IRIX® Admin: Networking and Mail
（日本語版）

ドキュメント番号 007-2860-006JP

編集協力者

執筆　Arthur Evans、Jeffrey B. Zurschmeide

改訂　Pam Sogard、Helen Vanderberg、Bob Bernard、Terry Schultz および Julie Boney

編集　Christina Cary、Cindy Kleinfeld および Susan Wilkening

制作　Amy Swenson および Glen Traefald

技術協力　Scott Henry、Carlin Otto、Kam Kashani、Chris Wagner、Paul Mielke、Robert Stephens、Joe Yetter、Gretchen Helms、John Schimmel、Robert Mende、Vernon Schryver、Michael Nelson および Landon Noll

イラスト　Dany Galgani

表紙デザインおよびイラスト　Rob Aguilar、Rikk Carey、Dean Hodgkinson、Erik Lindholm および Kay Maitz

このガイドにある新しい情報

『IRIX Admin: Networking and Mail』の改訂版であるこのガイドは、IRIX オペレーティング・システムの 6.5.7 リリースをサポートしています。

主な変更箇所

IRIX sendmail メール・システムは、標準の sendmail 設備を反映する目的で更新されました。`sendmail.cf` ファイルは現在では `sendmail.mc` ファイルから作成されます。`sendmail.cf` ファイルは `sendmail` の各コピーからリアルタイムに読込まれるため、sendmail.cf ファイルの必要性がなくなりました。これからは `sendmail.cf` ファイルを操作する必要はありません。そのかわり、ユーザは `sendmail.cf` ファイルにある定義、構成、コマンドの編集後に次のコマンドを使用して、さまざまな機能を使用可能にできます。

```
configmail mc2cf
```

第 9 章「IRIX sendmail」および付録 B「IRIX sendmail リファレンス」では、sendmail 設備を導入するためのマクロ定義や構成オプションについて説明があります。

改訂履歴

バージョン	説明
006	2000 年 2 月 IRIX 6.5.7 リリースに合わせ、情報を更新

目次

図一覧

表一覧

IRIX Admin マニュアル・セット

このマニュアルは、IRIX® Admin マニュアル・セットの中の 1 冊です。このマニュアルは、サーバ、マルチ・システム、およびファイル構造（ユーザのホーム・ディレクトリと作業用ディレクトリを除く）を管理するシステム管理者を対象としています。システムの保守を任されている方や、IRIX に関してエンド・ユーザ向けのマニュアルよりさらに専門的な知識が必要な場合は、このマニュアル・セットを参照してください。IRIX Admin マニュアルは、オンラインの IRIS InSight™ で参照できます。IRIX Admin マニュアル・セットは、次のマニュアルで構成されています。

- 『IRIX Admin: Software Installation and Licensing』— このマニュアルでは、IRIX 上で実行するソフトウェアのインストール方法とライセンス管理方法について説明します。IRIX は、Silicon Graphics 社の UNIX オペレーティング・システムです。このマニュアルでは、IRIX のインストール・ユーティリティのコマンド行インタフェースである Inst を使用してミニルート・インストールとライブ・インストールを行う手順について説明します。また、IRIX で実行する、特定のアプリケーションへのアクセスを制限するライセンス管理製品とそのマニュアルも紹介します。
- 『IRIX Admin: System Configuration and Operation』— このマニュアルでは、標準的なシステム管理方法について説明します。また、システム管理に関する作業として、オペレーティング・システムの設定、ユーザ・アカウント、ユーザ・プロセス、ディスク・リソースの管理、PROM モニタを介したシステムの操作、システム・パフォーマンスについても説明します。
- 『IRIX Admin: Disks and Filesystems』— このマニュアルでは、ディスク、ファイルシステム、論理ボリュームの各概念について説明します。また、SCSI ディスク、XFS ファイルシステム、EFS ファイルシステム、XLV 論理ボリューム、および帯域保証 I/O についてのシステム管理手順についても説明します。
- 『IRIX Admin: Networking and Mail』— このマニュアルでは、メール送信、UUCP、SLIP、PPP などを含むネットワーク・システムとメール・システムの計画、設定、使用、管理について説明します。
- 『IRIX Admin: Backup, Security, and Accounting』— このマニュアルでは、ファイルのバックアップとリストア、システムとネットワークのセキュリティ、ユーザ別のシステムの利用記録について説明します。
- 『IRIX Admin: Resource Administration』— このマニュアルでは、システム・リソース管理を紹介し、IRIX ジョブ制限および Miser のようないくつかの IRIX リソース管理機能の使用方法と管理方法について説明します。
- 『IRIX Admin: Peripheral Devices』— このマニュアルでは、端末、モデム、プリンタ、CD-ROM、テープ・ドライブなどの周辺デバイス用のソフトウェアの設定と管理方法について説明します。
- 『IRIX Admin: Selected Reference Pages』— このマニュアルは、InSight では利用できません。このマニュアルは、マン・ページ（マニュアル・ページ）をまとめたものです。システムがダウンしているときに必要となるコマンドについて説明します。各マン・ページでは、1 つのコマンドを説明していますが、関連のある複数のコマンドをまとめて説明したマン・ページもあります。オンラインのマン・ページにアクセスするには、`man` コマンドを使用します。

このマニュアルについて

このマニュアルでは、Silicon Graphics® ワークステーションおよびサーバから構成されるネットワークの設定方法と管理方法について説明します。TCP/IP をはじめ、SLIP、PPP、および UUCP を使用したネットワーク、また sendmail メール転送エージェントの設定について説明します。

Silicon Graphics ワークステーション上で動作するネットワーク通信の標準ソフトウェアは、カリフォルニア大学バークレイ校リリースの 4.3BSD UNIX® および Sun® Microsystems の RPC®（リモート・プロシージャ・コール）システムのネットワーク・ソフトウェアに基づいています。IRIX オペレーティング・システムは、インターネット・プロトコルと 4.3BSD UNIX ソケット・メカニズムを使用した UNIX ドメイン・ソケットを実装しています。また、このシステムではロー・ソケットを使用した下位のネットワーク媒体へのアクセスをサポートしています。

このマニュアルの内容

『IRIX Admin: Networking and Mail』は、次の章で構成されています。

- 第 1 章「ネットワーク製品について」では、Silicon Graphics のネットワーキング関連の標準ハードウェアと標準ソフトウェア、および標準ソフトウェアの設定（ファイル、デーモン、プロセス）について説明します。
- 第 2 章「ネットワークの計画」では、ネットワークの計画について説明します。インターネットのアドレス指定、`hosts` データベース・ファイル、ネットワーキング関連のアプリケーション、ネットワークのサブネット化、セキュリティ、異種ネットワークの問題などについて説明します。
- 第 3 章「ネットワークの設定」では、同種や異種のネットワークを設定するプロセス、ルータの設定、基本的なトラブルシューティングについて、例を挙げて説明します。
- 第 4 章「ネットワーク管理の概要」では、各種のネットワーク管理ツールについて説明します。バックアップの作成方法、パフォーマンスに影響を与える要因、および障害を特定化する方法について説明します。

- 第 5 章「SLIP と PPP」では、SLIP の特徴と機能について説明し、SLIP を使用した 2 台のステーションの接続方法について説明します。
- 第 6 章「BIND ネーム・サーバ」では、`named` として知られる BIND（Berkeley Internet Name Domain）サーバの概要について説明します。BIND の設定方法、その管理やトラブルシューティングについて説明します。
- 第 7 章「統一ネーム・サービス」では、統一ネーム・サーバである `nsd` の概要について説明します。また、このサーバとほかのネーム・サーバの関係や、UNS のトラブルシューティングに関する一般的な情報も説明します。
- 第 8 章「UUCP」では、TCP/IP と UUCP とを比較し、UUCP ネットワーク・ユーティリティの特徴と機能について説明します。また、UUCP の設定例を挙げ、一般的な UUCP のエラー・メッセージについて説明します。
- 第 9 章「IRIX sendmail」では、メール・システムである `sendmail` プログラム、およびエリアス・データベースの概要について説明します。また、メール・システムの計画におけるチェックリストや各種の `sendmail` の設定方法について説明します。
- 付録 A「BIND 標準リソース・レコードの形式」では、BIND 設定ファイルで使用されているすべての標準リソース・レコードの形式について説明します。
- 付録 B「IRIX sendmail リファレンス」では、sendmail スタンダードで実装した `sendmail` についてまとめます。

表記上の決まり

このマニュアルでは、次の表記法を用いています。

『 』	ほかのマニュアルのタイトルを表します。
「 」	本書のほかの章や節のタイトルを表します。
[ ]	メニュー名やボタン名などの UI（User Interface）を表します。
->	プルダウン・メニューの階層構造を表します。
< >	キーボードのジェネリック・キー（Ctrl、Shift、Alt など）を表します。 キーの操作方法として、次に例を示します。
<Enter>	**<Enter>** キーを押します。

<Alt>-h　　**<Alt>** キーを押しながら h キーを押します。

<Alt>-h c　　**<Alt>** キーを押しながら h キーを押した後、すぐに c キーのみを押します。

<Shift>-<Ctrl>-n
　　<Shift> キーを押しながら **<Ctrl>** キーと n キーを同時に押します。

<Ctrl>-x <Ctrl>-c
　　<Ctrl> キーを押しながら x キーを押した後、すぐに <Ctrl> キーを押しながら c キーを押します。

ほかのマニュアルへのリンクや、アプリケーションなどの実行可能な語句は赤く表示されます。

本書のほかの章、節、または図などへのリンクは青く表示されます。

参考文献

『Internet Request For Comment』は、インターネット・ネットワーク・インフォメーション・センター（InterNIC: Internet Network Information Center）から入手できます。連絡先は次のとおりです。

Network Solutions
Attn: InterNIC Registration Services
505 Huntmar Park Drive
Herndon, VA 22070
電話：1-800-444-4345 または 1-703-742-4777

『Internet Request For Comment』は、ftp.ds.internic.net など各サイトで anonymous `ftp` を使用して入手することも可能です。

Abitz, P, Liu, C., *DNS and BIND* (Sebastopol, CA: O'Reilly & Associates, Inc.).

Braden, R. "Requirements for Internet Hosts." *Internet Request For Comment 1112 (*1989).

Comer, D. E., *Internetworking with TCP/IP Volume 1*. (Englewood Cliffs, NJ: Prentice-Hall, 1995).

Costales, B. with Allman, E, *sendmail.* (Sebastopol, CA: O'Reilly & Associates, Inc., 1997).

Deering, S. "Host Extensions for IP Multicasting." *Internet Request For Comment 1112 (1989)*.

Everhart, C., Mamakos, L., Ullmann, R., Mockapetris, P. "New DNS RR Definitions." *Internet Request For Comment 1183* (1990).

Held, G., *LAN Management with SNMP and RMON*. (J. Wiley and Sons, 1996).

Huitema, C., *Routing in the Internet.* (Englewood Cliffs, NJ: Prentice-Hall, 1995).

Hunt, C., *TCP/IP Network Administration*. (Sebastopol, CA: O'Reilly & Associates, Inc., 1992).

Leinwand, A., Conroy, K.F., *Network Management - A Practical Perspective*. (Addison Wesley, 1996).

Lottor, M. "Domain Administrator's Guide." *Internet Request For Comment 1033 (*1987).

Lottor, M. "TCP Port Service Multiplexer (TCPMUX)." *Internet Request For Comment 1078 (*1988).

Loukides, M., *System Performance Tuning*. (Sebastopol, CA: O'Reilly & Associates, Inc., 1990).

Mockapetris, P. "DNS Encoding of Network Names and Other Types." *Internet Request For Comment 1101 (*1989).

Mockapetris, P. "Domain Names – Concept and Facilities." *Internet Request For Comment 1034 (*1987).

Mockapetris, P. "Domain Names – Implementation and Specification." *Internet Request For Comment 1035 (*1987).

Mogul, J., Postel, J. "Internet Standard Subnetting Procedure." *Internet Request for Comment 950* (1985).

Partridge, C. "Mail Routing and The Domain System." *Internet Request For Comment 974 (*1986).

Stahl, M. “Domain Administrator’s Guide.” *Internet Request For Comment 1032 (*1987).

Stern, H., *Managing NFS and NIS.* (Sebastopol, CA: O’Reilly & Associates, Inc., 1991).

Stevens, W. R., *TCP/IP Illustrated, Volume 1.* (Addison Wesley, 1996).

読者からのご意見

このマニュアルに記載されている技術情報の正確性や、マニュアルの内容または構成に対してご意見のある方は弊社までお寄せください。ご意見をお寄せの際は、必ずマニュアルのタイトルとドキュメント番号をご記入ください。(オンラインの場合、ドキュメント番号はマニュアルの最初のページにあります。印刷物の場合、ドキュメント番号は裏表紙にあります。)

弊社へのご連絡は、次のいずれかの方法で行うことができます。

- 次のアドレス宛てに電子メールを送信する

  `techpubs@sgi.com`

- Technical Publications Library の World Wide Web ページで「Feedback」オプションを使用する

  `http://techpubs.sgi.com`

- カスタマー サービスの担当者に連絡し、問題が SGI の問題追跡システムに記録されるように依頼する

- 次の住所宛てに手紙を書く

  Technical Publications
  SGI
  1600 Amphitheater Pkwy., M/S 535
  Mountain View, California 94043-1351, USA

- 技術出版部宛てにファックスを送信する（FAX 番号：+1 650 932 0801）

弊社ではお客様からのご意見を大切にしています。ご意見には迅速に回答いたします。

ネットワーク製品について

この章では、Silicon Graphics システムとともに提供されるネットワーク関連の標準ハードウェアおよび標準ソフトウェアについて説明します。Silicon Graphics システムのイーサネットおよびシリアル・ネットワークへの物理的な接続に関する説明、オプションのネットワーク・ハードウェア、およびネットワーク・デバイスのインタフェース名に関する情報を提供します。標準ネットワーク・ファイル、ディレクトリ、およびデーモンに関する説明、ネットワークの起動とシャットダウンのプロセスについても概説します。また、Silicon Graphics のオプションのネットワーク製品についても簡単に紹介します。

このマニュアルの第 2 章以降では、ネットワークの理論と動作に関する基本的な知識がすでにあることを前提としています。ネットワークに関する基本的な情報が必要な場合には、「このマニュアルについて」で紹介している参考文献を参照してください。

- ネットワーク・ハードウェアの概要。1 ページの「ネットワーク・ハードウェア」を参照してください。
- ネットワーク・インタフェース名の概要。4 ページの「コントローラのインタフェース名」を参照してください。
- ネットワーク・ソフトウェアの概要。5 ページの「ネットワーク・ソフトウェア」を参照してください。
- オプションのネットワーク・ソフトウェアのリスト。6 ページの「オプションのネットワーク製品」を参照してください。

ネットワーク・ハードウェア

すべての Silicon Graphics システムには、イーサネット・コントローラと 2 個のシリアル・ポートが標準搭載されています。シリアル・ポートや ISDN ポートを多数搭載した製品もあります。イーサネット・コントローラはボード型の場合と、集積チップの場合があります。このコントローラは、ネットワーク・ソフトウェアとネットワーク媒体間のインタフェースとして機能します。

システムがネットワークに接続されている場合には、さらに部品が必要になります。この説では、次のネットワーク・ハードウェアについて説明します。

- 「基本ネットワークの付属品」（2 ページ）
- 「ネットワーク・ハードウェアのオプション」（4 ページ）

基本ネットワークの付属品

イーサネット・コントローラをネットワークに接続するには、イーサネット・ケーブルが必要です。

図 1-1 に、イーサネット・ネットワークへのシステム（ネットワーク上ではステーションと呼びます）の接続方法を示します。

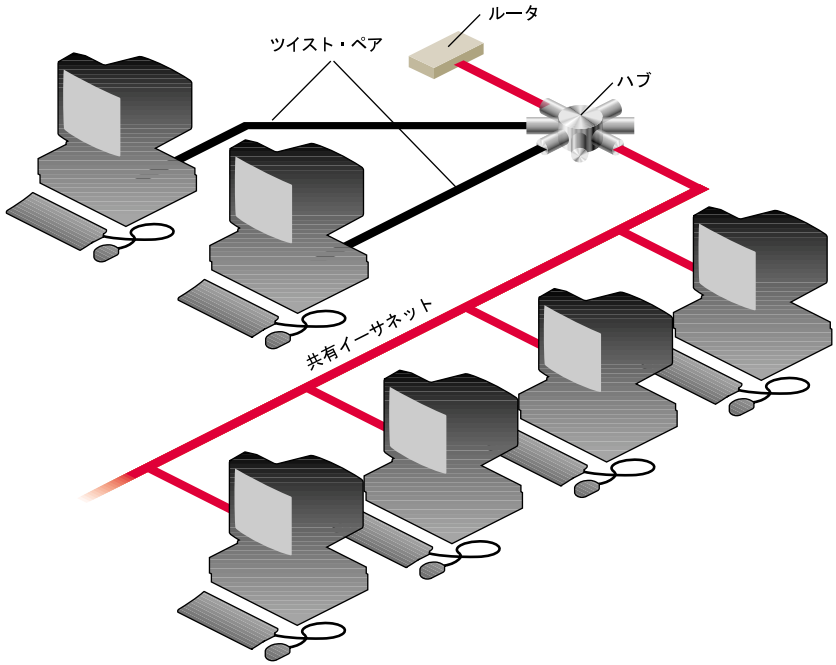

図 1-1　　イーサネット・ネットワークへの接続

Silicon Graphics システム上のシリアル・ポートは、シリアル回線ネットワークに接続するために使用します。シリアル回線ネットワークとは、シリアル回線とモデムによって接続されているシステムです。シリアル・ネットワークと接続するために、コンピュータに特別なハードウェアをインストールする必要はありません。

図 1-2 に、モデムを使用してシリアル・ネットワークに接続したシステムを示します。

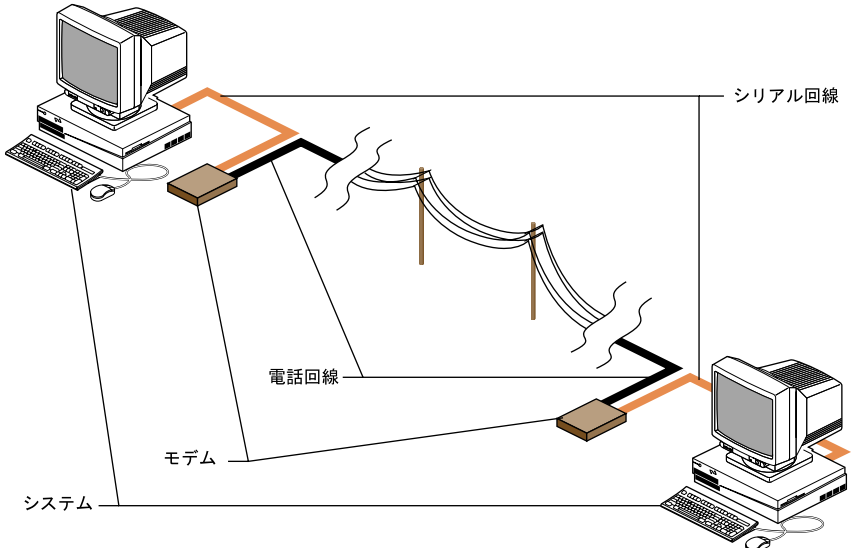

図 1-2 シリアル回線ネットワーク

ネットワーク・ハードウェアのオプション

イーサネットやシリアル回線ネットワークのためのハードウェアのほかに、オプションとして別のタイプのコントローラを Silicon Graphics のシステムにインストールすることができます。オプションのハードウェア製品の中にはユーザがインストールできるものもありますが、製品によっては Silicon Graphics 認定のシステム・サポート・エンジニアによるインストールが必要です。

Silicon Graphics から提供されるオプショナルなネットワーク製品は、FDDI、トークンリンク、X.25、SNA などのほかのタイプのネットワークをサポートします。ご使用のシステムで使用可能なネットワークのオプションについては、日本シリコングラフィックス株式会社のサポート部門にお問い合わせください。

コントローラのインタフェース名

システムをネットワークに接続する場合、ネットワークがシステムを認識する必要があります。物理的なボードまたはチップであるネットワーク・コントローラにより、システムは認識されます。インタフェースは、ソフトウェアのインタプリタであり、コントローラを処理します。インタフェース名は、ユーザーに分かりやすい名前になります。たとえば、ネットワーク管理ツールは、物理的なコントローラに関する情報を提供する場合に、インタフェース名を参照します。

コントローラを設定するには、システム上の各ネットワーク・コントローラに有効なインタフェース名がなければなりません。1 つのシステムが複数のコントローラを持つことができますが、コントローラごとに一意のインタフェース名を付ける必要があります。また、タイプの異なるコントローラを使用することもできます。この場合、タイプごとに固有のインタフェース名が付いています。ほとんどのネットワーク・ソフトウェアが、デフォルトで 4 つまでのネットワーク・インタフェース名をポートします。

システムにインストールされているインタフェースをリストする場合には、`hinv` コマンドを使用します。

```
% hinv -c network
Integral ISDN: Basic Rate Interface unit 0, revision 1.0
Integral Ethernet: ec0, version 1
```

この例では、イーサネット・コントローラのインタフェース名は「ec0」です。

ネットワーク・ソフトウェア

すべての Silicon Graphics システムに標準装備されているネットワーク・ソフトウェアは、インターネット・モデル規格とプロトコルに準拠しています。また、カリフォルニア大学バークレー校の 4.3BSD UNIX のネットワーク・ソフトウェア、および Sun Microsystems の RPC（リモート・プロシージャ・コール）システムに基づいています。IRIX オペレーティング・システムは、4.3BSD UNIX のソケット・メカニズムを使用したインターネット・プロトコルと UNIX ドメイン・ソケットを実装しています。このシステムは、ロー（raw）ソケットにより下位のネットワーク媒体へのアクセスもサポートしています。

すべての標準ネットワーク・ソフトウェアは、実行専用環境媒体（eoe および license_eoe）で提供されます。IRIS システム用の標準ネットワーク・ソフトウェアを表 1-1 に、IRIS システム用のオプションのネットワーク製品を表 1-2 にそれぞれ示します。

表 1-1　標準ネットワーク・ソフトウェア

標準ネットワーク・ソフトウェア	説明
TCP/IP	伝送制御プロトコル／インターネット・プロトコル
UUCP	UNIX-to-UNIX コピー・プログラム
sendmail	電子メール・プログラム
SLIP	シリアル回線インターネット・プロトコル
PPP	Point-to-Point プロトコル
BIND	Berkeley インターネット・ネーム・ドメイン
FLEXlm	フレキシブルなライセンス・サーバ
NCS	ネットワーク・コンピューティング・システム（NETLS のみをサポート）
RPC	リモート・プロシージャ・コール
gateway	インターネット・ゲートウェイ

オプションのネットワーク製品

Silicon Graphics では、オプショナルなネットワーク・ソフトウェアを各種取揃えているため、あらゆるベンダ間および媒体間の相互接続が可能です。表 1-2 は、入手可能なオプションのネットワーク製品を示しています。各製品に関する詳しい情報は、日本シリコングラフィックス株式会社のサポート部門にお問い合わせください。

