアの電話番号を使用してダイアルアップする場合のリンクネゴシエーションにおけるシーケンスを指す。ダイアルアップリンクは最も広く使用され、最小コストの PPP 構成である。
ダイアルインサーバー	ダイアルアウトマシンから呼び出しを受け、ダイアルアップ PPP リンクの受け取り側をネゴシエーションし、確立するピア。「ダイアルインサーバー」という用語が一般に使用されているが、クライアントサーバーという形では動作しない。形としては、ピアがダイアルアップリンクの設定要求に応答するだけである。構成後は、ダイアルインサーバーは任意の台数のダイアルアウトマシンからの呼び出しを受信できる。
タスク	資源管理において、長時間にわたる作業の集合を表すプロセスの集まり。各タスクは 1 つのプロジェクトに関連付けられる。
チャットスクリプト	モデムとリモートピアの間の通信リンクを確立する方法を、モデムに指示する手順。PPP プロトコルと UUCP プロトコルは、ともにダイアルアップリンク確立とダイアルバック呼び出しにチャットスクリプトを使用する。
チャネルサービス装置 **(CSU)**	専用通信回線へのローカルインタフェースを提供し、その回線を終端する同期通信装置。米国内では、CSU は T1 回線を終端し、DS1 イン

	タフェースまたは DSX インタフェースを提供する。国際的には、電話会社プロバイダが CSU を所有するのが一般的である。 CSU/DSUと加入者線終端装置 (DSU)も参照。
チャレンジハンドシェーク認証プロトコル **(CHAP)**	PPP リンク上の発呼者識別情報の検証に使用できる認証プロトコル。CHAP 認証では、チャレンジと応答の概念を使用する。呼び出しを受信したマシンが呼び出し側にチャレンジを送信してその識別情報を確認する。 パスワード認証プロトコル (PAP)も参照。
ディレクトリエージェント **(DA)**	オプションの SLP エージェント。サービスエージェント (SA) が送信するサービス通知をキャッシュに保存し、維持する。DA が配置された場合、DA がユーザーエージェント (UA) のサービス要求を解決する。DA は ディレクトリ通知に対して、SA および UA からの能動的な要請に応答する。その応答により、SA と UA は関連付けられた DA とスコープを検出する。DA は定期的に請求されていない通知を送るが、この通知を通して SA および UA は共有のスコープ内で DA を検出する。
適用範囲	管理上、位相上、またはその他の関係により整理された UA と SA のグループ化。適用範囲を使用して、企業全体のサービスへのアクセスを提供する方法を変更できる。
同期 **PPP**	同期デジタル回線上の PPP の形式。生のビットを連続ストリームとして転送する。専用回線 PPP リンクは同期 PPP を使用する。
認証	プログラムなどのエンティティまたはリモートユーザーがネットワークを通して提供する識別情報の検証作業。一部の認証プロトコルでは、潜在的ユーザーから認証資格のデータベースを構築できる。その他の認証プロトコルでは、認証を目的として認証局が生成する信頼の証明書チェーンを使用する。これらの資格を使用して、通信やサイトのサービスの利用を要求するユーザーを認証することができる。
パスワード認証プロトコル **(PAP)**	PPP リンク上での発呼者識別情報の検証に使用できる認証プロトコル。PAP は平文パスワードを使用し、このパスワードはリンク上に送信されるので、パスワードを端点のマシンの中の 1 つに保存できる。たとえば、呼び出しを受信するマシン上の UNIX `password` データベース内のログインとパスワードエントリを使用して、呼び出し元の識別情報を検証することができる。 チャレンジハンドシェーク認証プロトコル (CHAP) も参照。
ピア	PPP では、PPP 通信リンクの一端にある 1 台のコンピュータのこと。PPP 通信リンクは、通信媒体により接続された 2 台のピアから構成される。ワークステーション、パソコン、ルーター、メインフレームなど、多様な機器をピアとして構成可能。

非同期 **PPP**	非同期シリアル回線上の PPP の形式。同時に 1 文字ずつデータ転送する。最も一般的な PPP の形式であるダイアルアップリンクでは、非同期 PPP 通信が使用されている。
フェアシェアスケジューラ	公平さを基準に CPU 時間を割り当てるスケジューリングクラス。シェアは、システムの CPU 資源のうちプロジェクトに割り当てる部分を定義する。
プール	資源管理において、マシン資源の区分化に使用される構成機構。
ブロードキャスト	サブネット上の全マシンにパケットを転送するデータリンク層の手順。一般にブロードキャストパケットがサブネットを超えてルーティングされることはない。
プロジェクト	ネットワーク全体の関連作業に対する管理識別子。
ポイントツーポイントプロトコル **(PPP)**	ポイントツーポイント媒体上でデータグラムを転送する標準方法を提供するデータリンク層プロトコル。PPP 構成はピアと呼ばれる 2 台の端点コンピュータ、およびピアが通信に使用する電話回線またはその他の双方向リンクから構成される。2 台のピア間のハードウェアおよびソフトウェア接続が *PPP* リンクであると考えられる。 PPP は、PAP、CHAP、LCP、CCP などの複数のサブプロトコルから構成される。利用できる PPP 実装は多数存在する。Solaris 9 オペレーティング環境では Solaris PPP 4.0 が実装されている。
マルチキャスト	ネットワーク層手順。IP ネットワーク上の複数マシンにデータグラムパケットを送信するのに使用される。ブロードキャストルーティングの場合と同じく、パケットはすべてのマシンによって処理されるわけではない。マルチキャストでは、ルーターを特殊なルーティングプロトコルで構成する必要がある。
ユーザーエージェント **(UA)**	ユーザーアプリケーションの代わりに動作する SLP エージェント。ユーザーエージェントは、対応する適用範囲、ディレクトリエージェント、サービス通知の識別情報を問い合わせる。
予期-送信 **(expect-send)**	PPP チャットスクリプトや UUCP チャットスクリプトで使用されるスクリプト記述形式。チャットスクリプトは、リモートピアからの受け取りを期待する *(expect)* テキストまたは手順で始まる。次の行には、リモートピアから期待どおりの文字列を受信した後にローカルホストが送信する *(send)* 応答が記述される。その後に続く行では、通信確立に必要な手順が正常にネゴシエーションされるまで、ローカルホストとリモートピア間の予期-送信 (expect-send) 手順が繰り返される。
リンク	PPP では、2 つのピア間でネゴシエーションされ、確立される通信接続のこと。Solaris PPP 4.0 では、ダイアルアップと専用回線の 2 種類のリンクをサポートする。
リンク制御プロトコル **(LCP)**	PPP のサブプロトコル。ピア間リンクパラメータの初期セットのネゴシエーションに使用される。LCP の機能に接続完全性テストが含まれるため、リンク関連の問題の多くは LCP 異常として検出される。

レガシーサービス	SLP 対応していないネットワーク化サービス。プロキシ登録を作成して、レガシーサービスを SLP に登録することが可能。そうすると、SLP ベースのクライアントはレガシーサービスを検出できる (第 21 章を参照)。

索引

B

C

D

E

F

G

H

I

J

K

L

M

N

O

P

Q

R

S

T

U

V

W

X

あ

い

き

く

け

て

と

な

に

ひ

ふ

む

め

も

ゆ

よ

り

る

れ

ろ