表 1-2 オプションのネットワーク製品

オプションのネットワーク・ソフトウェア	製品説明
NFS™	ネットワーク・ファイル・システム（NFS）、ネットワーク・インフォメーション・システム（NIS、以前の YP）、ディスクレス・システムの各ソフトウェアを含む。
4DDN™	Silicon Graphics システムを Phase IV DECnet エンド・ノードとして機能できるようにする。
4DLT™	DECnet 端末サービス（LAT）を提供する。
FLEXlm License Server Developers Option	FLEXlm ライセンス・システム管理ツール、および FLEXlm をアプリケーションに統合するガイドラインを提供する。
NetVisualyzer™	グラフィカルなトラフィック・モニタ、診断、計測、およびパフォーマンス分析ツールを採用し、ネットワーク情報や統計量を視覚的に直感できる形式で提供する。
FDDI Visualyzer™	FDDI 環境のグラフィカル・インタフェースを提供する。
IRIX NetWorker™	ネットワーク上のシステムを自動的にバックアップするアプリケーション。バックアップしたすべてのファイルのオンライン・インデックスを維持する。

ネットワークの計画

この章では、物理的および論理的なネットワーク環境の計画について説明します。新たにネットワークを構築したり、既存のネットワークに統合する前にこの章をお読みください。

この章では、以下について説明します。

物理的なネットワーク計画

物理的なネットワークを計画するには、まず、ユーザの要求に適合するネットワーク媒体とトポロジはどうあるべきかを考えます。使用予定の製品の MAC（Medium Access Control）レベル、およびアプリケーション・レベルにおけるパフォーマンスを再検討し、適切なネットワーク媒体を選択します。ネットワークの規模（ステーションの数）も合わせて考慮します。ネットワークの規模によって選択する媒体のタイプとトポロジが変わってきます。ネットワークにいろいろなタイプの媒体が必要な場合は、それらの媒体を統合できる装置があるかどうか確認します。

次の章ではネットワークを計画する上で疑問を解決する情報を提供します。

- 物理的なネットワークはどのようになるか。8 ページの「リピータ、ブリッジ、ルータ、およびゲートウェイ」を参照してください。
- ネットワーク・マップを作るのか。10 ページの「広域ネットワーク」を参照してください。
- リピータ、ブリッジ、ルータ、ゲートウェイは必要か。8 ページの「リピータ、ブリッジ、ルータ、およびゲートウェイ」を参照してください。
- このネットワーク構成はユーザの要求に適合するか。9 ページの「ネットワークのパフォーマンス」を参照してください。
- パフォーマンス上の問題点はどこにあるのか。その問題点を解決または回避できるのか。9 ページの「ネットワークのパフォーマンス」を参照してください。

リピータ、ブリッジ、ルータ、およびゲートウェイ

使用する媒体、ステーションの数、ネットワーク、およびプロトコルにより、リピータ、ブリッジ、ルータ、またはゲートウェイが必要になる場合があります。次に、特定のネットワーク機能に必要なデバイスについて簡単に説明します。

リピータ　電子信号を再生、増幅するデバイスです。このデバイスの目的は、ネットワークの物理長を延長することです。

ブリッジ　異なるハードウェアと媒体間で伝送される MAC 層フレームを認識するデバイスです。このデバイスの目的は、ネットワーク媒体間の相違を解決することです。つまり、このデバイスを使用することにより、さまざまなタイプの媒体（イーサネット、ファイバ・ケーブル、シリアル回線など）をネットワークに接続できます。また、同じタイプの媒体をグループ分けし、低ネットワーク・トラフィックのためにグループを分離します。

ルータ　異なるネットワーク間でネットワーク層パケットを認識し、それを渡すデバイスです。このデバイスの目的は、ネットワークからほかのネットワークへの物理的ルートと論理的ルートを提供することです。

ゲートウェイ　あるステーションからほかのステーションへプロトコルを変換するデバイスです。このデバイスの目的は、異なるネットワーク・プロトコルを使用しているステーション間で通信できるようにすることです。

メモ：ルータとゲートウェイという用語はしばしば混同して使われます。これは、この 2 つの用語の技術的定義が曖昧なため、使用するデバイスの機能を十分に確認しておく必要があります。

各デバイスの機能は 1 つだけとはかぎりません。たとえば、ゲートウェイを、ルータとして設定するとルータの機能も実行します。表2-1は各ネットワーク・デバイスの特徴をまとめたものです。

表 2-1 ネットワーク・デバイスの特徴

デバイス名	媒体	プロトコル	LAN	目的
リピータ	同	同	同	ネットワークの物理長を延長
ブリッジ	異／同	異／同	同／異	ネットワーク媒体間の相違を吸収
ルータ	同／異	同	異	ネットワーク間に物理的ルートと論理的ルートを提供
ゲートウェイ	同／異	異	同／異	異なるネットワーク・プロトコルを使用しているステーション間の通信

ネットワークのパフォーマンス

パフォーマンスの問題には、適切な計画を立てれば回避できるものがあります。パフォーマンスの問題は、採用した媒体、トポロジ、ネットワーク・デバイスの数、コントローラ・ボード、ネットワークの設計などによって発生します。

媒体の選択　選択した媒体の容量がネットワークの規模やデータ伝送の特徴（データ量やトラフィック量の大小）に適切であるかどうか確認します。たとえば、イーサネットの容量範囲は、使用するイーサネット・ケーブルのタイプ（10base5、10base2、10baseT または 100baseT）によって変わってきます。また、媒体のタイプもデータ劣化の要因になります。たとえば、10baseT はカテゴリ 3 の非遮蔽のツイストペア線なので、10base5 よりも環境条件に敏感です。100baseT はカテゴリ 5 の非遮蔽のツイストペア線になります。静電放電がよく起こる製造などの環境でネットワークを計画している場合は、この点も考慮します。

デバイス数　ネットワーク・デバイスがネットワークのパフォーマンスを低下させる原因になることがあります。リピータは必要な場合にだけ使用します。デバイスを 1 つ追加するたびに、ネットワークの負荷が増大します。

コントローラの選択

媒体に対して最も効率的なコントローラを選択します。たとえば、Silicon Graphics 社では、標準イーサネット・コントローラを提供しています。オプションの Efast™ カードは、より多くのプロトコルをハードウェアで処理することができるので、ステーションの CPU をほかの処理に解放することができます。

ネットワークの設計

ネットワークの設定を開始する前に、ネットワークの設計について十分検討します。可能であれば、頻繁に対話を行う部門を分離し、ルータのトラフィックを低減します。また、ネットワーク間のトラフィックが重い場合は、専用のルータを使用します。

広域ネットワーク

ローカル・エリア・ネットワークを構築するための多くのオプションのほかに、ローカル・エリア・ネットワークを広域ネットワークに接続するための方法がいくつかあります。このようなシステムを使用すると、さまざまな場所に点在するローカル・エリア・ネットワークを接続したり、さまざまな場所から同じネットワークにアクセスしたり、現在使用しているネットワークを外部のネットワークと接続することができます。ここでは、さまざまなシステムについて説明します。

- 「シリアル回線インターネット・プロトコル（SLIP: Serial Line Internet Protocol）」（13 ページ）
- 「Point-to-Point プロトコル（PPP: Point-to-Point Protocol）」（13 ページ）
- 「Unix-to-Unix コピー・プログラム（UUCP: UNIX-to-UNIX Copy Program）」（13 ページ）
- 「統合デジタル通信網（ISDN: Integrated Services Digital Network）」（13 ページ）
- 「インターネット・ゲートウェイ」（13 ページ）
- 「ハイパフォーマンスな広域ネットワーク」（14 ページ）

図 2-1 に、さまざまな広域接続を組合わせた大きな異種型ネットワークの構築例を示します。

シリアル回線インターネット・プロトコル（SLIP: Serial Line Internet Protocol）と Point-to-Point プロトコル（PPP: Point-to-Point Protocol）の 2 つのシステムは、シリアル電話回線でインターネット・プロトコル（IP: Internet Protocol）パケットを転送します。このため、SLIP ユーザと PPP ユーザは、ローカル・エリア・ネットワークの場合と同じ方法でネットワーク資源にアクセスできます。また、PPP は、統合サービス・デジタル・ネットワーク（ISDN: Integrated Services Digital Network）でも使用できます。ISDN では高速デジタル電話回線を使用しており、モデムを介して接続した場合よりも高い処理能力を実現します。

また、UNIX-to-UNIX コピー・プログラム（UUCP: Unix-to-Unix Copy Program）も提供されています。これは従来から使用されているシステムであり、主にシリアル回線を介してバッチ・モードで情報（ネットワーク・ニュースや電子メールなど）をやり取りする場合に使います。

専用のハードウェアを使用すると、よりパフォーマンスの高いネットワーク接続を実現できます。通常、このような接続では電話会社からリースした専用回線、または電話会社のパケット交換ネットワークを使用します。

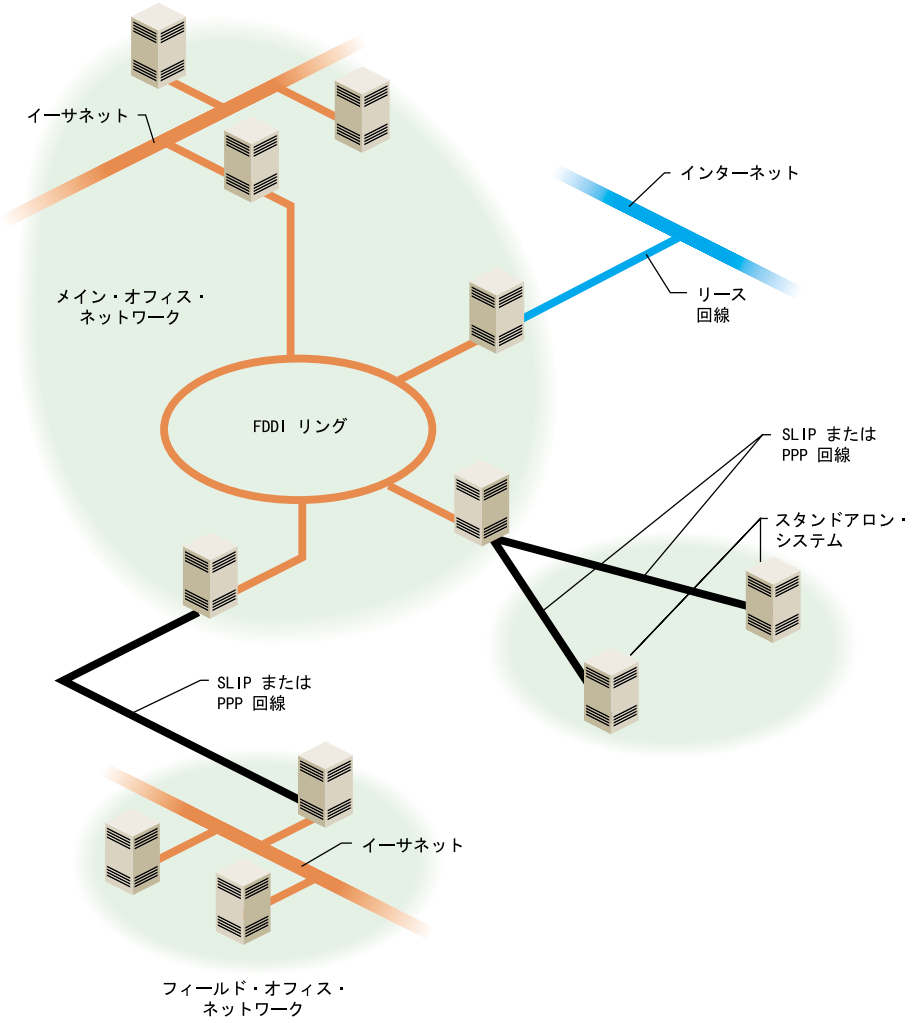

図 2-1　広域接続から構成される異種型ネットワーク

シリアル回線インターネット・プロトコル（SLIP: Serial Line Internet Protocol）

SLIP では、シリアル・ケーブルまたは電話回線を介して同時に複数の処理を実行できます。SLIP により、ネットワークのユーザはシリアル・ケーブルまたはモデムを介して TCP/IP ベースのアプリケーションを自由に使うことができます。

ネットワークの計画で費用と距離が問題点となる場合は、SLIP ネットワークを設定します。

Point-to-Pointプロトコル（PPP: Point-to-Point Protocol）

PPP の特徴は SLIP と似ています。PPP は、システムが LAN によってリモート・ホストに接続されているのと同様のネットワーク接続を提供します。PPP では、複数のプロセスと TCP/IP ベースのアプリケーションがサポートされています。

Unix-to-Unixコピー・プログラム（UUCP: UNIX-to-UNIX Copy Program）

基本ネットワーク・ユーティリティ（BNU: Basic Networking Utilities）とも呼ばれる UUCP は、UNIX オペレーティング・システム（IRIX など）を使用しているステーションがシリアル回線を介して相互に通信する際に使用するユーティリティです。このユーティリティは、コンピュータ間のファイル・コピーに使用する機能や、リモート・ログインやコマンドの実行などに使用する機能を提供します。

モデムと電話回線を介して長距離通信用に UUCP を設定することもできます。これは通常、電子メールやネットワーク・ニュースの配布に使用します。

統合デジタル通信網（ISDN: Integrated Services Digital Network）

ISDN は、高速デジタル電話回線を介してシステムを接続します。ISDN では、1 秒間に最高 128Kb の処理能力を実現できます。これは、通常のモデムを介した接続と比べ、数倍の速度です。ただし、ISDN サービスは費用が高くなる上、すべてのプラットフォームやすべてのエリアで使用できるわけではありません。ISDN については、『ISDN User's Guide』を参照してください。

インターネット・ゲートウェイ

インターネット・ゲートウェイは、インターネットに接続するためのサーバ・プロセスを提供し、あらゆるネーム・サービスを設定するために使用されます。これは、ルータとしても機能し、設定プロセスを通してユーザを補助する独自のヘルプ・スクリーン・システムを持っています。

ハイパフォーマンスな広域ネットワーク

モデム・リンクや ISDN で SLIP や PPP を使用するよりも高いパフォーマンスが必要な場合は、別のオプションを利用できます。フレーム・リレー・ネットワークや専用回線サービスなど、56 Kb（1 秒間に 56 Kb）から T1（1 秒間に 1.5 Mb）、さらに T3（1 秒間に最高 45Mb）までのオプションがあります。このようなサービスが必要な場合は、各インターネット・サービス・プロバイダの価格とサービスを比較、検討してください。

インターネット・プロトコル・アドレス

ネットワークのインタフェースには、ネットワーク上で一意な IP アドレスが必要です。個々のサイトに対するインターネット・アドレスは、インターネット・ネットワーク・インフォメーション・センター（InterNIC: Network Information Centers）が割当てます。たとえば、企業 A がインターネット・アドレスを申請すると、InterNIC は企業 A のサイト全体にインターネット・アドレスを提供します。このインターネット・アドレスのステーション ID の割当てと管理は、企業 A の担当部署が一括して行います。

インターネット・アドレスは、各ステーション上、または NIS や BIND などの共有ネットワーク・データベース上で管理します。データベースのタイプについては、27 ページの「名前とアドレスの対応付け」を参照してください。ステーションが通信するためには、有効なインターネット・アドレスが正しいデータベースに登録されていなければなりません。IRIX ステーションでは、`/etc/hosts` ファイルがホスト名とそのアドレスの標準データベースになっています。

このセクションでは次について説明します。これは、次の計画上の疑問に対する回答となります。

- IP アドレスとは何か。15 ページの「インターネット・プロトコル（IP）バージョン 4 のアドレス形式」を参照してください。
- サイトに指定する有効なインターネット・アドレスはどのような方法で取得できるのか。17 ページの「ネットワーク番号の取得」を参照してください。
- インターネット・アドレスを取得する前にどのような情報が必要か。18 ページの「インターネット・アドレスの取得に必要な情報」を参照してください。

インターネット・プロトコル（IP）バージョン 4 のアドレス形式

IP アドレスは、ネットワーク・ソフトウェアがネットワーク上のシステムを識別するのに使用する 32 ビットの数字です。このアドレスは、読みやすいようにドットで 4 つのセグメントに区切られています。たとえば、150.166.248.17 と表します。ネットワークが正しく機能するためには、IP ネットワーク上のシステムにはそれぞれ一意な IP アドレスが必要です。複数のネットワーク・インタフェースを備えたシステムの場合は、インタフェースごとに一意なIPアドレスが必要です。

メモ：システムのイーサネット・アドレスとは異なり、IP アドレスはネットワークおよびネットワーク・システム管理者が決定します。

概念的には、32 ビットの各 IP アドレスは、ネットワークを表す番号とシステム自体を表す番号が対になっています。ネットワークを表す上位ビットによって、4 種類（A から D）のアドレス・クラスが決まります。

- クラス A のアドレスは 0 で始まり、7 ビットでネットワーク番号を表し、24 ビットでホスト番号を表します。
- クラス B のアドレスは 10 で始まり、14 ビットでネットワーク番号を表し、16 ビットでホスト番号を表します。
- クラス C のアドレスは 110 で始まり、21 ビットでネットワーク番号を表し、8 ビットでホスト番号を表します。
- クラス D のアドレスは 1110 で始まり、特定のネットワーク・サイト内で使用される特殊なマルチキャスト・アドレスです。

いずれの場合も、ホスト番号 0 および 255 は予約されているので実際のシステムでは使用できません。

図 2-2 に各クラスのインターネット・アドレスの形式を示します。

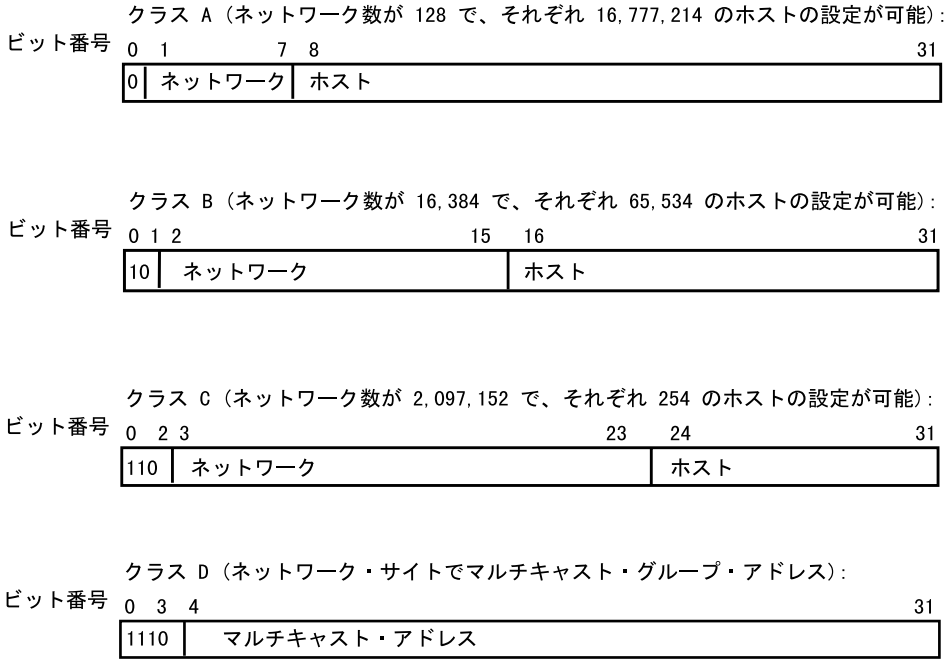

図 2-2 インターネット・プロトコル（IP）アドレスの形式

インターネット・アドレスをわかりやすく表示するため、32 ビットの番号を 10 進表記にして、ドットで区切られた 4 つの 10 進数に分割しています。

たとえば、IP アドレス 128.74.41.123 を 2 進数で表現すると次のようになります。

```
10000000| 01001010| 00101001| 01111011
```

または

```
128 | 74 | 41 | 123
```

クラス A、B、C の IP アドレスを正しいドット表記で表すと、次の形式になります。

クラス A -- 001.*hhh.hhh.hhh* から 126.*hhh.hhh.hhh*

クラス B -- 128.001.*hhh.hhh* から 191.254.*hhh.hhh*

クラス C -- 192.000.001.*hhh* から 223.255.254.*hhh*

メモ：*hhh* はローカル・システム、その前の番号はネットワークを表します。

通常、ネットワークはネットワーク番号で識別されます。ネットワーク番号とは、IP アドレスの中のホスト番号を除いた部分です。たとえば、150.166 はクラス B のネットワーク、192.26.80 はクラス C のネットワークを表します。

ネットワークをインターネットに接続する場合、17 ページの「ネットワーク番号の取得」で説明されているように、一意なネットワーク番号を取得する必要があります。また、ネットワーク上のすべてのシステムに対してネットワークから IP アドレスを割当てます。

既存のネットワークにマシンを追加する場合は、そのネットワークからIPアドレスを割当てます。

ネットワーク番号の取得

ネットワークを設定する前に、インターネットのネットワーク番号を取得します。ネットワーク番号の割当ては、NIC（日本国内では JPNIC）と呼ばれる組織が管理しています。22 ページの「ネットワーク・インフォメーション・センター（NIC: Network Information Centers）」を参照してください。ネットワークをインターネットに接続せずに、切り離して使用する場合は、理論的には任意のアドレスを使用できます。ただし、ネットワークをインターネットに接続する予定がある場合は、有効なネットワーク番号を取得しておく必要があります。ネットワーク番号を申請する前に、ネットワークに接続するシステムの数など組織の現時点のニーズと今後 5 年間で予想されるネットワークの拡張を確認しておきます。

ネットワーク番号を取得する方法はいくつかあります。通常、インターネットのサービス・プロバイダを介してインターネットに接続している場合は、サービス・プロバイダが NIC から割当てられたアドレス空間の一部を割当てます。

InterNIC では、ネットワークのサービス・プロバイダにネットワーク番号を申請することを薦めています。ネットワークのサービス・プロバイダからネットワーク番号を取得できない場合は、さらに上位のプロバイダに申請し、最終的な手段としては、NIC に直接申請します。詳細については、22 ページの「ネットワーク・インフォメーション・センター（NIC: Network Information Centers）」を参照してください。

インターネット・アドレスの取得に必要な情報

インターネットのネットワーク・アドレスを申請するには、通常 NIC に次の情報を提出します。

- 運用責任者の連絡先（POC）。これは、ネットワークの計画および管理責任者の連絡先です。この担当者の氏名、役職、メール・アドレス、電話番号を明記します。
- 技術担当者の連絡先（POC）。これは、ネットワークの技術サポートの責任者の連絡先です。この担当者の氏名、役職、メール・アドレス、電話番号を明記します。
- 組織名と住所。
- ネットワーク名（12 文字以内）。
- ネットワークの場所と組織名。
- ネットワーク・ドキュメント・プランの名前と場所。
- ゲートウェイに関する情報（接続性、ハードウェア、ソフトウェア、アドレス）。
- 現在および 1 年以内のネットワークのおおよその規模（ホストとサブネットの数）。
- ネットワークのタイプ（研究機関、教育機関、軍以外の政府機関、商業組織）。

組織にすでにネットワーク番号が割当てられている場合は、新しいネットワーク番号が本当に必要であることを証明するため、既存のネットワーク番号をどのように使用しているか NIC に報告する場合もあります。

クラス C のネットワーク番号を 16 個以上申請する場合は、InterNIC にネットワーク・トポロジを提出する必要があります。また、クラス C またはクラス B のネットワーク番号を 256 個以上申請する場合は、ネットワークの構成図を提出する必要があります。

ドメイン名

サイトでインターネットを利用したり、インターネットを利用しているほかのサイトと電子メールを交換する場合は、NIC にドメイン名を登録する必要があります。ドメイン名によって、それぞれの組織が一意に識別されます。たとえば、Silicon Graphics のドメイン名は sgi.com です。

次では、ドメイン名とサブドメインについて説明します。

- ドメインをどのように登録するか。19 ページの「ドメイン名の取得」を参照してください。
- ドメインをさらにサブドメインに分ける必要がある場合は、20 ページの「サブドメイン」を参照してください。

インターネットではドメイン・ネーム・サービス（DNS: Domain Name Service）を使って IP アドレスにドメイン名を対応付けています。このため、組織内で DNS を使用していない場合でも、ネットワークをインターネットに接続するためには、インターネットに DNS ネーム・サーバを設定する必要があります。少なくても、プライマリ・サーバとセカンダリ・サーバの 2 つのネーム・サーバが必要です。信頼性を高めるため、セカンダリ・サーバはプライマリ・サーバとは違うゲートウェイを介してインターネットに接続します。この場合、ほとんどの組織はインターネットへのゲートウェイを複数所有できるほど規模が大きくないため、別の組織と協力して互いに二次的なネーム・サービスを提供し合います。

インターネット・サービス・プロバイダを介してインターネットに接続している場合は、そのサービス・プロバイダが組織にネーム・サービスを提供してくれます。また、組織側でプライマリ・サーバを使用できる場合は、二次的なネーム・サービスを提供してくれる相手を探してくれます。

ドメイン名の取得

ネットワーク番号と同じように、ドメイン名の登録も NIC で管理しています。ドメイン名を所有するのに料金がかかる場合もあります。たとえば、InterNIC では現在、最初の 2 年間は 100 ドル、その後 1 年ごとにドメインの管理費として 50 ドルが必要になります。

ドメイン名は NIC で登録します。NIC への連絡方法については、22 ページの「ネットワーク・インフォメーション・センター（NIC: Network Information Centers）」を参照してください。ドメインを登録する場合は、*IN-ADDR* ドメインと呼ばれるリバース・ドメイン（*reverse domain*）

も登録する必要があります。リバース・ドメインは、IP アドレスからドメイン名に対応付けるのに使用します。

通常は、インターネットのサービス・プロバイダがドメインの登録を有料で代行します。

サブドメイン

ドメイン名を登録した後は、サブドメインを自由に設定できます。大きな組織で DNS を使用する場合は、サブドメインを設定しておくと便利です。DNS でサブドメインを使用すると、管理業務を分散化できます。

たとえば、salad.com という企業の支社がギルロイ（Gilroy）とパリ（Paris）にあるとします。この場合、サブドメインとして gilroy.salad.com と paris.salad.com を使います。

インターネットとの接続

システムやネットワークをインターネットに接続する場合、ローカルな範囲内で呼出し可能なインターネット・ゲートウェイが世界中のどこにでもあります。次に、インターネットへの接続について説明します。利用者ごとに状況は異なり、各地のサービス・プロバイダの提供するサービスや設備も違ってきます。したがって、インターネットを有効に利用するためには若干の事前調査が必要です。

- インターネットをどの程度利用するか。21 ページの「インターネットに接続する前に」を参照してください。
- インターネットにどのように接続するか。22 ページの「ネットワーク・インフォメーション・センター（NIC: Network Information Centers）」を参照してください。
- インターネット上でどのような情報を利用するか。24 ページの「オンラインで利用可能な情報源」を参照してください。

インターネットに接続する前に

インターネット接続を契約する前に、どのレベルのサービスが必要か考えます。基本的な電子メール、ニュース、ファイル転送機能だけが必要な個人ユーザの場合は、専用ネットワーク・ケーブルを家庭に導入するのは経済的ではありません。個人ユーザがインターネットにアクセスする場合は、ネットワーク・プロバイダと契約し、そのプロバイダのシステム（インターネットに既に接続されているシステム）上でアカウントを設定してもらいます。通常、このようなシステムにはモデムを介してアクセスします。

企業としてインターネットに接続する場合は、専用回線の帯域幅と、それに付随するハードウェアが必要となります。また、サイトの運用に関する次のような数多くの管理上の問題も考慮に入れます。

- 費用の分析と予算
- ドメインの設定
- IP アドレスの割当て
- サイト方針の確立
- サイト・セキュリティの確立
- ネットワーク・サービス（DNS、NIS、電子メールなど）の管理

プロバイダによって、提供するネットワーク接続サービスは異なります。これらのプロバイダをよく調べ、妥当な価格で必要なサービスを提供してくれる業者を選びます。

個人ユーザとしてサービス・プロバイダのマシン上のアカウントを使用する場合は、プロバイダが大部分の管理作業を引受けてくれるので、ほかのことを心配せずに単にインターネットへのアクセスだけを楽しめます。

より広範なサービスを希望する場合は、ほとんどのプロバイダが専用のモデムと電話回線を提供するか、モデムなどのネットワーク接続を介して SLIP、PPP、または UUCP のいずれかを使用したネットワーク専用サービスを提供しています。

ネットワークをインターネットに接続するまでには、ネーム・サーバの設定、ネットワーク番号の取得、ドメイン名の登録などいくつかの手続きが必要です。インターネットのサービス・プロバイダの多くが、この代行サービスを有料で提供しています。

企業としてインターネットへのアクセスを確立する場合は、前述の管理上の問題をよく検討します。収集した情報に基づき、組織のニーズと現状を考慮しながら、サイトの計画を立てます。最も有用な情報源の 1 つはインターネットそのものです。まず各地域のネットワーク・プロバイダから個人のアカウントを取得します。この個人アカウントを利用してインターネット上にサイトを設定するための膨大な量の情報を収集します。

ネットワーク・インフォメーション・センター（NIC: Network Information Centers）

サイトをインターネットに接続するには、NIC に申請する必要があります。ネットワーク番号やドメイン名の割当ては NIC が管理しています。表 2-2 に主な NIC を示します。

表 2-2 各地域の NIC

地域	NIC
アジアおよび環太平洋諸国	APNIC（Asia Pacific Network Information Center）
欧州	Reseaux IP Europeens Network Coordination Centre（RIPE NCC）
米国	InterNIC（Internet Network Information Center）
その他	InterNIC

IP アドレス取得の手続きやドメイン名登録の手続きはそれぞれの地域で異なるため、詳細についてはそれぞれの地域の NIC に問い合わせてください。

InterNIC（Internet Network Information Center）

以前は、NIC には InterNIC しかありませんでした。現在、InterNIC は北米／南米の大半と NIC が設立されていないその他の地域を担当し、NIC の中心的な役割を果たしています。InterNIC では莫大な量の資料が保存され、これらには WWW や FTP を利用したり、自動応答メール・サーバによる電子メールを利用してアクセスすることができます。一部の国（カナダやブラジルなど）に対する登録権限はその国の NIC に任されています。そのような国の NIC への連絡方法については、InterNIC から情報を入手できます。

Network Solutions
Attn: InterNIC Registration Services
505 Huntmar Park Drive
Herndon, VA 22070
電話番号：1-800-444-4345 または 1-703-742-4777
電子メール：question@internic.net（一般的な問い合わせ）
電子メール：hostmaster@internic.net（登録サービス）
WWW: http://www.internic.net/
FTP: ftp.ds.internic.net（完全な RFC など）
FTP: rs.internic.net （registration information）
電子メール・サーバ：mailserv@rs.internic.net（「HELP」というサブジェクトでメッセージを送信）

Reseaux IP Europeens

RIPE（Reseaux IP Europeens）は、欧州のサイトに登録サービスを提供する NIC です。ここでも InterNIC の FYI ドキュメントをはじめ、欧州でホストまたはネットワークを登録するための手続き手順など、莫大な量の情報を保管しています。

RIPE Network Coordination Centre
Kruislaan 409
NL-1098 SJ Amsterdam
The Netherlands
電話番号：+31 20 592 5065
Fax: +31 20 592 5090
電子メール：ncc@ripe.net
WWW: http://www.ripe.net/
FTP: ftp.ripe.net

Asia Pacific Network Information Center

APNIC（Asia Pacific Network Information Center）は、アジアと環太平洋地域のネットワーク情報を管理しています。一部の国に対する登録権限はその国の NIC に任されています。そのような国への連絡方法については、APNIC から情報を入手できます。

Asia Pacific Network Information Center
c/o United Nations University
53-70 Jingumae 5-chome
Shibuya-ku, Tokyo 150
Japan
電話番号：+81-3-5467-7014
Fax: +81-3-5276-6239
電子メール：info@apnic.net
WWW: http://www.apnic.net/
FTP: archive.apnic.net

オンラインで利用可能な情報源

個人アカウントまたはその他の方法でインターネットにアクセスすることによって、サイトの設定に必要な情報を収集することができます。

通常、個人アカウントを設定する場合でも、プロバイダはインターネットの基本的な使い方を記載したマニュアルを提供しています。WWW（World Wide Web）とファイル転送プロトコル（FTP: File Transfer Protocol）を使用すると、インターネットとの接続など多くのテーマに関する膨大な量の情報にアクセスできます。FTP の使い方については、26 ページの「anonymous FTP によるファイルの検索」を参照してください。Web へのアクセス方法は、使用している Web ブラウザによって異なりますが、大半の Web ブラウザはオンライン・ヘルプを備えていますので、それを参照してください。

次では、インターネットの使い方について簡単に説明します。

- インターネットに接続するときのネットワーク・インフォメーション・センター（NIC: Network Information Centers）の利用方法。25 ページの「各地域の NIC」を参照してください。
- インターネット・プロバイダの見つけかた。25 ページの「Internet Society」を参照してください。
- サイトにアクセスした後、必要なファイルをシステムに取込む方法。26 ページの「anonymous FTP によるファイルの検索」を参照してください。

各地域の NIC

NIC は、インターネットに接続に関する情報を保管しています。ネットワーク番号の申請方法やドメイン名の登録方法だけでなく、その地域のサービス・プロバイダのリストを保管している場合もあります。NIC のほとんどは、WWW や FTP を使ってこのような情報にアクセスできるようにしています。主な NIC の WWW および FTP アドレスについては、22 ページの「ネットワーク・インフォメーション・センター（NIC: Network Information Centers）」を参照してください。

InterNIC では、FYI という情報冊子を発行しています。その中でも特に『Connecting to the Internet—What Connecting Institutions Should Anticipate』というタイトルの FYI 16 は大変役に立ちます。これは、主に米国の教育機関を対象にしたものですが、インターネット上にサイトを作成するときに必要な情報が記載されています。この FYI は、WWW または FTP を使って InterNIC および RIPE から入手することができます。

Internet Society

Internet Society は、インターネットを世界的に展開し、発展させていくことを目的とした国際的な非政府組織です。この組織も、インターネット・サービス・プロバイダのリストや世界中のネットワークのサービス・プロバイダのリストなど、各種オンライン情報を提供しています。この情報は WWW で入手できます。また、サブセットは anonymous FTP で利用できます。

WWW: http://www.isoc.org/
FTP: ftp.isoc.org

anonymous FTP によるファイルの検索

anonymous（匿名）FTP は、インターネット上のコンピュータに接続し、公開されているファイルにアクセスする従来の方法です。ソフトウェアやさまざまな情報を配布するために anonymousFTP アカウントを提供しているサイトもあります。以下に `ftp` コマンドの使い方を簡単に説明します。リモート・ホストに接続するには、次のようにコマンド行でホスト名を指定します。

```
ftp ftp.ds.internic.net
```

ftp でリモート・システムに接続すると、ftp はログイン名を要求してきます。ログイン名としては「anonymous」を入力します。

```
Connected to ftp.ds.internic.net.
Name （ftp.ds.internic.net:guest）: anonymous
331 Guest login ok, send ident as password.
Password:
```

ほとんどのシステムに対しては、パスワードとしてユーザ ID を指定します。このパスワードでログインできない場合は、通常「guest」をパスワードとして入力します。

```
230 Guest login ok, access restrictions apply.
Remote system type is UNIX.
Using binary mode to transfer files.
ftp>
```

接続とログインが完了したら、`ftp` の `cd` と `ls` コマンドを使ってリモート・システム上のファイルを検索できます。リモート・システムからファイルをコピーするには、`get` コマンドを使用します。`get` コマンドは、リモート・システムからローカル・システムにファイルを 1 つコピーします。リモート・システムから複数のファイルをコピーする場合は、`mget` コマンドを使用します。

```
ftp> cd fyi
250 CWD command successful.
ftp> get fyi6.txt
local: fyi6.txt remote: fyi6.txt
200 PORT command successful.
150 Opening BINARY mode data connection for fyi6.txt （3459 bytes）.
226 Transfer complete.
3459 bytes received in 0.46 seconds （7.34 Kbytes/s）
ftp>
```

名前とアドレスの対応付け

IP アドレスは覚えにくいため、通常は名前と対応付けます。IP アドレスを 1 つしか持たないマシンの場合、IP アドレスに対応する名前はホスト名とドメイン名から構成されます。たとえば、ドメイン salad.com 上にあるマシン fruit は、通常 fruit.salad.com という名前になります。ここでは、このような名前をネットワーク接続名と呼びます。

ネットワーク接続名は、通常はマシンのホスト名と対応しているのでホスト名「Hostname」と呼ばれますが、紛らわしい場合があります。実際のホスト名は `/etc/sys_id` ファイルで定義されています。デフォルトでは、このホスト名はマシンのプライマリ・インタフェースのネットワーク接続名として認識されますが、この動作は変更することができます。ネットワーク・インタフェースが複数あるマシンでは、そのインタフェースに対応するネットワーク接続名がそれぞれ割当てられます。これらの接続名にはそれぞれホスト名が含まれます。たとえば、ホスト fruit がドメイン salad.com の 2 つのネットワークを結ぶゲートウェイとして機能する場合は、次の 2 つの名前を使用します。

```
fruit.salad.com
gate-fruit.salad.com
```

ネットワーク接続名をIPアドレスに対応付けるプロセスは、ホスト名の検索（*hostname resolution*）と呼ばれます。ホスト名の検索には、いくつかのシステムが利用できます。ホスト名に対応する IP アドレスを問い合わせるには、ローカル・データベース（`/etc/hosts` データベース）を使用する方法、およびネットワーク・インフォメーション・サービス（NIS: Network Information Service）やドメイン・ネーム・システム（DNS: Domain Name System）を使用してネットワーク上のサーバから情報を得る方法があります。次に、それぞれの方法の長所と短所について説明します。

- 「/etc/hosts データベース」（27 ページ）
- 「ドメイン・ネーム・システム（DNS: Domain Name System）」（28 ページ）
- 「ネットワーク・インフォメーション・サービス（NIS: Network Information Service）」（28 ページ）

/etc/hosts データベース

`/etc/hosts` データベースは、テキスト・エディタで編集できる ASCII ファイルです。このファイルには、IP アドレスとネットワーク接続名を指定するテキストが記述されています。

ネットワークに接続されているシステムの数が少なく、同じ管理下にある場合は、/etc/hosts データベースを管理するのは簡単です。1 台のステーションでマスター・コピーを作成し、そのファイルにデータを追加したり、ファイルからデータを削除します。その後、rcp または rdist コマンドを使用して、そのファイルをネットワーク上の別のステーションにコピーします。

大規模なネットワークの場合、すべてのステーションで同一バージョンの /etc/hosts データベースを維持するのは容易なことではありません。そこで、NIS や BIND のネーム・サーバを利用して集中型のホスト・データベースを構築し、簡単に管理できるようにします。

ドメイン・ネーム・システム（DNS: Domain Name System）

インターネットでは DNS を使用して名前を IP アドレスに対応付けています。DNS サーバの中で最も一般的なのが Berkeley インターネット・ネーム・ドメイン（BIND: Berkeley Internet Name Domain）です。ネットワークをインターネットに接続するには、プライマリ・サーバとセカンダリ・サーバの少なくとも 2 つの DNS ネーム・サーバが必要です。サイトでこの 2 つのサーバを運用せずに、インターネットのサービス・プロバイダのネーム・サーバを利用することもできます。

BIND は大規模なネットワークやインターネットに直接的、間接的に接続されたネットワークに適しています。BIND は、/etc/hosts データベースと比べ、数多くのステーションにアクセスすることができます。BIND の欠点は、設定が複雑なことです。BIND の詳細については、第 6 章「BIND ネーム・サーバ」を参照してください。

ネットワーク・インフォメーション・サービス（NIS: Network Information Service）

NIS はネットワーク・ベースの情報サービスおよび管理ツールです。NIS を使うと、共有データベースを集中管理したり、分散検索することができます。NIS は通常のテキスト・ファイル・ベースのデータベースを複数サポートします。たとえば、NIS データベースは NIS マスターの hosts、passwd、group、および aliases ファイルから作成できます。

NIS は中規模のネットワーク（ステーション数が約 1,000 のネットワーク、または少数のネットワークを相互接続したネットワーク・グループ）に最適です。NIS は NFS のオプション・ソフトウェアの一部です。NIS と NFS の詳細については、『NIS Administrator's Guide』を参照してください。

サブネットのガイドライン

サブネット化により、1 つのネットワークを複数のサブネットに分割することができます。ネットワークをサブネット化すると便利な点が多くあります。たとえば、支社が本社のネットワークに接続する場合、固有のネットワーク番号またはサブネットを使用できます。イーサネットにより多数のシステムが接続されている場合は、サブネット化することで、1 本のイーサネット・ネットワークでサポートできるホスト数やネットワーク・ケーブル長の物理的な限界を克服することができます。

IP アドレスのクラスで扱うネットワークの規模が非現実的に大きい場合には、サブネットを使用します。たとえば、クラス B では、1 つのネットワークに約 64,000 台のステーションを割当てることができます。ただし、これは大半のネットワークで使用できるステーションの数をはるかに上回っています。このような場合は、ホスト ID のビットの一部をサブネットに割当てます。サブネットでは 1 つのネットワークに現実的な数のステーションを接続します。サブネット化に伴う変更は、サイトの管理グループが行います。これらの変更は外部には影響を与えません。

サブネットを設定する前に、計画を立てます。サブネット化の手順については、52 ページの「ネットワークのサブネット化」を参照してください。まず、32 ビットのインターネット・アドレスのホスト部分をどのように分割するかを決めます。ローカル・サブネットを定義するには、ホスト番号の一部のビットを使用して IP アドレスのネットワーク部分を拡張します。このような IP アドレスの再解釈は、ローカル・ネットワークに対してだけ行います。これは、サイト外のステーションには影響しません。サブネットを計画する前に、少なくとも物理的なレイアウトを把握しておきます。たとえば、オフィスの各階ごとにサブネットを設定したり、支社が本社のネットワークに接続する場合は、支社用のサブネットを設定します。SLIP クライアントと PPP クライアント用にサブネットを設定することもできます。詳細については、125 ページの「SLIP および PPP の経路制御とアドレス割当て」を参照してください。

クラス A のネットワーク番号を持つサイトは、24 ビットのホスト部分を使用できます。クラス B のネットワーク番号を持つサイトは 16 ビット、クラス C のネットワーク番号を持つサイトは 8 ビットを使用できます。たとえば、クラス B のネットワーク番号を持つサイトでは、ネットワーク上の各ステーションのインターネット・アドレスは、16 ビットのネットワーク番号と 16 ビットのホスト番号で構成されます。最高 254 台のステーションが存在する 254 のローカル・サブネットを定義するには、アドレスのホスト部から 8 ビットを使用できます。新しいネットワーク番号は、元の 16 ビットのネットワーク番号、およびローカル・サブネット番号に割当てた 8 ビットを連結されたもので構成されます。

メモ：サブネット ID に使用されるホスト番号シーケンスの一部分のサイズは、すべてのサブネットで同じにすることを強くお薦めします。さまざまなサブネットがサポートされていますが、修正が難しくなります。

図 2-3 に、クラス B のインターネット・アドレスにサブネットを割当てる方法を示します。

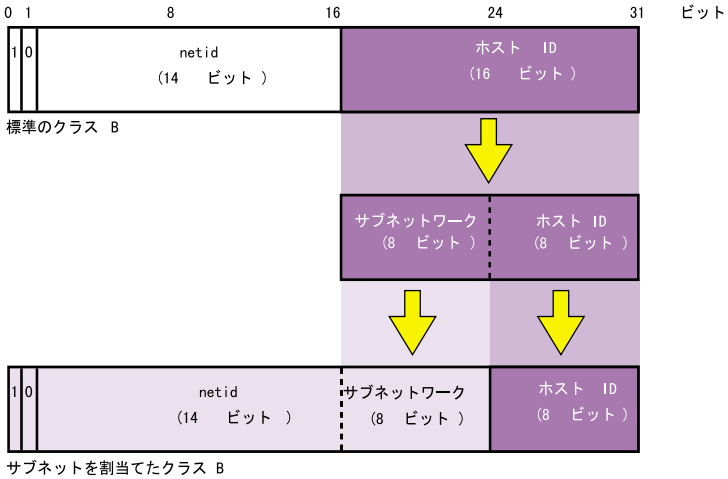

図 2-3 サブネットを割当てたクラス B のアドレス

たとえば、あるサイト全体のクラス B のインターネット・アドレスが 128.50 の場合に、このサイトでサブネット化を行うと、そのサブネットには 128.50.20、128.50.21、128.50.22 などのネットワーク ID が割当てられます。サブネット 128.50.21 に常駐するステーションのインターネット・アドレスは 128.50.21.5 のようになります。

メモ：すべて 0 または 1 で構成されるアドレスは、ブロードキャスト・アドレス用に予約されています。このため、すべて 0 または 1 だけで構成されるサブネット番号は使用できません。

IP アドレスの割当て

ネットワーク番号を取得し、ホスト名の検索に使用するシステムや、ネットワークのサブネット化を決定した後、それぞれのシステムに IP アドレスを割当てます。これは、適切なネットワークまたはサブネットから未使用の IP アドレスをシステムに割当てるだけです。`/etc/hosts` ファイルの構文を使用する場合は、次のように指定します。

```
150.26.80.1     green.salad.com green
150.26.80.2     tossed.salad.com tossed
150.26.80.3     jello.salad.com jello
<IP address>    <host>.<domain> <host> <alias>
```

メモ：ホスト・ナンバー 0 と 255 は予約されているため使用できません。

システムが複数のネットワーク・インタフェースを持つ場合は、複数のサブネットに接続することができますが、接続されるインタフェースごとにアドレスが 1 つ必要です。それぞれのインタフェースに、そのインタフェースが接続されているサブネットからアドレスを割当てます。たとえば、fruit.salad.com を 150.26.80 ネットと 150.26.42 ネットを結ぶゲートウェイとして使用する場合は、次のようにアドレスを指定します。

```
150.26.80.19    fruit.salad.com fruit
150.26.42.1     gate-fruit.salad.com gate-fruit
```

ホスト名の検索に、NIS または BIND を使用する場合でも、`/etc/hosts` ファイルを設定しておきます。システムをネットワークに設定するときにこのファイルをシステムにインストールしておくと、NIS または BIND の起動中も通信を行うことができます。

ネットワークを設定したら、新しいシステムに IP アドレスを割当てる方法を考えます。組織が大きい場合は組織全体を分割してグループを構成し、そのグループごとにアドレス割当ての権限を委任します。たとえば、支社に専用のサブネットがある場合は、そのサブネットから必要に応じて IP アドレスを割当てます。組織がいくつかのサブドメインに分割されている場合は、特定のサブネットに対する権限をそれぞれのサブドメイン管理者に委任します。

ネットワーク・セキュリティ

ネットワーク上でセキュリティを確保するのは容易なことではありません。潜在的な侵入者を未然に防止したり、侵入者が現れた場合に早急に特定し、切離すことができれば、そのネットワークはセキュリティが確保されているといえます。ネットワークをインターネットに接続する前に、ネットワークのセキュリティ対策を立てておきます。ネットワーク・セキュリティの詳細については、『IRIX Admin: Backup, Security, and Accounting』の第 5 章「ネットワークのセキュリティ」を参照してください。

一般的なネットワーク・アプリケーション

このマニュアルは、特にネットワークの標準ハードウェアとソフトウェア、つまりイーサネットを介したインターネット・プロトコルについて説明しています。ただし、標準でなくてもネットワーク環境で一般的に使用されるネットワーク・アプリケーションがいくつかあります。ここでは、ネットワークを計画するときに考慮すべき一般的なネットワーク・アプリケーションについて簡単に説明します。

次に、ネットワークを計画するときに考慮すべき、一般的なネットワーク・アプリケーションについて簡単に説明します。

- 電子メールの説明については、32 ページの「電子メール」を参照してください。
- リモート・ファイルの共有の説明については、33 ページの「ネットワーク・ファイル・システム（NFS: Network File System）」を参照してください。

電子メール

電子メールは、同一のローカル・ステーション上またはリモート・ステーション間でメッセージを送受信するときに使用するプログラム（`sendmail`）です。メールは、UUCP または TCP/IP プロトコルを使って送信することができます。IRIX は、電子メール用にグラフィカル・インタフェースを提供する `Netscape Mail` をはじめ、System V（`/bin/mail`）と 4.3BSD（`/usr/sbin/Mail`）メール・プログラムをサポートしています。

ネットワーク・ファイル・システム（NFS: Network File System）

NFS は、リモート・ステーションのファイルシステムにアクセスし、そのファイルシステムとデータをローカル・ステーションのファイルシステムに接続するネットワーク・プログラムです。NFS を使用すると、ローカル・ステーションからローカル・ファイルシステムにアクセスしているかのようにリモート・ファイルシステムにアクセスできます。

NFS は、ファイルをステーション間で共有したい場合に採用します。NFS を使うと、グループで使用するソフトウェアやデータが 1 台の NFS サーバ上に置かれます。このデータには、権限のある NFS クライアントだけが必要に応じてアクセスできます。NFS を採用すると、一貫した情報を維持でき、クライアント・ステーションのディスク領域を開放できるほか、バックアップの作成も簡単になります。NFS はオプションのソフトウェアです。詳細については、『ONC3/NFS Administrator's Guide』を参照してください。

メモ：NFS は IRIX オペレーティング・システムには含まれていません。別発注になります。

ネットワークの設定

この章では、以下について説明します。

ネットワークで使用するためのシステムの設定

ここでは、BIND または NIS を使用せずに、ローカルの `/etc/hosts` ファイルを使用して IRIS をイーサネット・ネットワーク上に接続し、1 つのインタフェースで使用する方法について説明します。ステーションを設定するには、次の操作を行います。

1. ステーションの電源を落とします。
2. ステーションをネットワークに接続します。
3. ステーションのネットワーク構成を確認します。
4. `/etc/hosts` データベースを変更します。
5. ステーションに名前を指定します。
6. 接続をテストします。

これらの手順に関する説明は、以下を参照してください。

- 「イーサネット・ネットワークへのステーションの接続」（36 ページ）
- 「イーサネット接続の確認」（37 ページ）
- 「ネットワーク・ソフトウェア構成の確認」（39 ページ）
- 「ホスト・データベースについて」（40 ページ）
- 「ホスト・データベースの変更」（41 ページ）
- 「ステーション名の決定」（42 ページ）
- 「ネットワーク接続のテスト」（43 ページ）

イーサネット・ネットワークへのステーションの接続

ステーションをネットワークに接続するには、イーサネット・ケーブルの一方をステーションの裏にある入出力ポートに接続し、もう一方をネットワークに接続します。

イーサネット接続の確認

イーサネットの接続状態を調べるには、ping コマンドを実行します。このコマンドは、イーサネット・ネットワーク上の別のシステムに接続できるかどうかを調べます。次の操作を実行します。

1. システムが接続しているローカル・エリア・ネットワーク上のステーションのホスト名を調べます。調べることが可能な場合には、ドメイン名付きのホスト名と IP アドレスを取得します。この例では、ホスト名は hancock、ドメイン名付きのホスト名は hancock.corp.gen.com、IP アドレスは 128.70.3.56 になります。選択するステーションは、イーサネットに正しく接続されており、動作中でなければなりません。

2. ホスト名と IP アドレスを確認したら、次のコマンドを入力します。

   ping -r *hostname*

   リモード・ホストから返されたパケットを示す一連のレコードが表示されます。この例では、次のレコードが表示されます。

   ```
   PING hancock (128.70.3.56): 56 data bytes

   64 bytes from 128.70.3.56: icmp_seq=0 ttl=255 time=2 ms
   64 bytes from 128.70.3.56: icmp_seq=1 ttl=255 time=2 ms
   64 bytes from 128.70.3.56: icmp_seq=2 ttl=255 time=2 ms
   64 bytes from 128.70.3.56: icmp_seq=3 ttl=255 time=2 ms
   64 bytes from 128.70.3.56: icmp_seq=4 ttl=255 time=2 ms
   ```

3. ping コマンドを中止するには、**<Ctrl>-C** キーまたは **<Delete>** キーを押します。次のように ping コマンドの結果が表示されます。

   ```
   ----hancock PING Statistics----

   5 packets transmitted, 5 packets received, 0% packet loss

   round-trip min/avg/max = 2/2/2 ms
   ```

4. ネットワークが正常に動作していれば、上記のような結果が表示されます。ping の出力内容では、パケット損失が 0% であり、送信されたパケット数と受信されたパケット数が同じです。パケットの一部が失われている場合は、まず、ケーブルがしっかりと接続されているかどうか確認します。通常、ケーブルがしっかりと接続されていない場合にパケット損失の原因となります。往復時間（round-trip）はネットワークの規模と負荷によって異なるので、必ずしもイーサネット接続の問題を示す指標とはなりません。

ping コマンドを正常に実行できない場合は、次の手順に従ってネットワークを調べます。

1. ステーションの IP アドレスを指定して ping コマンドを実行します。ホスト名に hancock を使用した上記の例では、次のように入力します。

   ping -r 128.70.3.56

2. ローカル・ネットワーク上の別のステーションに対して、ping コマンドを使用するか、**ifconfig ec0** コマンドを入力したときに表示されるブロードキャスト・アドレスを使用します。別のシステムから、このシステムが認識されていることと、同じような応答が返されることを確認します。

3. netstat -in コマンドでネットワーク構成を調べます。次のような情報が表示されます。

   ```
   Name Mtu  Network  Address    Ipkts Ierrs Opkts Oerrs
   ec0  1500 128.70.3 128.70.3.9 18    0     18    0
   ```

   ec0 エントリは、プライマリ・イーサネット接続を表し、Ipkts と Opkts フィールドはネットワーク・インタフェースが処理した受信パケットと送信パケットの数を表します。Ierrs と Oerrs フィールドは受信パケットと送信パケットのエラーの数をそれぞれ表します。

   ここではトラブルシューティングを目的としているため、Network ヘッダに表示される IP アドレスが、ping コマンドで指定したホスト名の IP アドレスと一致しているかどうかを確認します。IP アドレスが一致しない場合は、ステーションが別のネットワーク上にあるために ping コマンドが失敗したと考えられます。ping コマンドは、ローカル・ネットワーク上にあるステーションに対して実行します。

4. /var/adm/SYSLOG ファイルにイーサネットのエラー・メッセージが書込まれていないかどうかを調べます。netmask および broadcast の宛先を確認します。**ifconfig ec0** で表示される netmask および broadcast の宛先が、同じネットワーク上またはサブネット上の宛先と一致していることを確認します。

5. 別のステーションがローカル・ネットワーク上で正常に動作していることを確認します。

6. 正しいソフトウェア・パッケージ（eoe.sw.tcp）がシステムにインストールされていることを確認します。

7. イーサネット・ケーブルとトランシーバが物理的に正しく接続されていることを確認します。イーサネット・ハードウェアがしっかりと接続されていない場合には、/var/adm/SYSLOG ファイルおよびシステム・コンソールから次のようなメッセージが表示されます。

   ```
   ec0: no carrier: check Ethernet cable
   ```

8. ケーブルの接続が正常であるにもかかわらずエラー・メッセージが出力される場合は、システムの外側にあるイーサネットの接続部分を取替えます。
9. 同じ IP アドレスを持つ別のホストが存在するというメッセージが表示されたら、同じ IP アドレスを所有するホストを調べ、アドレスの設定が間違っているホストを見つけます。同じアドレスが誤ってセカンダリ・ホストに割当てられていたり、テスト中の新しいシステムに間違ったアドレスが設定されている可能性があります。

ネットワーク・ソフトウェア構成の確認

ステーションが起動された後、ステーションのソフトウェア構成を確認します。まず、*root* でログインします。

`chkconfig` コマンドを実行し、ステーションの標準ネットワーク・ソフトウェアが正しく設定されていることを確認します。

`/etc/chkconfig`

次のような情報が表示されます。

```
named      off
network    on
rwhod      off
timed      on
timeslave  off
gated      off
mrouted    off
rtnetd     off
snmpd      on
routed     on
```

メモ：インストールしたソフトウェアと構成フラグの設定により、出力が上記の例と異なる場合があります。

ネットワーク関連のデーモンを詳しく知っている場合には、ネットワークのニーズに合わせて構成フラグを設定することができます。ネットワーク関連のデーモンに関する知識があまりない場合は、ネットワーク関連のオプションを上記に従って設定します。特に、*network* 変数は必ず有効に設定します。

ホスト・データベースについて

`hosts` ファイルはホスト名のデータベースです。このファイルには、ローカル・ステーションが認識できるステーションに関する情報が記述されています。

`/etc/hosts` データベースは、テキスト・エディタで編集可能な ASCII ファイルです。このファイルには、ステーションのアドレス、正式名称、エリアスを指定したテキスト行が記述されています。正式名称とは、ドメイン名を含む名前のことです。アドレスと名前は、空白とタブのいずれか、またはその両方を使用して区切ります。コメント行は、行の先頭にシャープ記号（#）を指定し、この記号から行の終わりまでがコメントとして解釈されます。

31 ページの「IP アドレスの割当て」で説明しているように、ホスト・データベースを変更する前に、ネットワーク上にあるすべてのステーションの名前と有効なインターネット・アドレスのリストが必要です。ネットワークにルータ、つまり、複数のネットワーク・インタフェースを持つステーションがある場合は、各インタフェースごとの有効なインターネット・アドレスと名前が必要です。IP アドレス設定については、14 ページの「インターネット・プロトコル・アドレス」を参照してください。

この例では、NIS および BIND を使用していないことを前提としています。NIS を使用している場合には、『NIS Administrator's Guide』を参照し、BIND を使用している場合は第 6 章「BIND ネーム・サーバ」を参照してください。

次に、`/etc/hosts` データベースの例を示します。

```
# This is a comment
127.0.0.1   localhost
119.0.3.20  tuna.salad.com tuna  # tuna is an alias
119.0.3.21  chicken.salad.com  salad
119.0.3.22  walrus.salad.com  walrus
```

各システムには、**localhost** とそのすべてのネットワーク・インタフェースのエントリが記述された /etc/hosts のコピーが必要です。出荷時には、/etc/hosts データベースに 2 つのエントリが記述されています。最初のエントリは、ローカル・ネットワーク・ソフトウェアをテストするために使用するホスト名です。

```
127.0.0.1 localhost
```

localhost を参照した場合は、メッセージは内部でループ・バックされ、ネットワーク上では伝送されません。

注意：数多くの重要なプログラムが **localhost** エントリに依存しているため、**localhost** エントリを削除したり変更しないように注意してください。/etc/hosts のマスター・コピーが IRIS ステーション上にない場合や、BIND または NIS を使用している場合でも、ホスト・データベースに **localhost** エントリがあることを確認してください。

メモ：192.0.2.1 は、特殊なアドレスです。このアドレスを割当てられたシステムでは、ネットワーク接続が無効になります。

ホスト・データベースの変更

IRIS システムがネットワークにアクセスできるためには、/etc/sys_id に新しく割当てた IP アドレスと名前を含むエントリを追加します。このエントリには、正式なホスト名またはエリアスのいずれかの sys_id 名が必要です。

上記の /etc/hosts ファイルの例では、ホスト walrus の /etc/sys_id ファイルに walrus または walrus.salad.com のいずれかが記述されていなければなりません。

/etc/sys_id の IRIS システム名を変更する場合は、/etc/hosts のエントリを必ず更新します。更新しないと、ネットワーク・ソフトウェアが正しく初期化されません。ステーションの起動時に次のメッセージが表示された場合、/etc/hosts ファイルと /etc/sys_id ファイルが矛盾しています。この矛盾を訂正してからステーションを再起動します。

```
*** Can't find hostname's Internet address in /etc/hosts
```

IRIS システムがゲートウェイの場合には、異なるネットワークで、各ネットワーク・インタフェースにインターネット・アドレスを割当てる必要があり、/etc/hosts にエントリが必要です。43 ページの「ルータの設定」を参照してください。

各ステーションのホスト・データベースのバージョンは同じでなければなりません。この一貫性を保つ方法は、ネットワークの規模やネットワークをインターネットに接続するかどうかによって異なります。rcp または rdist プログラムで cron コマンドを組合わせて使い、hosts ファイルを同期化します。

/etc/hosts ファイルを編集し、ネットワーク上にあるすべてのステーションのホスト名とインターネット・アドレスを追加します。ネットワーク上の各ステーションには、すべてのステーション名を記述したローカルな /etc/hosts ファイルが必要です。ローカル・ステーションに置く /etc/hosts ファイルが大きすぎる場合は、代わりに必要最低限の内容を記述した hosts ファイルをインストールします。つまり、そのファイルにはそのインストール先であるシステムと、権限付きの /etc/hosts ファイルが置かれているシステムに関する情報だけを記述します。このようにすれば、新しいシステムをネットワークに接続したときに、rcp または ftp コマンドで別のシステムから完全に記述された hosts ファイルをコピーできます。別の方法としては、テープやディスクに格納されている hosts ファイルをコピーすることもできます。

ステーション名の決定

ステーション名を決定したら、/etc/sys_id ファイルを編集してその新しい名前を設定します。

1. デフォルトのステーション名（IRIS）を削除し、新しいステーション名を指定します。この例では、setup1 を使います。次のコマンドを実行します。

   ```
   echo setup1 > /etc/sys_id
   ```

2. 変更内容を有効にするため、次のコマンドでステーションを再起動します。

   ```
   reboot
   ```

ステーションを再起動した後は、その新しいステーション名がログイン・プロンプトとして使用されます。

ネットワーク接続のテスト

ネットワーク管理ツールの rup と ping は、ネットワークの接続に関する情報を即座に提供します。rup は、ローカル・ステーションからリモート・ステーションに接続できないなど、ネットワークに物理的な問題がないかどうか報告します。rup はデフォルトでブロードキャストを使用するので、ルータを経由しません。ステーションがネットワーク上のほかのステーションを認識したら、*ping* で通信機能をテストします。ping はインターネット制御メッセージ・プロトコル（ICMP: Internet Control Message Protocol）を使用します。このプロトコルは、指定されたステーションからエコー・バックを要求します。*ping* は特定のステーション間のパケット送受信が可能かどうかを知らせます。

1. rup コマンドを実行し、ローカル・ステーションからネットワーク上のほかのステーションに接続できるかどうか確認します。

   /usr/bin/rup

   ネットワーク上の各ステーションからの応答が返されます。これらのステーションが起動されていなければ、ユーザ・フレンドリな応答は返されません。また、電源が入っていて、ネットワークに接続していてもユーザ・モードで立ち上がっていなければ、情報は 16 進数で返されます。

2. ping コマンドを実行し、ローカル・ステーションがネットワーク上のリモート・ステーションと通信できるかどうか確認します。

   /usr/etc/ping *station_name*

   数秒間出力した後、**<Ctrl>-C** キーを押して中止します。ping の統計情報を調べます。ping は送信パケット数、受信パケット数、パケット損失率、および往復所要時間（最短／平均／最長）を示します。これらはすべて、ネットワークの一般的な状態を調べる上で有益な指標となります。パケット損失 0%、短い往復所要時間が望ましいことは言うまでもありません。

ルータの設定

ルータは複数のネットワーク・インタフェースを備えたステーションであり、ネットワーク間のパケット転送を行います。ここでは、2 つのインタフェースを備えるルータと 3 つ以上のインタフェースを備えるルータを設定する手順について説明します。1 つのステーションが複数のインタフェースを備えていても、ルータとして機能していないこともあります。ここでは、複数のインタフェースを備えるステーション上で転送を無効化する手順についても説明します。

2 つのインタフェースを備えるルータの設定

デフォルトの方法でインタフェースに名前を付けた場合、`/etc/init.d/network` スクリプトは、2 つのインタフェースを備えるルータを自動的に検出して設定します。第 1 インタフェースと第 2 インタフェースのデフォルトのインターネット・アドレスには、`/etc/sys_id` ファイルで定義されている名前を使います。つまり、第 1 インタフェースには `sys_id` ファイルに定義されている名前を使用し、第 2 インタフェースには `sys_id` ファイルに定義されている名前に接頭辞 *gate-* を付けます。

デフォルトの命名規則を使用して、2 つのインタフェースを備えるルータを設定するには、次の手順に従います。

1. **root** でログインします。
2. `/etc/hosts` ファイルの 2 つのインタフェースに対して、有効なインターネット名とインターネット・アドレスを割当てます。たとえば、ステーション *biway* の第 1 インタフェースと第 2 インタフェースに対する `/etc/hosts` ファイルのエントリは次のように指定します。

   ```
   198.70.75.2     biway.salad.com   biway
   198.70.80.3     gate-biway.salad.com   gate-biway
   ```

3. ルータに適切な名前が `/etc/sys_id` ファイルで定義されていることを確認します。この例では、`/etc/sys_id` ファイルは次のように記述されています。

   ```
   biway
   ```

4. カーネルを再構築し、ステーションを再起動することにより、変更内容とインタフェースの設定を有効にします。次の例に示すように、一部のシステムでは確認を求めるメッセージを出力します。単にシェル・プロンプトに戻るシステムもあります。いずれの場合も、カーネルを再構築した場合には、`reboot` コマンドを実行してください。

```
/etc/autoconfig
Automatically reconfigure the operating system? (y/n)y
reboot
```

メモ：標準の命名規則を使用しない場合には、/etc/config/netif.options ファイルを変更する必要があります。インタフェース名を変更する手順については、53 ページの「/etc/config/netif.options ファイルのネットワーク・インタフェース構成の変更」を参照してください。

3 つ以上のインタフェースを備えるルータの設定

ルータが 3 つ以上のインタフェースを備える場合は、/etc/hosts ファイルと /etc/sys_id ファイルのほかに、/etc/config/netif.options ファイルも変更する必要があります。netif.options ファイルでは、インタフェース・タイプ（enp1、ipg0 など）を定義します。デフォルトでは、第 3 インタフェースと第 4 インタフェースの名前はそれぞれ *gate2-$HOSTNAME* と *gate3-$HOSTNAME* になります。ここで $HOSTNAME は、*hostname* コマンドを実行したときに返される値です。インタフェース名を変更する場合には、53 ページの「/etc/config/netif.options ファイルのネットワーク・インタフェース構成の変更」で詳しい手順を説明しているので参照してください。

デフォルトの命名規則を使用して、3 つ以上のインタフェースを備えるルータを設定するには、次の手順に従います。

1. **root** でログインします。
2. 有効なインターネット名とインターネット・アドレスを /etc/hosts ファイルに登録されているすべてのインタフェースに割当てます。たとえば、4 つのインタフェースを備えるルータ freeway の /etc/hosts ファイルのエントリは、次のように指定します。

   ```
   198.70.30.1     freeway
   198.70.32.4     gate-freeway
   198.70.41.5     gate2-freeway
   198.70.59.6     gate3-freeway
   ```

3. ルータに対して適切な名前が /etc/sys_id ファイルで定義されていることを確認します。この例では、/etc/sys_id ファイルは次のように記述されています。

```
freeway
```

4. `netif.options` ファイルを編集し、インタフェースのタイプを定義します。この例では、第 3 と第 4 のインタフェースは FDDI(ipg*) です。`if3name` と `if4name` 変数を次のように変更します。

   ```
   if3name=
   if4name=
   ```

   から

   ```
   if3name=ipg0
   if4name=ipg1
   ```

5. `netif.options` ファイルに変更内容を保存します。

6. カーネルを再構築し、ステーションを再起動することにより、変更内容とインタフェースの設定を有効にします。次の例に示すように、一部のシステムでは確認を求めるメッセージを出力します。単にシェル・プロンプトに戻るシステムもあります。いずれの場合も、カーネルを再構築した場合には、`reboot` コマンドを実行してください。

   ```
   /etc/autoconfig
   Automatically reconfigure the operating system? (y/n)y
   reboot
   ```

ルーティング動作の設定

ステーションが 2 つ以上のネットワーク・インタフェースを備えている場合、デフォルトで 2 つのインタフェース間の（ルータの）パケット転送が自動的に行われます。ステーションでルータ転送を行わない場合には、パケット転送を無効に設定します。

1. `/etc/config/routed.options` ファイルを変更して、一般のネットワーク経路（**-q**）あるいはローカル・インタフェース経路（**-h**）にルーティング情報を提供しないように設定します。次のように入力します。

   ```
   echo -qh > /etc/config/routed.options
   ```

2. `/etc/init.d/network` スクリプトで、ネットワークを一時的に切った後、再起動します。次のように入力します。

   ```
   /etc/init.d/network stop
   /etc/init.d/network start
   ```

3. ネットワークを再起動した後、パケット転送が行われないことを `netstat` コマンドで確認します。次のように入力します。

   ```
   netstat -s -p ip |grep forward
   ```

転送機能の内容を変更するには、`/etc/config/routed.options` ファイルを作成または編集し、希望するオプションを指定します。次に、`/etc/config/routed.options` ファイルによって変更できる機能をいくつか示します。

- ルーティング情報を公開または非公開にします。
- 受信するすべてのパケットの追跡を可能にします。
- 特定のネットワークに宛てたパケットにフィルタをかけます。

詳細については、routed(1M) マン・ページを参照してください。

マルチキャスト・ルーティングの使用

マルチキャスト・ルーティングは、最短経路でメッセージを受信グループに送信する配信方法です。図 3-1 を参照してください。Silicon Graphics システムでは、`mrouted` プロセスにより、RFC-1075で知られるDVMRP（Distance-Vector Multicast Routing Protocol）を採用しています。

メモ：IRIX 6.3 では、マルチキャスト・ルーティングはネットワーク・インフォメーション・サービス（NIS: Network Information Service）と組み合わせて使用することにより、ネットワークで必要とされる NIS サービス数を減少させていました。詳細については、portmap(1M) および nisserv(7P) マン・ページのマルチキャスト YP に関する説明を参照してください。

マルチキャスト・ルーティングを使用するには、次の操作を実行します。

1. マルチキャスト・ルーティングをサポートする必要のある各ネットワークごとに、ルータを決めます。選択したルータが IRIX バージョン 5.2 またはそれ以降のバージョンをサポートしていることを確認してください。
2. まだインストールしていない場合は、IRIXバージョン5.2の提供ディスクから`eoe.sw.ipgate` サブシステムを各ルータにインストールします。必要に応じて、*autoconfig* コマンドを実行します。

3. **root** で次のコマンドを入力します。

   `chkconfig mrouted on`

4. `reboot` コマンドを実行して、システムを再起動します。

マルチキャスト・パケットの送信について

図 3-1 に、3 個のルータを使用したネットワークの例を示します。

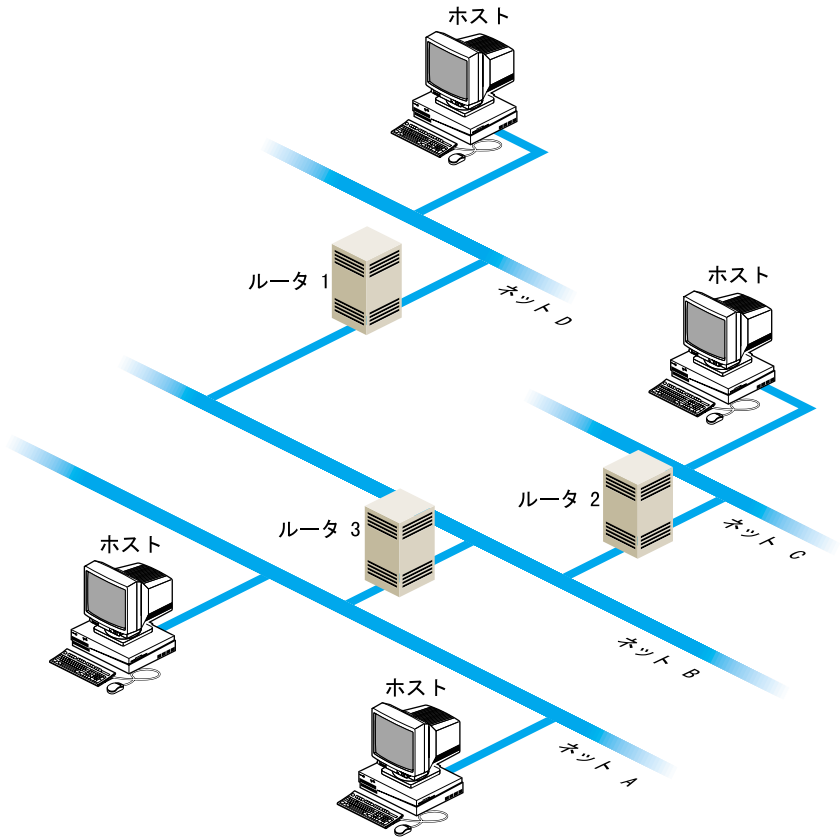

図 3-1　　マルチキャスト・ルータを使用したネットワーク

- この図に示すように、ネットワーク A、C、D 上のユーザがパケットの受信を待っている場合は、4 個のネットワークのすべてがマルチキャスト・パケットを受信します。
- ネットワーク A と C 上のユーザがパケットの受信を待っている場合は、ネットワーク A、B、C がマルチキャスト・パケットを受信します。
- ネットワーク A 上の 3 人以上のユーザがパケットの受信を待っている場合は、ネットワーク A と B がマルチキャスト・パケットを受信します。マルチキャスト・ルーティング・プロトコルは、パケットがネットワークの「葉」（この例ではネットワーク C と D）には送信されないようにしますが、内部のネットワーク（この例ではネットワーク B）には送信します。

マルチキャスト・パケットをサポートするトンネルの設定

ルータがマルチキャスト・ルーティングをサポートしていない場合は、ネットワーク間のトンネルを作成し、マルチキャスト・パケットをサポートします。IRIX バージョン 5.2 以降を実行する Silicon Graphics ワークステーションであれば、トンネルの終端として設定することができます。トンネルを使用すると、マルチキャスト・パケットはユニキャスト・パケット内部にカプセル化された後、トンネルの別の終端に送信されます。これらのユニキャスト・パケットは、受信時にマルチキャスト・パケットに戻されます。

図 3-2 に、ネットワーク A と C との間にトンネルを設定した例を示します。

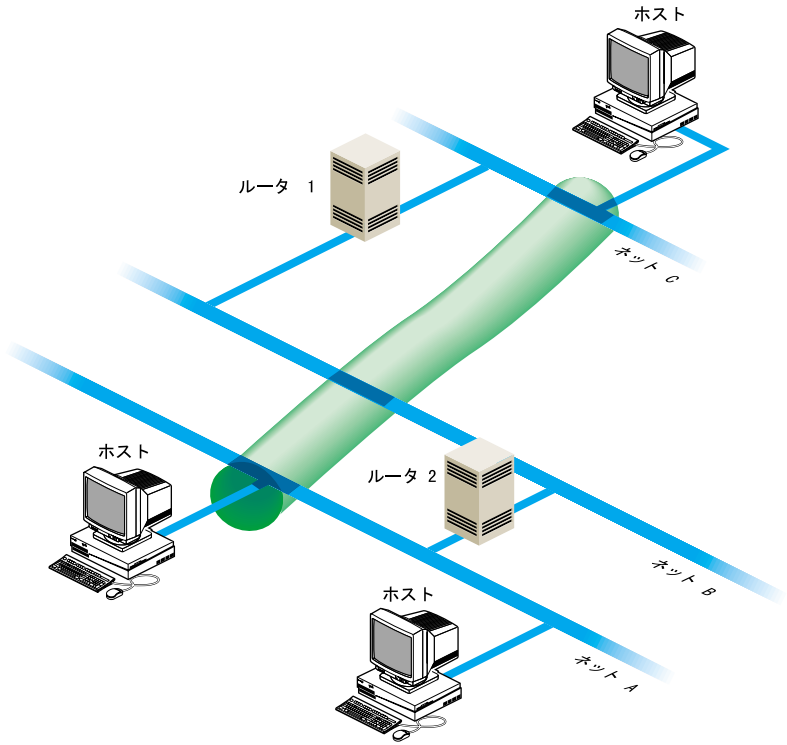

図 3-2 ネットワーク A と C の間のトンネル

トンネルを作成するには、次の手順に従って `/etc/mrouted.conf` ファイルを編集します。

1. ネットワークごとにトンネルの送信側終端および受信側終端として使用するシステムを決めます。

   IRIX バージョン 5.2 以降を実行するシステムのうち高速で負荷の軽いものを選びます。低速で負荷の重いシステムを選ぶと、オーディオ・データやビデオ・データが中断される場合があります。

2. まだインストールしていない場合は、IRIX ディストリビューション・ソースから `eoe.sw.ipgate` サブシステムをインストールします。

3. root になって、トンネルの送信側終端および受信側終端の /etc/mrouted.conf ファイルを編集します。これらの終端の間には数多くのルータが入る可能性があることに注意します。IRIX バージョン 5.2 以降を実行するマシンであれば、どのマシンでもネットワークで使用することができます。

   トンネルを作成するネットワークごとに次の行を追加します。

   tunnel *<local IP address> <remote IP address>*

   上記の例では、ネットワーク D 上のシステムに対しては次の行を追加します。

   tunnel 128.70.58.1 128.65.170.2

   ネットワーク A 上のシステムに対しては次の行を追加します。

   tunnel 128.65.170.2 128.70.58.1

4. トンネルに関してその他のオプションを指定することもできます。詳細については、mrouted(1M) マン・ページを参照してください。

5. システムを再起動します。

メモ：マルチキャスト・パケットの 1 つのコピーが mrouted.conf ファイルの各トンネルのエントリごとに送信されます。このため、ネットワークのトラフィックが増大します。たとえば、1 つのネットワーク・インタフェースを持つワークステーションがある場合に、ほかの 3 つのネットワーク上のワークステーションに対してトンネルを設定すると、パケットの数が 3 倍に増えます。

NIS ユーザのための /etc/rpc ファイルの更新

ネットワーク・インフォメーション・サービス（NIS: Network Information Service）を実行している場合に、IRIX の /etc/rpc ファイルには、特定のマルチキャスト・ユーティリティを使用するための基本的なエントリが含まれます。NIS マスターが IRIX バージョン 5.2 以降のバージョンで実行されていない場合、または NIS マスターが Silicon Graphics ワークステーションではない場合は、NIS マスターの /etc/rpc ファイルに次のエントリがあるかどうか確認します。

```
sgi_iphone 391010
sgi_videod 391011
```

ネットワークのサブネット化

サブネット・アドレス・スキームは、非常に簡単な手順で設定できます。サブネット化は、次の方法を使用します。

1. 「ネットマスクの設定」（52 ページ）
2. 「ステーションの再起動」（53 ページ）

メモ：この手順を実行していない場合は、第 2 章「ネットワークの計画」を参照してください。サブネットのインターネット・アドレスを設定する方法が説明されています。

ネットマスクの設定

`ifconfig` コマンドの **`netmask`** オプションは、インターネット・アドレスのネットワーク部（サブネットを含みます）を解釈したり定義するのに使います。ネットマスクは、インターネット・アドレスのどの部分をネットワーク部とみなすかを定義します。このマスクは通常、標準のネットワーク部とサブネットワークに割当てられたホスト部の一部に相応するビットを所有しています。アドレスを設定するときにネットマスクを指定しなかった場合には、ネットワークのクラスに応じて設定されます。

サブネット化されたクラス B のアドレス（サブネットの定義に、ホスト ID の 8 ビットを使用）を認識するようにステーションの第 1 インタフェースを設定するには、`/etc/config/ifconfig-1.options` ファイルを作成または変更し、次の行を挿入します。

```
netmask 0xffffff00
```

このネットマスク値は、第 1 インタフェースにおいて、インターネット・アドレスの上位の 24 ビットがアドレスのネットワーク部を、下位の 8 ビットがホスト ID を表すことを示しています。クラス B のアドレス（16 進値）のサブネット化されていないネットマスクは 0xffff0000 となります。ネットマスク値は、16 進数のドット表記法によるインターネット・アドレスか、または疑似ネットワーク名の形式で設定します。この入力はサブネット上のすべてのステーションで行う必要があります。

メモ：複数のインタフェースを備えるステーションでは、適切な `ifconfig-*.options` ファイルで各インタフェースごとにネットマスクを設定します。

異なるインタフェース上の異なるマスクなど、さまざまなサブネット・マスクがあり、それらは IRIX 6.2 ではサポートされています。ただし、設定と維持は簡単ではありません。そのため、サイトのニーズを満たすためにほかに手段がない場合をのぞき、これらを使用することはお薦めできません。

ステーションの再起動

ネットマスク値を設定した後で、ステーションを再構築して再起動することにより、新しいネットワーク・アドレスをステーションのルーティング・テーブルに登録します。ルータは、ほかのステーションやネットワークにルーティング情報を提供するので、ステーションを再起動する前に必ず再起動します。

カーネルを再構築しステーションを再起動することにより、変更内容とインタフェースの設定を有効にします。一部のシステムではカーネルを再構築する前に確認を求めるメッセージが出力されます。単にシステム・プロンプトに戻るシステムもあります。いずれの場合も、カーネルを再構築した場合には、`reboot` コマンドを実行する必要があります。

```
/etc/autoconfig
Automatically reconfigure the operating system? (y/n)y
reboot
```

/etc/config/netif.options ファイルのネットワーク・インタフェース構成の変更

ステーションのネットワーク・インタフェースは必ずしも変更する必要はありません。ほとんどの場合、デフォルトの構成を使用することができます。ネットワーク・インタフェースの設定を変更した場合には、`/etc/config/netif.options` ファイルも変更する必要があります。次のような場合には、このファイルを変更する必要があります。

- ステーションが 3 つ以上のインタフェースを備えている場合
- 名前の指定にデフォルトの命名規則を使用しない場合

- デフォルトで設定されている順番が正しくない場合、あるいは順番を変更する場合

ステーションにインストールされているネットワーク・インタフェースは、`hinv` コマンドにネットワーク・オプションを指定することにより、調べることができます。4 ページの「コントローラのインタフェース名」と hinv(1M) マン・ページを参照してください。また、複数のネットワーク・インタフェースの設定に関しては、66 ページの「複数のネットワーク・インタフェースの設定」を参照してください。

`netif.options` ファイルには設定可能な変数が 2 つあります。インタフェース名とインタフェース・アドレスです。インタフェース名の変数は、使用するインタフェースの順番（第 1、第 2、第 3 または第 4）とタイプを指定します。インタフェース・アドレスの変数は、有効なインターネット・アドレスを各インタフェースに割当てます。`/etc/hosts` ファイルには、各インタフェースに対して有効なインターネット・アドレスが指定されていなければなりません。

ここでは、以下について説明します。

- 「/etc/config/netif.options ファイルのインタフェース名の変更」（55 ページ）
- 「/etc/config/netif.options ファイルのインタフェース・アドレスの変更」（56 ページ）

表 3-1 は、インタフェース名とインタフェース・アドレスの変数をまとめています。

表 3-1 netif.options ファイルの変数

変数名	変数	例
インタフェース名	if*xname*= ただし、 *x* = 1、2、3 または 4 *name*=ec0、et0、enp0、enp1、fxp1、ipg0、ipg1 など	if1name=enp0 if2name=ipg0 if3name=enp1 if4name=enp2
インタフェース・アドレス	if*xaddress*= ただし、 *x* = 1、2、3 または 4 *address*=$HOSTNAME、ステーション名、またはインターネット・アドレス	if1address=$HOSTNAME if2address=fddi-$HOSTNAME if3address=gate-goofy if4address=128.70.28.2

これらの変数は、どちらか一方または両方を変更することができます。次に、両方の変数を変更する方法について説明します。

メモ：`$HOSTNAME` は、`/etc/sys_id` の値で設定されます。

/etc/config/netif.options ファイルのインタフェース名の変更

ステーションが 3 つ以上のネットワーク・インタフェースを備えている場合は、`/etc/config/netif.options` ファイルのインタフェース名のエントリを変更し、インタフェースの順番を設定します。デフォルトでは最初の 2 つのインタフェースの順番しか設定されていません。また、設定されている順番（第 1 インタフェース、第 2 インタフェースなど）やインタフェースのタイプを変更したい場合も、`/etc/config/netif.options` ファイルを変更します。ここでは、第 1 インタフェースの FDDI を第 2 インタフェースに設定します。

1. `netstat` コマンドを使用して、ネットワーク・インタフェースの名前を確認します。

   ```
   /usr/etc/netstat -ina
   ```

2. `vi` または任意のエディタを使用し、`netif.options` ファイルを開いて編集します。

   ```
   vi /etc/config/netif.options
   ```

3. 該当するインタフェース名の変数を見つけて変更します。ここでは、第 2 インタフェース名の変数（*if2name*) を見つけ、FDDI による第 1 インタフェースを第 2 インタフェースとして指定します。

   この設定を、

   ```
   : if2name =
   ```

   次のように変更します。

   ```
   if2name=ipg0
   ```

   注意：デフォルト変数（第 1 インタフェースと第 2 インタフェース）はすべて、先頭がコロン（:）で始まります。デフォルトのインタフェース名を変更するには、この先頭のコロン（:）を削除してインタフェース名を入力してください。

4. ファイルを保存して終了します。

これ以外を変更しない場合は、ステーションを自動構築し再起動します。引き続き変更する場合は、各インタフェース名に対して上記の手順を繰り返して変更します。

メモ：1 つのネットワーク・インタフェースの順番を入れ替えても、それ以外のインタフェースの相対的な順番は変わりません。たとえば、3 つのインタフェースを備えるステーション（a= 第 1 インタフェース、b= 第 2 インタフェース、c= 第 3 インタフェース）において、デフォルトの第 2 インタフェースを第 1 インタフェースに変更すると(b= 第 1 インタフェース)、インタフェース a と c はそれぞれ第 2 インタフェースと第 3 インタフェースに設定されます。

/etc/config/netif.options ファイルのインタフェース・アドレスの変更

デフォルトのインタフェース･アドレスを変更するには、`/etc/config/netif.options` ファイルを変更します。すべてのインタフェース名は、`/etc/hosts` ファイルで定義されている有効なインターネット・アドレスと一致していなければなりません。$HOSTNAME 変数は `/etc/sys_id` ファイルからステーション名を検索します。この例では、第 2、第 3、および第 4 インタフェース・アドレスを次のように変更します。

- 第 2 インタフェース・アドレス：fddi-$HOSTNAME
- 第 3 インタフェース・アドレス：gate-goofy
- 第 4 インタフェース・アドレス：128.70.28.26

上記の例に従って、ネットワーク・インタフェース・アドレスを変更します。

1. `/etc/hosts` ファイルにある各インタフェースのエントリが有効かどうか確認します。または、有効なエントリを指定します。各インタフェースの名前とアドレスをメモしておきます。
2. `vi` または任意のエディタを使用し、`netif.options` ファイルを開いて編集します。

   ```
   vi /etc/config/netif.options
   ```

3. 該当するインタフェース・アドレスの変数を見つけて変更します。ここでは、第 2、第 3、および第 4 インタフェース・アドレスの変数を変更します。各変数を見つけて、次のように変更します。

   この設定を、

```
: if2addr=gate-$HOSTNAME
if3addr=gate2-$HOSTNAME
if4addr=gate3-$HOSTNAME
```

次のように変更します。

```
if2addr=fddi-$HOSTNAME
if3addr=gate-goofy
if4addr=128.70.28.26
```

注意：デフォルト変数（第 1 インタフェースと第 2 インタフェース）はすべて、先頭がコロン（:）で始まります。デフォルトのインタフェース・アドレスを変更するには、この先頭のコロン（:）を削除してインタフェース・アドレスを入力してください。

4. ファイルを保存して終了します。

これ以外にも変更を行う場合は、上記の手順を繰り返し、個々のインタフェース・アドレスを変更します。これ以外に変更を行わない場合は、ステーションを再設定して再起動します。

IP エリアスの割当て

`ifconfig` コマンドの IP エリアシング機能を使用することにより、複数の IP アドレスを 1 つのネットワーク・インタフェースに割当てることができます。ifconfig(1M) マン・ページを参照してください。この機能は、同じネットワーク・インタフェースを複数のネットワークに接続する場合に便利です。また、ユーザが新しいアドレスを取得するまでの間、別のアドレスを使用したり、それを無効に設定する場合にも利用できます。IP エリアシングはすべてのネットワーク・インタフェース・タイプでサポートされています。

次に、IP エリアシング機能を理解し使用する方法を説明します。

- 「IP エリアシングの使用」（58 ページ）
- 「IP エリアス割当てのガイドライン」（58 ページ）
- 「IP エリアス・ファイルの作成」（58 ページ）
- 「IP エリアシングとルーティング」（59 ページ）

IP エリアシングの使用

IP エリアシングがホストで使用されている場合には、/etc/host ファイルが /etc/sys_id のホスト名に割当てるアドレスが、インタフェースのプライマリ・アドレスになり、ifconfig コマンドで割当てられる代替え名が IP エリアスになります。ネットワーク情報を表示すると（103 ページの「netstat によるネットワーク統計情報の収集」を参照）、インタフェースのプライマリ・アドレスが最初に表示され、次にそのエリアスが表示されます。

IP エリアスを割当てるには、ifconfig コマンドの次のフォームを使用してください。

```
# ifconfig interface alias address [netmask mask_num] [broadcast address]
```

メモ：IP エリアスを割当てる前に、58 ページの「IP エリアス割当てのガイドライン」を必ず参照してください。

エリアスの値には、IP アドレスの代わりに /etc/hosts に登録されているホスト名を指定することができます。たとえば、以下に示すどちらのコマンドを使用しても IP エリアスを ec0 インタフェースに割当てることができます。この例で使用しているインタフェースは、ネットマスクとブロードキャスト・アドレスですが、これらのフィールドの指定は任意です。

```
# ifconfig ec0 alias 128.70.40.93 netmask 0xffffff00 broadcast 128.70.40.255
# ifconfig ec0 alias plum netmask 0xffffff00 broadcast 128.70.40.255
```

IP エリアス割当てのガイドライン

指定したネットワーク・インタフェースに割当てることのできる IP エリアスの数には制限がありません。ただし、推奨する最大エリアス数は、1000 です。また、インタフェースのプライマリ・アドレスとそのすべての IP エリアスは、サブネット・アドレスを共有する必要があります。たとえば、クラス B ネットワーク上のインタフェースのプライマリ・アドレスが 128.70.12.23 の場合、その IP エリアスは、128.70.12.19、128.70.12.53 のようになります。

IP エリアス・ファイルの作成

IP エリアス設定は、/etc/config/ipaliases.options ファイルに保存し、ネットワークの初期化プロセスで自動的に割当てることができます。/etc/config/ipaliases ファイルで on の値が設定されている場合には、ネットワーク・スクリプトが ipaliases.options ファ

イルを読込みます。次の手順を使用して、ipaliases.options ファイルを作成し、ネットワークの初期化プロセスで自動的に IP エリアスを割当てるように設定します。

1. 任意のエディタを使用して、ipaliases.option ファイルを編集します。

   ```
   # vi /etc/config/ipaliases.options
   ```

   次の例のようなエントリになります。この例では、ブロードキャストがイーサネット上で実行されますが、FDDI ネットワークでは実行されません。

   ```
   ec0  128.64.56.51 netmask 0xffffff00 broadcast 128.64.56.255
   ec0  128.64.56.12 netmask 0xffffff00 broadcast 128.64.56.255
   ipg0 190.111.79.16 netmask 0xffffff00
   ipg0 190.111.79.10 netmask 0xffffff00
   ```

2. ipaliases ファイルを作成し、このファイルの IP エリアシングをオンに設定します。

   ```
   # chkconfig -f ipaliases on
   ```

3. ネットワークを停止し再起動して、変更を有効にします。

   ```
   /etc/init.d/network stop
   /etc/init.d/network start
   ```

IP エリアシングとルーティング

各 IP エリアスは、プライマリ・アドレスとエリアス間のホスト経路として、IRIX ルーティング・テーブルで管理されます。デフォルト設定では、IRIX は複数のネットワーク・インタフェースを備えているホストをルータとして認識します。これらのマルチホーム・ホストは、パケット転送を実行し、接続しているネットワークのルーティング情報を表示します。ただし、1 つのネットワーク・インタフェースを備えるホストで IP エリアシングを実行（たとえば、ローカル・ネットワークまたはインターネットの接続のために）している場合には、ルーティング情報は表示されません。このような場合には、ローカル・ホストのネットワーク位置を知るために経路情報を表示する必要があります。

経路情報を表示するには、IP エリアスが割当てられているホストの routed.options ファイルに -s オプションを入力します。

ifconfig-#.options ファイルのネットワーク・パラメータの変更

ネットワーク・パラメータは、インタフェースがネットワーク情報をどのように処理または提供するかを設定します。ネットワーク・パラメータを変更するには、該当する `/etc/config/ifconfig-#.options` ファイルを作成または変更します。ここで、シャープ記号（#）はインタフェース名に応じて 1、2、3、または 4 のいずれかを指定します。

デフォルトのパラメータ設定は、変更しなくても十分機能し、サイトのニーズに適応します。この設定を変更すると、ネットワーク機能に異常が発生する可能性があります。これらのパラメータの変更は、経験豊富なネットワーク管理者が行うようにしてください。

`ifconfig-#.options` ファイルには、デフォルト設定のないネットワーク・パラメータが 4 つあります。

- ネットマスク
- ブロードキャスト・アドレス
- アドレス解決プロトコル（ARP: Address Resolution Protocol）
- ルート・メトリック

これらのデフォルトで設定されていないパラメータを設定するには、次の手順に従います。

1. `netstat` コマンドを使用して、構成時に割当てるネットワーク・インタフェースの順番を確認します。

   `/usr/etc/netstat -i`

2. vi または任意のエディタを使用し、`/etc/config/ifconfig-#.options` ファイルを作成または編集します。ここで、シャープ記号（#）はネットワーク・インタフェース名を表します。たとえば、第 1 インタフェースを設定するには、`/etc/config/ifconfig-1.options` ファイルを作成または編集します。

3. ネットワーク・インタフェースのネットマスク値を変更するには、`netmask`、スペース、32 ビットの 16 進値、ドット表記法によるインターネット・アドレス、あるいは疑似ネットワーク名の順に入力します。

   `netmask 0xffffff00`

   ネットマスクの詳細については、47 ページの「マルチキャスト・ルーティングの使用」を参照してください。

4. ネットワーク・インタフェースのブロードキャスト・アドレスを変更するには、`broadcast`、スペース、ドット表記の 10 進数 IP ブロードキャスト・アドレス、の順に入力します。

   `broadcast 189.170.6.0`

   アドレス解決プロトコル（ARP: Address Resolution Protocol）を有効または無効に設定するには、それぞれ `arp`（有効）または `-arp`（無効）を入力します。

   `arp`

   ARP テーブルは `netstat`、`ifconfig` などのネットワーク管理ツールで使用します。これは、管理者に有用なネットワーク情報を提供します。

5. ネットワーク・インタフェースのルーティング・メトリック・カウントを変更するには、`metric`、スペース、カウント数、の順に入力します。

   `metric 7`

   hop と呼ばれるデフォルトのメトリック・カウントは、0 です。*routed* デーモンは、あるネットワークから別のネットワークへのデータの経路制御に必要な hop の数をモニターします。特定のルートのネットワーク・トラフィックスを軽減したい場合には、そのルートのメトリック・カウントを増分します。

第 2 インタフェースのインタフェース設定ファイルは、次のように記述されています。

`cat /etc/config/ifconfig-2.options`

```
netmask 255.255.255.0
broadcast 129.78.50.0
-arp
metric 4
```

この設定ファイルでは、クラス B のネットワークが、サブネットワークのホスト ID の 8 ビットを使用してサブネット化されています。この例では、ブロードキャスト・アドレスとしてデフォルト値の 1 の代わりに 0 を使用しています。ARP は無効に設定され、メトリック・カウント（hop）はルータ・トラフィックを軽減するために 4 に設定されています。

ブロードキャストまたはマルチキャストをサポートしないネットワーク用の /etc/gateways ファイルの設定

メモ：ゲートウェイは、ホストまたはルータであり、**-s** オプションを指定した `routed` デーモンを実行して、ルーティング情報を提供します。一方で、クライアントはルーティング情報を提供しません。つまり **-q** オプションを指定した `routed` を実行しています。詳細については、routed(1M) マン・ページを参照してください。

`/etc/gateways` ファイルには、遠隔にあるゲートウェイ（ルーティング情報を提供するホストまたはルータ）の一覧が含まれています。この一覧は、ブロードキャストまたはマルチキャストをサポートしないネットワーク（ATM または HIPPI など）によって使用されます。

このファイルにより、以下のいずれかの情報がホストに通知されます。

- サブネットにホストを持つゲートウェイはどれか
- 直接接続されたゲートウェイを介してアクセス可能なサブネット／ゲートウェイはどれか

ホストは、このファイルを使用してルーティング情報プロトコル（RIP）要求の送信先を判別します。ホストによってルーティング情報が提供されている場合は、RIP 応答の返送先も判別します。

/etc/gateways ファイルのフォーマット

`/etc/gateways` ファイル内のエントリには、ATM ネットワークまたは HIPPI ネットワーク上でアクセス可能なすべてのゲートウェイを含める必要があります。このファイルのフォーマットを以下に述べます。

`/etc/gateways` ファイルは複数の行で構成されます。それぞれの行は、次のいずれかのフォーマットで示されます。

フォーマット 1：

`host` *Nname* [`mask` *value]* `gateway` *Gname* `metric` *value* `[passive|active|external]`

フォーマット 2：

`host` *Hname* `gateway` *Gname* `metric` *value* `[passive|active|external]`

それぞれの意味は次のとおりです。

Nname、*Hname*　接続先のネットワークまたはホストの名前。

`mask` *value*　オプション。1 から 32 までの値を指定して、*Nname* に関連するネットマスクを示します。

Gname　RIP から発行される要求の送信先ゲートウェイのアドレス。ホストによってルーティング情報が提供されている場合は、応答の返送先も示します。

`metric` *value*　送信先のホストまたはネットワークまでのホップカウント

`passive`　ゲートウェイは RIP パケットを交換しません。この種のゲートウェイを介したルートは、起動時にカーネルのルーティング・テーブルにインストールされます。ただし、ルーティングの更新情報には含まれません。

`active`　ゲートウェイは RIP パケットを交換します。

`external`　`passive` のゲートウェイが、カーネルのルーティング・テーブルに記録されていることを示します。エントリが「external」として指定されると、必要に応じてそのルートが他のルーティング・プロセスによってインストールされることを示します。

`net` または `host` 以外の文字で始まる行には、次のパラメータのうち 1 つ以上を設定する必要があります。パラメータは、カンマや空白で区切ります。

`if=ifname`　このパラメータは、行の他のパラメータがインタフェース名 ifname を適用していることを示します。

`subnet=`*nname* `[/mask` *value*`][,metric]`
このパラメータは、`mask` 値および `metric` 値（デフォルトは 1）を指定したネットワーク *nname* へのルートを宣言します。

ネットワーク番号には、32 ビットを完全に指定する必要があります。`192.0.2` ではなく `192.0.2.0` と指定します。ループバック・インタフェースのエリアスを追加する場合は、追加するエリアスはアクセス可能である必要があります。

`passive` このパラメータは、別のインタフェースを使って送信された更新情報内に、このインタフェースが宣言されないようにします。また、このインタフェースを使ったすべての RIP およびルータの情報通達を無効にします。これは、フェイルセーフなインタフェースまたは RIP トラフィックを送信しないためのインタフェースに有効です。

`no_ripv1_in` このパラメータは、RIPv1 受信応答を無視します。

`ripv2_out` このパラメータは、可能な場合に RIPv2 出力を生成し、マルチキャストのために RIPv2 通達を行います。

`ripv2` このパラメータは、`no_ripv1_in` および `ripv2_out` を指定した場合と同じです。

`rdisc_interval=`*N*
このパラメータを使用して、ルータからの情報通達が転送される間隔を N 秒に設定します。情報のライフタイムは 3 × *N* 秒です。この値は、45 秒に設定することを推奨します。

また、`routed` を **`-P parm`** オプションで実行する以外にも、`/etc/gateways` ファイルに *parm* パラメータ行を追加することもできます。

/etc/gateways ファイルの例

例 3-1 に示す `/etc/gateways` ファイルでは、ホストには 2 つの HIPPI があり、1 つは `192.168.0.0/27` サブネット、もう 1 つは `192.168.0.96/27` サブネット上にあることを示しています。このホストはクライアントとして機能します。ホストは RIP 要求を `192.168.0.0/27` サブネット上の 1 つのゲートウェイ、および `192.168.0.96/27` サブネット上の 2 つのゲートウェイに送信します。`et0` インタフェースでは、RIP パケットの生成や処理は実行しません。

例 3-1 2 つの HIPPI インタフェースを持つホストのための /etc/gateways ファイル

```
#if brodcast is not supported
#-- 192.168.0.0/27 subnet
host 192.168.0.1 gateway 192.168.0.1 metric 1 active i
#-- 192.168.0.96/27 subnet
host 192.168.0.97 gateway 192.168.0.97 metric 1 active
host 192.168.0.99 gateway 192.168.0.99 metric 1 active
```

```
ripv2_out
rdisc_interval=45
if=et0 passive
if=et1 passive
```

例 3-2 に示す /etc/gateways ファイルは、ゲートウェイとして機能するホストに属しています。ホストには 2 つの HIPPI があり、1 つは 192.168.0.96/27 サブネット、もう 1 つは 192.168.0.128/27 サブネット上にあります。また、ルーティングの更新情報をこのホストと交換するゲートウェイが、192.168.0.96/27 サブネット上に 1 つと 192.168.0.128/27 サブネット上に 1 つあります。

このホストは、192.168.0.202 サブネットに単独で恒久的な IP アドレスを持っています。この IP アドレスは、実際にはエリアスで、少なくとも 1 つのアクティブ・インタフェースが存在するあいだは宣言されます。よって、エントリーは subnet=192.168.0.202/31,1 です。et0 インタフェースでは、RIP パケットの生成や処理は実行されません。

例 3-2 ホストのゲートウェイ機能のための /etc/gateways ファイル

```
#if brodcast is not supported
#-- 192.168.0.96/27 subnet
host 192.168.0.97 gateway 192.168.0.97 metric 1 active
host 192.168.0.99 gateway 192.168.0.99 metric 0 active
#-- 192.168.0.128/27 subnet
host 192.168.0.129 gateway 192.168.0.129 metric 1 active
host 192.168.0.131 gateway 192.168.0.131 metric 0 active

subnet=192.168.0.202/31,1

ripv2_out
rdisc_interval=45
if=ef0 passive
```

詳細については、routed(1M) マン・ページを参照してください。

複数のネットワーク・インタフェースの設定

複数のネットワーク・インタフェースを備えるステーションを設定する場合には、転送機能を無効設定することを除いて、ルータの設定方法と同じです。複数のネットワーク・コントローラ（物理ボードあるいはチップ）がある場合には、システムはそれぞれに対して一意のインタフェース名を持つ必要があります。複数のインタフェースを備えるシステムは通常複数のサブネットに接続するため、各接続インタフェースに対してアドレスが必要になります。

デフォルトのネットワーク設定ファイルである `/etc/init.d/network` は、ネットワーク・インタフェース名およびアドレスを初期化しテストします。サイト固有のネットワーク設定ファイルはデーモンを管理し、オプションとして `/etc/config/netif.options` および `/etc/config/ifconfig-<n>.options` を使用できます（ここで <n> は、インタフェースの変数名に応じて、1、2、3 のように指定します）。`/etc/config/netif.options` ファイルは、`/etc/init.d/network` ファイルで指定されているデフォルト設定を上書きします。

デフォルトでは、ネットワーク構成スクリプトは、システムに存在するネットワーク・インタフェースの数に関係なく、2 つのネットワーク・インタフェースのみを設定します。設定するネットワーク・インタフェースの数を増分するには、次の操作を実行します。

1. すべての一意のネットワーク・インタフェースを識別します。4 ページの「コントローラのインタフェース名」を参照してください。
2. `/etc/config/netif.options` ファイルのネットワーク・インタフェースの設定を変更します。53 ページの「/etc/config/netif.options ファイルのネットワーク・インタフェース構成の変更」を参照してください。
3. IP エリアシングを割当てます。57 ページの「IP エリアスの割当て」を参照してください。
4. `ifconfig-#.options` ファイルのネットワーク・パラメータを変更します。60 ページの「ifconfig-#.options ファイルのネットワーク・パラメータの変更」を参照してください。
5. ステーションを再起動します。

proclaim によるダイナミック・ホストの構成

ここでは、IRIX の `proclaim` 機能について説明します。この機能は、RFC 2131 で規定されている DHCP（Dynamic Host Configuration Protocol）に基づいています。proclaim と DHCP を使用すれば、ホスト名とネットワーク・アドレスを自動的に割当てることができます。

DHCP に従ってサーバ・システムを設定すると、サイト管理者は新しいクライアント・システム、IP アドレスを必要としているクライアント・システムに IP アドレスとサイトの構成パラメータを動的に割当てることができます。このため、割当てられるアドレスがわずかしかないサイトでも、ネットワークに接続する頻度の少ないホストにその都度アドレスを割当てることができます。また、大規模なサイトでも、サイト管理者をわずらわせることなく永久的なアドレスをホストに割当てることができます。

`proclaim` 機能は、マスター・サーバ上で動作するデーモン、サブネット上にあるサーバのリレー・デーモン（オプション）、GUI 設定プログラム、DHCP サーバと通信するクライアント・アプリケーションから構成されます。これらすべてのプログラムは、IRIX オペレーティング・システムとともに提供されます。

ネットワーク、IP アドレス、およびネットマスクに関する基本概念を知りたい場合には、これ以降の説明に進む前に、第 1 章「ネットワーク製品について」、14 ページの「インターネット・プロトコル・アドレス」、および 52 ページの「ネットマスクの設定」をまずお読みください。

次に、`proclaim` サーバとクライアントの設定について説明します。

- 「DHCP サーバの設定」（68 ページ）
- 「DHCP リレー・エージェントの設定」（68 ページ）
- 「proclaim クライアントについて」（69 ページ）
- 「DHCP の制約事項」（69 ページ）

DHCP サーバの設定

`dhcp_bootp` プログラムは、DHCP クライアントや `proclaim` クライアントと通信し、IP アドレスなどのホストの構成情報を提供するサーバです。サイトで DNS を使用して hosts マップを管理している場合は、スクリプトなどの外部メカニズムを使って DHCP サーバの DNS マップを更新します。NIS を使用して hosts マップと ethers マップを管理する場合は、NIS マスター・サーバと同じマシン上で `dhcp_bootp` を実行します。

DHCP サーバでは、通常は要求を発信したクライアントのサブネット番号に基づく構成パラメータを使用します。すべての設定ファイルは、`/var/dhcp/config` ディレクトリにあり、`config.`*<netnumber>* 形式で名前が付けられます。たとえば、192.78.61 ネットワークのクライアントの設定ファイルには、`config.192.78.61.0` という名前が付けられます。サブネット設定ファイルが存在しない場合は、`config.Default` ファイルを使います。このファイルがない場合や読込むことができない場合は、DHCP サーバのアドレスを基にクライアントが設定されます。

DHCP サーバを設定するのに `ProclaimServerMgr` グラフィカル・インタフェースを使うことができます。ProclaimServerMgr(1M) マン・ページを参照してください。必要であれば、テキスト・エディタを使用して dhcp_bootp(1M) マン・ページに示されているキーワードを変更し、サーバのオプションを設定することもできます。

DHCP リレー・エージェントの設定

`dhcp_relay` プログラムは、DHCP メイン・サーバと通信し、DHCP クライアントにホストの構成情報を提供するサブネット・エージェントです。サイトで DNS を使用して hosts マップを管理している場合は、スクリプトなどの外部メカニズムを使って DHCP サーバの DNS マップを更新します。NIS を使用して hosts マップと ethers マップを管理している場合は、NIS マスターと同じマシン上にサイトの DHCP メイン・サーバが必要です。いずれの場合でも、DHCP メイン・サーバのあるサブネットだけでなく、すべてのサブネットで `dhcp_relay` を実行する必要があります。

DHCP リレー・エージェントでは、設定ファイル `/var/dhcp/config/dhcp_relay.servers` を読込んでメインの DHCP サーバの場所を確認します。この設定ファイルには、DHCP サーバの IP アドレスとホスト名がそれぞれ別の行に指定されています。

リレー・エージェントを設定する場合に、ProclaimRelayMgr グラフィカル・インタフェースを使うこともできます。ProclaimRelayMgr(1M) マン・ページを参照してください。必要であれば、テキスト・エディタを使用し、dhcp_relay(1M) マン・ページに示されている手順に従って、リレー・エージェントのオプションを設定します。

proclaim クライアントについて

proclaim クライアントは DHCP サーバと通信し、IP アドレスと IP アドレス・リースなどのホスト構成パラメータを取得します。proclaim を使用して、新しいシステムを自動的に設定し構成することができます。また、サイト管理者をわずらわせることなく、ネットワーク間でシステムを移動することもできます。proclaim クライアントは、再起動時に構成を確認することもできます。

クライアント・システム上に proclaim をセットアップする方法については、proclaim(1M) マン・ページを参照してください。

DHCP の制約事項

現行のリリースでは、次の制限事項があります。

- サイトが、ホスト名、IP アドレス、またはネットワーク・ハードウェア・アドレスの検証とマッピングに NIS を使用している場合には、DHCP サーバは NIS マスターでなければなりません。NIS はオプションのソフトウェア製品であり、必ずしもすべてのシステムやネットワークで使用されているわけではありません。
- サイトが DNS を使用して hosts マップを管理している場合は、DHCP から DNS マップに更新するために、外部メカニズム（手動メカニズムまたは自動メカニズム）が必要です。次回のリリースよりこの制約は無くなります。
- サーバは、DHCP と単一のパケットに含まれる bootp クライアント要求の両方に応答することはできません。ただし、通常の bootp 要求に対しては、bootp サーバとして動作します。

ローカル・ネットワーク・スクリプトの作成

ローカルで構築されたネットワーク・デーモンの起動と終了、または ARP エントリの発行やルート設定を行うには、シェル・スクリプト `/etc/init.d/network.local` を作成します。このファイルに対して `/etc/rc0.d` と `/etc/rc2.d` をシンボリック・リンクすることにより、ステーションの起動時と停止時にこのファイルが呼び出されます。

```
ln -s /etc/init.d/network.local /etc/rc0.d/K39network
ln -s /etc/init.d/network.local /etc/rc2.d/S31network
```

このスクリプトの基本形式に関しては、`/etc/init.d/network` を参照してください。また network(1M)、rc2(1M) および rc0(1M) のマン・ページも合わせて参照してください。

リモート・アクセスのログ

ネットワーク・デーモンには、`syslogd` を使用してリモート・アクセスをステーションのログ・ファイル `/var/adm/SYSLOG` に記録するオプションがあります。インターネットに接続しているサイトでは、この機能を利用すると便利です。`ftpd`、`tftpd`、および `rshd` のログ機能を有効にするには、`/etc/inetd.conf` を編集し、`ftpd` と `tftpd` の行の最後に **`-l`** を、*rshd* の行の最後に **`-L`** を追加します。これ以外の `ftp` のログをさらに取るには、`ftpd` のエントリに **`-ll`** を追加します。変更後、`inetd` にファイルを再度読込ませます。

```
/etc/killall -HUP inetd
```

`rlogin`、`telnet`、および 4DDN `sethost` プログラムによるリモート・ログインは、`login` でログ・ファイルに記録されます。`/etc/default/login` を編集し、キーワード `SYSLOG=ALL` または `SYSLOG=FAIL` を追加します。たとえば、`login` ファイルにこれらのキーワードが登録されていると、ローカルおよびリモートのログインが正常に行われた場合とそうでない場合を `syslogd` に記録します。

```
syslog=all
```

詳細については、login(1) マン・ページを参照してください。

ネットワーク全体に対するサービスの設定

ネットワーク上の各システムに設定する必須ソフトウェアのほかに、ネットワーク全体にサービスを提供するシステムを設定したい場合があります。たとえば、あるシステムを InSight サーバとして設定しておくことにより、InSight の全マニュアルをローカル・ディスクにインストールする必要がありません。

ここでは、anonymous `ftp` サーバと InSight サーバの設定について説明します。

- 「Anonymous FTP アカウントの設定」（71 ページ）
- 「パスワード保護による FTP アカウントの設定」（76 ページ）
- 「IRIS InSight サーバ／クライアント・システムの設定」（78 ページ）
- 「CD-ROM IRIS InSight サーバ／クライアント・システムの設定」（80 ページ）
- 「リモート IRIS InSight Viewer の起動」（82 ページ）

Anonymous FTP アカウントの設定

anonymous FTP アカウントは、システム上の情報を誰でもアクセスできるようにするための 1 つの方法です。ただし、システムに対するアクセスは制限されています。anonymous FTP アカウントを使用することにより、anonymous または ftp ユーザとして誰でもシステムにログインすることができます。FTP デーモンは、anonymousFTP ユーザに対しては、FTP のホーム・ディレクトリ（`~ftp`）に `chroot` します。これにより `~ftp` のサブディレクトリを除くシステム上のほかのディレクトリにアクセスできないようにします。chroot(1M) のマン・ページを参照してください。ユーザにパスワードを与え、アクセス権を制限する場合には、anonymous アカウントの設定に関する説明の後に、76 ページの「パスワード保護による FTP アカウントの設定」に進んでください。

ネットワークからアクセスできる anonymous FTP アカウントを設定するには、次の手順に従います。システムのセキュリティを確立するには、ここに示す手順を理解し、アカウントがどのように使用されているかを絶えず監視する必要があります。

メモ：IRIX 6.3 または IRIX 6.4 を使用している場合には、特定のファイルを `/lib32` と `/usr` ディレクトリにコピーする必要があります。77 ページの「IRIX 6.3、6.4 および IRIX 6.5 への FTP の更新」を参照してください。これ以外のバージョンの IRIX の場合には、次の手順を実行してください。

1. `/etc/passwd` に anonymous FTP ユーザ・エントリを作成します。ユーザ名を ftp にします。パスワード・フィールドにアスタリスク（*）を入力し、ユーザ ID とグループ ID を割当て、ホーム・ディレクトリとログイン・シェルを指定します。以下の例は、anonymous FTP アカウントに対する `/etc/passwd` の一般的な入力例を示しています。

   ```
   ftp:*:997:995:Anonymous FTP Account:/disk2/ftp:/dev/null
   ```

   ログイン・シェル `/dev/null` を指定しておくことをお薦めしますが、必須ではありません。ホーム・ディレクトリはどこを指定してもかまいませんが、次の条件を満たす必要があります。

   ヒント：パブリック・サーバの場合、Silicon Graphics は shadow パスワード・ファイルを作成することをお薦めします。次のように、`/etc` ディレクトリで `pwconv` コマンドを実行します。

   ```
   # pwconv
   ```

   このコマンドは、`/etc/passwd` の内容を更新し、暗号化されているパスワードを `/etc/shadow` に移動します。これにより権限を所有していないユーザがアクセス不可能になります。NIS を実行している場合の情報については shadow(4) マン・ページを参照してください。

2. anonymous FTP ディレクトリを作成します。任意の場所に作成できますが、書込み許可を与える場合には、`/` や `/usr` とは独立したパーティションに設定します。このようにしておくと、パーティションが一杯になった場合でも、システムの基本的な動作が損なわれません。次の例では、`/disk2/ftp` は anonymous FTP ディレクトリ名になります。まず始めに、ディレクトリを作成します。

   ```
   # mkdir /disk2/ftp
   ```

   独立したディスクまたはディスク・パーティションの場合には、この時点でデバイスをマウントできます。mount(1M) を参照してください。作成する anonymous FTP ホーム・ディレクトリは、`/etc/passwd`ファイルで指定したディレクトリと同じでなければなりません。

3. ディレクトリを ftp ホーム・ディレクトリに変更し、FTP アクセスに使用するサブディレクトリを作成します。

```
# cd /disk2/ftp
# mkdir bin dev etc lib pub incoming
```

標準の bin、dev、etc、lib、および pub ディレクトリのほかに、ファイルを置くための incoming ディレクトリを作成します。

4. ls コマンドを /sbin から ~ftp/bin にコピーします。

```
# cp /sbin/ls bin
```

メモ：IRIX 6.5.x で /s コマンドを使用するには、次の手順に従ってください。

```
# cp /lib32/libc.so.1 /dir2/ftp/lib32
# cp /lib32/rld /dir2/ftp/lib32
# chmod -R 555 /dir2/ftp/lib32
```

5. /etc/passwd と /etc/group を ~ftp/etc にコピーし、ファイルを編集しアクセスに最低必要な権限を指定します。

```
# cp /etc/passwd /etc/group etc
```

~ftp/etc/passwd の内容は、次のように設定します。

```
root:*:0:0:Super-User:/:/dev/null
bin:*:2:2:System Tools Owner:/bin:/dev/null
sys:*:4:0:System Activity Owner:/var/adm:/dev/null
ftp:*:997:999:Anonymous FTP Account:/disk2/ftp:/dev/null
```

~ftp/etc/group の内容は、次のように設定します。

```
sys:*:0:
other::995:
guest:*:998:
```

6. 次のように、anonymous FTPに対して適切なデバイスとライブラリ・ファイルを追加します。

```
# /sbin/mknod dev/zero c 37 0
# cp /lib/libc.so.1 /lib/rld lib
```

dev/zero ファイルは、出力ゼロのセンシティブ・データを補助します。~ftp/bin/ls には、ライブラリ・ファイルを必要とします。

7. ~ftp/etc/passwd、~ftp/etc/group および ~ftp/dev/zero へのアクセス許可を制限します。

   ```
   # chmod 444 etc/* dev/*
   ```

8. bin、dev、etc、lib および ~ftp ディレクトリを、書込みを許可を制限する sys グループの root が所有していることを確認してください。

   ```
   # chown root.sys bin dev etc lib .
   # chmod 511 bin dev etc lib .
   ```

   この chown コマンドのドットは所有者とグループを分けています。

9. pub ディレクトリの場合、所有者を root に、グループを sys に設定し、グローバルな読取りとアクセス許可を指定します。

   ```
   # chown root.sys pub
   # chmod 755 pub
   ```

10. incoming ディレクトリを作成した場合は、そのディレクトリへの書込みのみを許可し、内容の読取りができないように設定します。

    ```
    # chown ftp.other incoming
    # chmod 333 incoming
    ```

    これで FTP ユーザは incoming ディレクトリからファイルを取得したり、そこにファイルを置くことができるようになりました。ただし、ディレクトリの内容をリスト表示することができないため、FTP を実行する前にファイル名を知っておく必要があります。

    注意：書込み許可を与えることにより、anonymous FTP ユーザがディスク・パーテション全体を使用してしまう危険性があります。さらに権限を制限する方法については、76 ページの「パスワード保護による FTP アカウントの設定」を参照してください。

    ---

11. セキュリティを保護するために、次のエントリを /etc/aliases ファイルに追加することにより ftp ユーザにメールが送信されます。

    ```
    ftp: postmaster
    ```

    newaliases コマンドを実行し、これを有効にします。ここでは、ポストマスタのエリアスが /etc/aliases に指定されているものとします。aliases(4) および newaliases(1M) のマン・ページを参照してください。

12. 『IRIX Admin: Backup, Security, and Accounting』の第 5 章「inetd サービスの制限」で説明しているように、FTP ログを有効にします。`ftpd` で **`-l`** 引数を 1 つだけ指定すると、正常終了と異常終了した FTP ログインだけが記録されます。**`l`** を 2 つ指定すると、ftp ログイン・セッション中に実行された retrieve (get)、store (put)、append、delete、make directory、remove directory、rename 操作（およびこれらのファイル名引数）も記録されます。また、**`l`** を 3 つ使用すると、`get` と `put` で転送されたバイト数も記録されます。

    たとえば、`/etc/inetd.conf` の以下のエントリは、FTP セッションの get および put 操作（転送バイト数を除く）を `/var/adm/SYSLOG` に記録します。

    ```
    ftp     stream  tcp     nowait  root    /usr/etc/ftpd   ftpd -ll
    ```

13. `/etc/inetd.conf` の編集が終了したら、次のコマンドで `inetd` を再起動します。

    ```
    # /etc/killall -HUP inetd
    ```

メモ：`/var/adm/SYSLOG` の FTP ログ記録には、ログインしているユーザが入力したパスワードが記録されますが、anonymous FTP に対するパスワード・チェックは行われません。通常、anonymous ユーザはパスワードとして各自の電子メールのアドレスを使用しますが、別のユーザのアドレスなど任意のパスワードを入力することもできます。

`/etc/issue` ファイルに任意のテキストを入力しておくと、だれかがシステムにアクセスしたときに、そのテキストがログイン・プロンプトが表示される前に表示されます。このファイルには、FTP サイトが提供するサービスの種類や、トラブルが発生した場合の連絡先などの情報を記述するとよいでしょう。また、anonymous FTP ユーザがログインすると、`~/ftp/README` ファイルに登録されているテキストが表示されます。

システム・ログ・ファイルの保守の頻度や設定の変更方法については crontab(1)、syslogd(1M) マン・ページ、および `/var/spool/cron/crontabs/root` ファイルを参照してください。たとえば、ログ・ファイルの存続期間を延ばすことができます。パブリック FTP サーバ上で要求された内容を記録するには、システム・リソースの使用状況とシステムの一般的なアカウント情報について知っておくと便利です。前者については『IRIX Admin: Backup, Security, and Accounting』の第 6 章「システム監査トレールの管理」を、後者については『IRIX Admin: Backup, Security, and Accounting』の第 7 章「システム・アカウンティング」を参照してください。

パスワード保護による FTP アカウントの設定

パスワード保護の FTP アカウントは、有効なアクセス権を持つユーザのみにシステムへのアクセスを許可します。これにより、ほかのシステムへのアクセスを制限した上で、指定したユーザにファイルを転送することができます。`/etc/passwd` にパスワードが登録されているユーザは、これらの FTP サービスを使用できます。また、制限された FTP ユーザは、システムの自分のホーム・ディレクトリ以外の場所にアクセスすることができません。システムは、`chroot` を使用して制限します。

ネットワークからアクセスできる安全な FTP アカウントを設定するには、次の手順に従います。システムのセキュリティを確立するには、ここに示す手順を理解し、アカウントがどのように使用されているかを絶えず監視する必要があります。この手順を実行する前に、71 ページの「Anonymous FTP アカウントの設定」の手順を完了している必要があります。

1. `/etc/passwd` の `ftp` ユーザ・エントリを変更します。ユーザ名は、「ftpX」になります。ここで、*X* は変数を意味します。パスワード・フィールドには入力せず、ユーザ ID、グループ ID、ホーム・ディレクトリを入力します。以下にパスワード保護の FTP アカウントの、一般的な `/etc/passwd` エントリの例を示します。

   ```
   ftp1::197:995:FTP user:/disk2/ftp:/dev/null
   ```

   ログイン・シェルの `/dev/null` は、FTP ユーザが UNIX シェルを使用できないようにします。2 つ目のフィールドには、ユーザの暗号パスワードが入ります。

   ヒント：FTP デーモンは、anonymous FTP 権限を `ftp` という名前のユーザに与えます。つまり、`/etc/passwd` に `ftp` 行がない場合には、システムは anonymous FTP を許可しません。

2. FTP ユーザにパスワードを割当てます。

   ```
   # passwd ftp1
   New password:
   Re-enter new password:
   ```

3. `/etc/passwd` から FTP ユーザの行をコピーし、`~ftp/etc/passwd` にペーストします。暗号パスワードの部分をアスタリスクに変更します。

   ```
   ftp1:*:197:995:FTP user:/disk2/ftp:/dev/null
   ```

4. `/etc/ftpusers` ファイルを作成または変更し、以下の行を登録します。

   ```
   ftp1 restrict
   ```

これにより、FTP デーモンはこのユーザをそのホーム・ディレクトリに chroot します。ftpd(1M) マン・ページを参照してください。

5. FTP ユーザのホーム・ディレクトリを ftp/pub に作成します。

   ```
   # mkdir /disk2/ftp/pub/ftp1
   ```

6. FTP ユーザのホーム・ディレクトリに対して、所有権およびグループを *other* に設定し、ユーザに読取り、書込み、アクセス許可を与えます。

   ```
   # chown ftp1.other /disk2/ftp/pub/ftp1
   # chmod 700 /disk2/ftp/pub/ftp1
   ```

   注意：書込み許可については、FTP ユーザがそのディスク・パーティションすべてを使い果たす可能性があるため、~ftp/incoming ディレクトリにのみ与えます。

7. FTPユーザにホーム･ディレクトリの場所を知らせる~ftp/README ファイルを作成します。

   ```
   Your home directory is under pub/.
   ```

8. セキュリティの保護のために、以下のエントリを /etc/aliases に追加し、FTP ユーザに送信されたメールをポストマスタに送信するように設定します。

   ```
   ftp1: postmaster
   ```

   newaliases コマンドを実行して、この変更を有効にします。newaliases(1M) マン・ページを参照してください。

IRIX 6.3、6.4 および IRIX 6.5 への FTP の更新

IRIX 6.3、6.4 および IRIX 6.5 を実行している場合には、実行時ローダを /lib32 ディレクトリにコピーし、/usr/lib/iconv/iconvtab を /usr/iconv にコピーする必要があります。

1. ~ftp/lib32 のディレクトリを作成します。

   ```
   # mkdir ~ftp/lib32
   ```

2. 実行時ローダをそのディレクトリにコピーします。

   ```
   # cp /lib32/rld ~ftp/lib32
   ```

3. ~ftp/usr/iconv のディレクトリを作成します。

   ```
   # mkdir ~ftp/usr/iconv
   ```

4. フォント・ファイルをそのディレクトリにコピーします。

```
# cp /usr/lib/iconv/iconvtab ~ftp/usr/lib/iconv
```

IRIS InSight ファイル・サーバについて

オンライン・マニュアルの InSight システムを構成するファイルとディレクトリには、かなりのディスク領域を必要とします。ネットワーク環境において、すべてのシステムが実質的に同じバージョンの InSight ソフトウェアを使用する場合には、各システムで InSight ドキュメントのコピーを維持する必要はありません。複数のシステム間で同じリビジョン・レベルの IRIS Insight ソフトウェアを使用している場合には、1 つのシステムをほかのシステムに対する InSight ファイルのサーバとして設定することにより、クライアント・システムのディスク領域を節約することができます。

負荷が重いシステムをサーバとして指定しないよう注意してください。また、ネットワークのパフォーマンスも考慮に入れる必要があります。ユーザが InSight を頻繁に使用し、ネットワークにすでに重い負荷がかかっている場合は、InSight とネットワークの両方の応答時間が遅くなります。InSight サーバを通常のワークステーションとして使用する場合は、ディスク領域、CPU、ネットワークにオーバーヘッドがかかります。

InSight サーバを設定するには、2 通りの方法があります。次に、これらの方法について説明します。いずれの方法においてもオプションのソフトウェアである NFS をインストールする必要があります。NFS の用語や概念についてよくわからない場合には、サーバを設定する前に、『ONC3/NFS Administrator's Guide』と『NIS Administrator's Guide』を読むことをお薦めします。ここで説明する 2 番目の方法では、InSight 提供メディアのインストールに CD-ROM ドライブが必要です。

IRIS InSight サーバ／クライアント・システムの設定

IRIS InSight Viewer とドキュメント・ライブラリをリモート・サーバにインストールしてローカル・クライアント・システムから情報を取出すには、次の手順に従います。

サーバ・システム上で次の操作を行います。

1. `root`（特権ユーザ）でログインします。
2. `inst` を起動し、CD-ROM ドライブまたはディストリビューション・ディレクトリから IRIS InSight をインストールします。次のコマンドを入力します。

   ```
   Inst> install insight insight_gloss *.books.*
   Inst> go
   ```

   InSight Viewer とドキュメント・ライブラリの合計サイズは約 23 MB です。
3. 「システム・マネージャ（System Manager）」または `exportfs` コマンドを実行し、`/usr/share/InSight/library/SGI_bookshelves` ディレクトリをエキスポートします。

   ```
   exportfs -i /usr/share/Insight/library/SGI_bookshelves
   ```

   サーバでグラフィックス表示ができない場合には、終了時に警告メッセージが出力されます。この警告メッセージは X サーバのフォント・ディレクトリの更新に関するものであり、無視してください。

クライアント・システム上で次の操作を行います。

1. *root*（特権ユーザ）でログインします。
2. 次のコマンドを実行します。

   ```
   versions remove *.books.*
   ```

   ブックが bookshelves に移動されます。
3. クライアント・システム上で `inst` を起動し、次のコマンドを入力して *insight.sw.client* サブシステムをインストールします。必要に応じて、*insight.man.man* と *insight.man.relnotes* の各サブシステムもインストールします。

   ```
   Inst> keep insight insight_gloss
   Inst> install insight.sw.client
   Inst> go
   ```
4. サーバから、クライアント・システムの `SGI_bookshelves` ディレクトリをマウントします。次の例では、サーバ・マシンの名前を *capra* にしています。

   ```
   mkdir /usr/share/Insight/library/SGI_bookshelves
   mount capra:/usr/share/Insight/library/SGI_bookshelves
   /usr/share/Insight/library/SGI_bookshelves
   ```

クライアント・システム上で mount コマンドを入力する場合には、コマンド行全体を 1 行で入力します。ここでは書式の都合上 2 行に分かれることがあります。

メモ：InSight をリモート・マウントしている間は、Silicon Graphics の desktophelp システムが正常に動作しない場合があります。desktophelp を参照する場合は、Insight を一時的にアンマウントしてください。

CD-ROM IRIS InSight サーバ／クライアント・システムの設定

この方法では InSight 専用の CD-ROM ドライブを用意することにより、InSight ファイルを格納するためのディスク領域を使用しなくてすみます。この場合、InSight の CD をドライブに挿入し、マウントされている InSight の CD を InSight ファイルがインストールされるディレクトリにリンクするだけです。この方法の欠点は CD-ROM ドライブを InSight 専用にしなければならないことですが、NFS と合わせて使えばネットワーク全体からサーバにアクセスすることができます。

メモ：IRIS InSight ドキュメント・ライブラリがクライアント・ディスク上にすでにインストールされている場合に CD-ROM のマニュアルをマウントしたい場合は、/usr/share/Insight/library ディレクトリの全ファイルを削除してから次の 2 の操作を実行してシンボリック・リンクを設定します。versions remove コマンドでマニュアルを削除した後、ディレクトリの中のすべてのファイルが削除されたことを確認します。

CD-ROM 上の IRIS InSight Viewer とドキュメント・ライブラリを使用し、ローカル・システムまたはリモート・システムからマニュアルにアクセスするには、次の手順に従います。

CD-ROM ドライブがあるサーバ・システム上で次の操作を行います。

1. *root*（特権ユーザ）でログインします。
2. IRIS InSight CD を CD-ROM ドライブに挿入します。ドライブがマウントされていない場合は、「システム・マネージャ（System Manager）」または mount コマンドを使用してドライブをマウントします。一般的なマウント・ポイントは /CDROM です。ドライブが正常にマウントできれば、/CDROM にディレクトリを変更して IRIS InSight CD のすべてのファイルを参照できます。ファイルを参照するには、次のコマンドを入力します。

   ```
   cd /CDROM/insight
   ```

3. *root* ユーザで、`/usr/Insight/library/SGI_bookshelves` から CD `insight` ディレクトリへのシンボリック・リンクを作成します。`/usr/Insight/library` ディレクトリが存在しない場合には、このディレクトリを作成します。次のコマンドを入力します。

   ```
   mkdir -p /usr/share/Insight/library
   ln -s /CDROM/insight/SGI_bookshelves /usr/share/Insight/library
   ```

   クライアント・システム上でリンク・コマンドを実行する場合には、コマンド行全体を 1 行で入力します。ここでは書式の都合上 2 行に分かれることがあります。

   この時点で InSight の bookshelves がサーバ上にマウントされ、そのサーバ・システム上で利用できるようになります。ネットワークのほかのシステム上のユーザが bookshelves を利用できるようにしたい場合は、4 に進みます。

4. ネットワーク上のほかのユーザと bookshelves を共有する場合は、「システム・マネージャ（System Manager）」または `exportfs` コマンドで `/CDROM` ディレクトリをエキスポートします。

   ```
   exportfs -i /CDROM
   ```

各クライアント・システムで次の操作を行います。

1. *root*（特権ユーザ）でログインします。

2. `inst` を起動し、次のコマンドを入力して *insight.sw.client* サブシステムをインストールします。必要に応じて、*insight.man.man* と *insight.man.relnotes* の各サブシステムもインストールします。これらは、サーバの `/CDROM/dist` ディレクトリの CD にあります。

   ```
   Inst> keep *
   Inst> install insight.sw.client
   Inst> go
   ```

3. サーバから bookshelves をマウントします。次の例では、サーバ・マシンの名前を *capra* にしています。次のコマンドを入力します。

   ```
   mount capra:/CDROM/insight/SGI_bookshelves
   /usr/share/Insight/library/SGI_bookshelves
   ```

メモ： クライアント・システムで `mount` コマンドを実行する場合には、コマンド行全体を 1 行で入力します。ここでは書式の都合上 2 行に分かれることがあります。

リモート IRIS InSight Viewer の起動

ソフトウェアをインストールしたら、利用可能なプログラムとコマンドのリストを再作成するようにシェルで指示します。次のコマンドを実行します。

```
rehash
```

この後 `iiv` コマンドを入力すると、IRIS InSight Viewer が起動します。`iiv` は、IRIS InSight Viewer の頭字語です。

インターネット・ゲートウェイを介したネットワーク・サービスへのアクセス

IRIX 6.5 より、Internet Gateway ソフトウェアを使用してインターネットへのアクセスを設定できます。つまり、このソフトウェアを使用することによりグラフィカル・ユーザ・インタフェースを使用して、ネットワーク・サービスにアクセスすることが可能になります。使用方法については、オンライン・ヘルプを参照してください。ここでは、概要を説明します。

グラフィカル・ユーザ・インタフェースでは、インターネット・サービス・プロバイダを介したネットワークへの接続だけでなく、FTP、DHCP、あらゆるネーミング・サービス、ネットワーク・サービス・エントリ・スクリーンからの経路制御などを管理することができます。

インターネット・ゲートウェイ・サーバにアクセスするには、Web ブラウザを起動し、構成プロセスを完了するための一連のフォームに必要事項を入力します。

1. サーバが起動していることを確認します。`chkconfig` コマンドを実行してインターネット・ゲートウェイを有効にします。

   ```
   # chkconfig webface on
   ```

2. インターネット・ゲートウェイ・サーバを起動します。

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

   Web ブラウザは、インターネット・ゲートウェイ・サーバと同じ LAN 上であれば、どのコンピュータまたはワークステーションからでも起動することができます。

3. 次のユニフォーム・リソース・ロケータ（URL: uniform resource locator）を開きます。

   http://*server_ip_address***:2077/**

この URL では、インターネット・ゲートウェイ・サーバに割当てた IP アドレスは、*server_ip_address* になります。次の例を参照してください。

```
http://151.166.96.36:2077/
```

管理アカウント・ユーザ ID の入力と、この管理アカウントのパスワードを作成することを求めるメッセージが表示されます。root パスワードも入力する必要があります。

オンライン・インストラクションに従って、インターネット・ゲートウェイとして動作するようサーバを設定します。

RSVP によるリソースの予約

IRIX 6.5 の起動時に、RSVP（Resource Reservation Protocol）は、帯域幅を予約することによりアプリケーションがコンピュータ・ネットワークを介して経路を設定できるようにします。これは、ビデオ会議、ビデオ・ブロードキャスト、インターネット電話などのアプリケーションには特に有用であり、現状のネットワーク負荷にかかわらず特定の帯域幅を保証することになります。

RSVP のために定義されている統合サービスには制御ロードと保証の 2 つのタイプがありますが、IRIX 6.5 では、制御ロードのサービスのみをサポートしています。このサービスは、パケットが最小の損失と遅延で配信するようにします。経路にしたがって、各ホストよびルータには、現在の設定およびその他の要因によりサービスを提供するかどうかを決定するオプションがあります。

アプリケーションは、X/Open で定義される API を使用して IRIX システム・デーモンの `rsvpd` と通信します。その後、`rsvpd` デーモンは、アプリケーションの代わりに RSVP 制御メッセージを受信したり送信します。`rsvpd` は、1 つまたは複数の統合サービスが使用可能であることを示す PATH メッセージを送信し、PATH メッセージを受信したシステムは RESV メッセージを戻して帯域幅を予約します。IRIX カーネルのパケット・ハンドラは RSVP セッションに属するパケットを識別し、制御ロード仕様により定義されるサービスを提供します。パケット・ハンドラは、高い回線利用と低い CPU オーバーヘッドを保ちながら、これらすべてを実行します。

RSVP のインストール

RSVP を使用する前に、適切なドライバがシステムにインストールされており、そのシステムにサポートされているネットワーク・インタフェースがあることを確認してください。システムのネットワーク・インタフェースを確認するには、hinv コマンドを使用します。

```
% hinv
```

表 3-2 に示す、出力が表示されます。

表 3-2 RSVP でサポートされているカード

ネットワーク・インタフェースのタイプ	名称	hinv からの出力（ただし *n* は整数値）
FDDI	xpi	xpi*n*
FDDI	ipg	ipg*n*
FDDI	rns	rns*n*
イーサネット	me	ec*n*
イーサネット	ec2	ec*n*
イーサネット	ef	ef*n*
イーサネット	ecf	ecf*n*
イーサネット	ee	et*n*
イーサネット	ep	ep*n*

サポートされているハードウェアがインストールされている場合には、正しいバージョンの IRIX オペレーティング・システムがインストールされていることを確認します。現在インストールされているソフトウェアは、デスクトップの「ソフトウェア・マネージャ」を使用するか、コマンド行から `showprods` を使用して調べることができます。ソフトウェアをインストールする必要がある場合には、『IRIX Admin: Software Installation and Licensing』の説明を参照してください。

システムが RSVP を使用できるように設定されている場合には、RSVP デーモンの `/usr/etc/rsvpd` を起動します。システムの起動時に自動的に `rsvpd` を起動するには、`chkconfig` コマンドを使用します。

```
# chkconfig rsvpd on
```

これにより、次回よりシステムの起動時に RSVP 機能が有効になります。コマンド行オプションに関する詳しい説明は、rsvpd(1M) マン・ページを参照してください。

ネットワークの帯域幅の予約状態は、`rstat` コマンドを使用して監視することができます。このコマンドを実行すると、各インタフェースを行ごとに、セッション・アドレスとポート、次のトップ・アドレスを表示します。セッションに予約が存在していない場合には、「`no resv`」が表示されます。詳しい説明およびオプションについては、rstat(1M) マン・ページを参照してください。

`psifconfig` コマンドを使用することにより、特定のインタフェース用に RSVP を設定することができます。これにより、そのインタフェースに対して予約可能な帯域幅を設定し、現在予約されている帯域幅を調べたり、そのインタフェースでの RSVP の無効化およびリストアを設定することができます。詳しい説明およびオプションについては、psifconfig(1M) マン・ページを参照してください。

グラフィカル形式で `rsvpd` を設定し、現状の RSVP ステータスを監視したい場合には、インターネット・ゲートウェイを使用することができます。ただし、このソフトウェアがインストールされている必要があります。82 ページの「インターネット・ゲートウェイを介したネットワーク・サービスへのアクセス」を参照してください。この場合、次のオプションを使用することができます。

- `chkconfig` による RSVP 設定のオンまたはオフ
- そのセッションが `rstat` で表示され予約されているのかの判断
- `psifconfig` による特定のインタフェースでの RSVP の設定
- デバッグ用のログ・データを制御するためのログ・レベルの設定

RSVP は、インターネット・ゲートウェイを使用しなくてもコマンド行から実行することができます。

RSVP のトラブルシューティング

RSVP は、非常に新しい技術であるため、トラブルシューティングが必要になります。以下に、問題と対処方法を示します。

ファイル転送が遅い

ネットワークでのファイル転送速度が遅くなった場合、RSVP が通常のファイル転送を妨害している可能性があります。この場合、次のコマンドを使用してまず RSVP をオフに設定します。

```
# killall -TERM rsvpd
```

ファイル転送が標準の速度に戻ったら、RSVP を再起動します。

ビデオ妨害

RSVP および RSVP 対応ビデオ・アプリケーションを使用している場合、通常スムーズで途切れずにビデオ・ピクチャを配信します。ビデオ・ピクチャが正しく配信されない場合には、`rstat` を実行してこのマシンにある予約状況を確認します。出力に関する詳しい説明は rstat(1M) マン・ページを参照してください。

イーサネット接続のトラブルシューティング

ここでは、イーサネットで起こる一般的な問題について説明します。

- 「ケーブル問題に対するトラブルシューティング」（86 ページ）
- 「遅延衝突問題に対するトラブルシューティング」（87 ページ）
- 「パケット・サイズ問題に対するトラブルシューティング」（88 ページ）
- 「サーバ接続問題に対するトラブルシューティング」（89 ページ）

『IRIX Admin: System Configuration and Operation』の付録 B「システム・エラー・メッセージを使用したシステム構成に関するトラブルシューティング」も参照してください。

ケーブル問題に対するトラブルシューティング

イーサネットのエラーで一般的なものは、プラグに差し込まれていないケーブルや緩んだケーブルです。

```
unix: <ethernet_device>: no carrier: check Ethernet cable
```

このメッセージは、送信しようとしたときにキャリアが検出されなかったことを示します。このエラーを訂正するには、次の操作を行います。マシンの背面のプラグにイーサネット・ケーブルがしっかりと接続されているかどうか確認します。ケーブルの接続については、ハードウェアのマニュアルを参照してください。ケーブルの接続や取外しの際に、システムを停止する必要はありません。ケーブルを再接続した後に、ネットワークの接続をテストします。

トランシーバ・ケーブルがマシンの背面やトランシーバ・ボックスにしっかりと接続されている場合は、ほかの原因を調べます。

- マシンのトランシーバを調べます。
- トランシーバ・ケーブルを調べ、ほかのマシンで正常に動作しているほかのケーブルと取替えます。
- 10baseT のハブに問題がないかどうか調べます。
  - ハブの別のポートを試してみます。
  - 異なるケーブルを試してみます。
  - 異なるハブを試してみます。
- 何も問題がないのにネットワークにアクセスできない場合は、サービス部門にお問い合わせください。ネットワーク自体が一時的に使用不能になっている可能性があります。

このエラーの詳細については、ethernet(7) マン・ページを参照してください。

遅延衝突問題に対するトラブルシューティング

ケーブルに関する別の問題として、遅延衝突があります。遅延による衝突はエラーになり、エラーの詳細情報は *syslog* に記録されます。

コントローラがパケットを送信しようとしたが、別のマシンから遅延衝突の信号が出された、ということがよくあります。通常、これはイーサネット・ケーブルの配置に問題があります。このエラーの原因として最も多いのはケーブルが長すぎること、またはケーブルの接続が緩んでいることです。

エラーの起こった個所を特定するには、まずイーサネットのケーブル制限長を超えていないかどうか調べます。10BASE5 ケーブル（thicknet）の場合、セグメントの最大長は 500m です。10BASE2 ケーブル（thinnet）の場合、セグメントの最大長は 200m です。トランシーバ・ケー

ブル（マシンのイーサネット・ポートとそのトランシーバを結ぶケーブル）の長さは、50 メートル以内です。すべてのケーブルが規定範囲内にあり、すべてのコネクタがしっかりと接続されている場合は、このエラーの原因はイーサネット・コントローラまたはトランシーバのハードウェア・トラブルです。次にその対策を示します。

- ネットワーク上にあるほかのマシンのイーサネット・コントローラに問題がないかどうか調べます。
- ネットワーク上にある別のマシンのトランシーバに問題がないかどうか調べます。問題がある場合は、別のシステムで正常に動作しているトランシーバと取替えます。
- ネットワークから特定のマシンをはずし、状況に変化があるかどうか、問題が起こらなくなるかどうか調べます。

パケット・サイズ問題に対するトラブルシューティング

問題のあるイーサネット・コントローラがネットワーク上にあると、送出パケット・サイズが大きすぎるまたは小さすぎるというエラー・メッセージが表示されます。

```
unix: <ethernet_device>: packet too small (length = <packet size>)
unix: <ethernet_device>: packet too large (length = <packet size>)
```

この例では、イーサネット・コントローラが、イーサネットで使えるパケットの最小サイズに満たないパケット、またはパケットの最大サイズを超えたパケットを受信したことを示しています。これは、別のマシンのコントローラまたはトランシーバに問題があります。次にその対策を示します。

- ネットワーク上にあるほかのマシンのイーサネット・コントローラに問題がないかどうか調べます。
- ネットワーク上にある別のマシンのトランシーバに問題がないかどうか調べます。問題がある場合は、別のシステムで正常に動作しているトランシーバと取替えます。
- ネットワークから特定のマシンをはずし、状況に変化があるかどうか、問題が起こらなくなるかどうか調べます。

このエラーの詳細については、ethernet(7) マン・ページを参照してください。

サーバ接続問題に対するトラブルシューティング

システムがネットワーク・システムと接続できない場合、次のようなメッセージが表示されます。

```
portmapper not responding: giving up
```

この問題は次のいずれかの状況で発生します。

- システムが動作していない場合

  直接システムを調べるか、またはそのユーザまたはシステム管理者にそのシステムの電源が入っていて正常に動作しているかどうか確認してもらいます。

- ネットワークが動作していない場合

  ネットワーク上にある別のシステムにアクセスできるかどうか確認します。

- ネットワーク管理者がシステムに関する特定の情報またはネットワーク上でのシステムの論理的位置を変更した場合

  何か原因なのかを特定し、必要な情報を集めて問題を解決します。

- システムのリソースを使用しているユーザやシステムの数が多すぎるためサービスを提供できない場合

  システムの主要ユーザや管理者に状況を調べてもらいます。

一般的に、次のようなメッセージが表示されます。

- `Host unreachable`：通常、構成エラーを示します。ローカル・ネットワークが IP アドレスをイーサネット・アドレスに変換する `arp` ブロードキャストで発生します。
- `Network unreachable`：通常、デフォルトまたはネットワーク・ルートがルーティング・テーブルにないことを示します。

ネットワーク・インタフェースごとの情報の表示

システムに複数のネットワーク・インタフェース（FDDI リンクや SLIP 接続などのイーサネット・ボード）がある場合でも、ネットワークに関する情報をインタフェースごとに簡単に確認することができます。

各インタフェースの情報を出力するには、netstat -in コマンドを使用します。次に示すのは、このコマンドの出力例です。

```
Name Mtu   Network    Address    Ipkts Ierrs Opkts Oerrs Coll
ec0  1500  128.70.0   128.70.0.9  15    0     15    0     24
ec1  1500  128.70.2   128.70.2.5  15    0     15    0     24
sl0  1006  (pt-to-pt) 128.70.0.9  0     0     0     0     0
lo0  8304  loopback   localhost   8101  0     8101  0     0
```

2 番目のイーサネット接続は、ネットワーク 128.70.2（1 行目のイーサネット接続とは異なる LAN）に対するものです。2 番目の LAN 上にあるローカル・ステーションのアドレスは 128.70.2.5 です。この接続をテストするには、そのネットワーク上にある別のステーションに接続できるかどうかを ping コマンドで調べます。

この例では、SLIP リンクも実行しています。SLIP リンクは、ec0 と同じ LAN を別の場所にあるシステムに拡張しています。この SLIP リンクをテストするには、リンクのもう一方の終端にあるステーションのホスト名または IP アドレスを調べ、接続可能かどうかを ping コマンドで調べます。

lo0 インタフェースは、ローカル・ホスト上のループバック・ネットワーク・インタフェースです。

どのハードウェアがどのインタフェースを制御してかを調べるには、次のコマンドを実行します。

```
ls -lS /hw/net/XXN
```

N は、タイプ *XX* のネットワーク・インタフェース番号になります。次の例を参照してください。

```
% ls -lS /hw/net/ef0
crw-------    1 root     sys            0 6月  6日 14時14分 /hw/net/ef0 ->
/hw/module/4/slot/io1/baseio/pci/2/ef
```

ネットワーク管理の概要

この章では、インストール後のネットワーク管理について簡単に説明します。まず、参考文献を示し、次にネットワークの起動プロセスと停止プロセスについて説明します。続けて、基本的ないくつかのネットワーク管理ツールと、ネットワーク統計情報の解釈の仕方について説明します。この章では、以下について説明します。

ネットワーク管理についての参考文献

ネットワーク管理はきわめて複雑な作業です。制御や調整について広範囲におよぶ知識が必要です。このマニュアルで説明されていない基本的な情報を得るためには、次の文献を参照してください。

- 『DNS and BIND』Paul Abitz、Cricket Liu 共著、第 2 版、O'Reilly and Associates 発行。
- 『Internetworking with TCP/IP Volume 1』Douglas E. Comer 著、第 3 版、Prentice Hall 発行、1994 年。
- 『LAN Management with SNMP and RMON』Gilbert Held 著、John Wiley and Sons 発行、1996 年。
- 『Managing Internetworks with SNMP』Mark Miller 著、M & T Books 発行、1993 年。
- 『Managing NFS and NIS』Hal Stern 著、第 1 版、O'Reilly and Associates 発行、1991 年。

- 『Network Management - A Practical Perspective』Allan Leinwand、Karen Fang Conroy 共著、第 2 版、Addison Wesley 発行、1996 年。
- 『Routing in the Internet』Christian Huitema 著、Prentice Hall 発行、1995 年。
- 『System Performance Tuning』Mike Loukides 著、第 1 版、O'Reilly and Associates 発行、1990 年。
- 『TCP/IP Network Administration』Craig Hunt 著、第 1 版、O'Reilly and Associates 発行、1992 年。
- 『TCP/IP Illustrated, Volume 1』W. Richard Stevens 著、Addison Wesley 発行、1994 年。

この分野の情報は日進月歩で変化しているので、参考図書として、関係する白書や最新の雑誌も参照してください。

ネットワーク管理機能

ネットワーク管理は、ネットワーク・イベントの保守、モニタリング、および制御から、問題の解決にいたるまで広範囲にわたります。ネットワーク管理ツールと管理方針における必須の要素は、以下のとおりです。

- フォールト管理 — 管理デバイスが故障したときに、これを交換したり修理します。また、ネットワーク・エラー・ログを記録し、故障したコンポーネントを交換します。故障を検出するいくつかの方法については、106 ページの「低いネットワーク・パフォーマンスの改善」を参照してください。
- アカウンティング管理 — ネットワーク・リソースのユーザを識別し、部門ごとに料金を課金します。このようなネットワーク管理については、『IRIX Admin: Backup, Security, and Accounting』を参照してください。
- 構成および名前管理 — ネットワーク・ソフトウェアおよびハードウェアを計画、初期化、および更新します。その内容の一部については、これまでの章で説明されています。
- パフォーマンス管理 — ネットワーク・コンポーネントが求められるとおりに動作し、ネットワークが有効な帯域幅を適切に使用することを確認します。場合によっては、プロトコルを作成し直し、管理ツールを開発します。これらの内容の多くについては、関係するアプリケーションのマニュアルに説明されています。『Performance Co-Pilot for ORACLE Administrator's

Guide』、『Developer Magic: Performance Analyzer User's Guide』、および『IRIS HIPPI API Programmer's Guide』などがその例です。

- セキュリティ管理 — 管理オブジェクトへの侵入や誤用を防ぎます。セキュリティ関連については、『IRIX Admin: Backup, Security, and Accounting』を参照してください。

多くのネットワークはハイブリッド設計であり、複数のプロトコルと各種のデバイスを備えています。このため、機能を管理し混乱を避ける上で、ネットワーク管理が各種要素を結合する唯一の要因です。ネットワークの起動と停止に関する以下の節で、ネットワークの操作の概要を説明します。

- 「ネットワークの初期化プロセス」（94 ページ）
- 「ネットワークの停止プロセス」（95 ページ）

ネットワークの起動と停止

ネットワーク関連の中心的なスクリプトは、`/etc/init.d/network` です。ほかのネットワーク・アプリケーション（UUCP、メールなど）に対するスクリプトもこのディレクトリにありますが、それらのスクリプトについては別の章で説明します。ここでは、`network` スクリプトについて簡単に説明します。

`network` マスター・スクリプトは、システムの起動時と停止時に呼出されます。このスクリプトはシステム名とホスト ID を定義し、システムに有効なインターネット・アドレスを設定し、ネットワーク・デーモンを起動し、ネットワーク・インタフェースを初期化します。ローカル・デーモンの起動と停止、静的経路の追加、ARP エントリの登録を行うサイト固有の構成コマンドは、`/etc/init.d/network.local` という別のシェル・スクリプトで事前に定義します。`/etc/rc0.d` と `/etc/rc2.d` から `/etc/init.d/network.local` にシンボリック・リンクを設定することで、システムの起動時と停止時に `network.local` ファイルが呼出されます。この設定の手順については、70 ページの「ローカル・ネットワーク・スクリプトの作成」を参照してください。現在、この機能は `/etc/config/static-route.options` により実行されます。

`network` マスター・スクリプトは `/etc/rc0.d/K40network` に対してリンクが設定されており、これはシステムの停止時に `/etc/rc0` から起動されます。また、`/etc/rc2.d/S30network` に対してもリンクが設定されており、これはシステムの起動時に `/etc/rc2` から起動されます。

このスクリプトは、**start** と **stop** という 2 つの引数を認識します。このスクリプトはシステムを再起動しなくても手動で実行でき、ネットワークに関連した問題を調べてそれを解決するために使用します。

ネットワークの初期化プロセス

システムの初期化中に、シェル・スクリプト `/etc/init.d/network` が呼出されます。システムの起動時に、スクリプトによって次の動作が実行されます。

1. ホスト名とインターネット・アドレスを調べ、システムをスタンドアロンとして構成するのか、ネットワークとして構成するのかを判断します。`sys_id` ファイルと `hosts` ファイルを確認します。**network** 構成フラグがオフの場合、システムはスタンドアロンとして構成されます。
2. 一般的な構成に対する名前とアドレス、または主要インタフェースとルータ・インタフェースを決定します。
3. `netif.options` ファイルからインタフェースに対するサイト固有の情報を取得します。
4. システムがディスクレスでない場合は、シェル・スクリプトがこれまで使用していたすべての経路をクリアします。
5. ループバックを含むすべてのインタフェースを `ifconfig` コマンドで構成します。
6. IP パケットのフィルタリングが構成されている場合は、シェル・スクリプトが IP パケット・フィルタリング・デーモン（`/usr/etc/ipfilterd`）を起動します。`ipfilterd` デーモンは、ゲートウェイ・インタフェースを初期化する前に起動する必要があります。
7. ゲートウェイ・インタフェースを初期化します。
8. `netif.options` ファイルで指定されているほかのインタフェースを初期化します。
9. スクリプトで指定されている場合は、`ifconfig-hy.options` ファイルに従って Hypernet インタフェースを初期化します。
10. ループバック・インタフェースを初期化します。
11. `chkconfig` コマンドで、デーモンの構成を調べ、該当するデーモンの設定ファイル（`*.options`）を読込みます。
12. すべての IP マルチキャスト・パケットに設定されているデフォルトの経路を主要インタフェースとして設定します。

13. NIS ソフトウェアが構成されている場合は、NIS ドメイン名を定義し、設定します。
14. NIS ソフトウェアが構成されている場合は、適切な NIS デーモンを起動します。
15. NFS ソフトウェアが構成されている場合は、適切な NFS デーモンを起動し、`/etc/fstab` にリストされている NFS ファイルシステムをマウントします。
16. NFS ソフトウェアが構成されていて、`chkconfig` で `autofs`、`cachefs`、または `lockd` がオンに設定されている場合は、ほかの NFS デーモンを起動します。
17. `chkconfig` でオンに設定されている場合、標準的なデーモン（`inetd`、`timed`、`timeslave`、`rarpd`、`rwhod`、`snmpd` など）を起動します。

ネットワークの停止プロセス

システムを停止すると、`/etc/init.d/network` によってデーモンが停止し、ネットワーク・デバイスが使用できなくなります。システムの停止時に、スクリプトによって以下の動作が実行されます。

1. シェルに関連付けられているすべてのネットワーク・サービス（`rlogind`、`rexecd`、`rshd`、`ftpd`、`telnetd` など）を強制終了します。
2. 直ちに一部のネットワーク・デーモン（`inetd`、`bootp`、`tftpd`、`snmpd` など）を強制終了します。
3. NFS が実行中の場合は、リモート・ファイルシステムをアンマウントします。
4. すべてのリモート・デーモンを強制終了します。
5. NFS が実行中の場合は、エキスポートされているファイルシステムをアンエキスポートします。オプションの NFS ソフトウェアの詳細については、『ONC3/NFS Administrator's Guide』と『NIS Administrator's Guide』を参照してください。
6. 最後まで有効にしておく必要があったデーモン（`portmap`、`slip`、`ipfilterd`）を強制終了します。
7. システムが FDDI リングで接続されている場合は、リングから外します。
8. NIS の `ypbind` プロセスを停止します。

ネットワーク管理ツール

ここでは、日常的なネットワーク管理に使用する最も標準的なネットワーク・ツールについて説明します。特に明記しないかぎり、標準のネットワーク・ツールは `/usr/etc` ディレクトリにあります。ツールの詳細については、オンラインのマン・ページをそれぞれ参照してください。ここでは、ifconfig(1M)、netstat(1)、arp(1M)、rpcinfo(1M)、ping(1M)、spray(1M)、rtquery(1M)、traceroute(1M)、route(1M)、rup(1C)、ttcp(1)、および netsnoop(1M) について簡単に説明します。また、オプションの Silicon Graphics ネットワーク管理ツールである NetVisualyzer についても説明します。

ifconfig(1M)
: `ifconfig` コマンドは、TCP/IP に対するネットワーク・インタフェースを設定または確認します。また、各インタフェースに対して IP アドレス、サブネット・マスク、ブロードキャスト・アドレスを割当てます。`ifconfig` は、マスター・ネットワーク構成スクリプトの `/etc/init.d/network` によって起動時に実行されます。`ifconfig-#.options` ファイルの変更例については、第 3 章「ネットワークの設定」を参照してください。`ifconfig` を使用して各種インタフェースを構成する例については、Hunt 著の『TCP/IP Network Administration』を参照してください。また、このコマンドの各種オプションの詳細については、ifconfig(1M) マン・ページを参照してください。

netstat(1)
: `netstat` コマンドはよく使用されるコマンドで、構成されているネットワーク・インタフェースや、実際に使用できるネットワーク・インタフェースを表示します。また、宛先への有効経路を使用できるかどうかも表示します。`netstat` を使用すると、待ち行列情報（**`-iq`**）、ネットワーク・メモリ（**`-m`**）、プロトコル（**`-p`**）に関する情報を表示できます。

  `netstat` のサンプル出力については、103 ページの「netstat によるネットワーク統計情報の収集」 を参照してください。Hunt 著の『TCP/IP Network Administration』には、`netstat` の使用例が数多く示されています。衝突とネットワークの集中度を計測する上で、`netstat` を使用して分析を行う方法については、Stern 著の『Managing NFS and NIS』を参照してください。また、このコマンドの各種オプションの詳細については、netstat(1) マン・ページを参照してください。

arp(1M)
: `arp` コマンドは、IP アドレスとイーサネット・アドレスとの間の動的変換を管理する ARP テーブルの内容を表示します。このコマンドを使用すると、正しくない IP アドレスで構成されているローカル・ネット上のシステムを検出できます。

arp コマンドには、すべてのエントリを表示する **-a**、エントリを削除する **-d**、エントリを登録しそのエントリのサーバとして動作する **-s**、/dev/kmem.arp ではなく指定されたファイルから情報を取出す**-f**などのオプションがあります。これらのオプションの使用については、arp(1M) マン・ページを参照してください。

arp コマンドを使用して、不適切なアドレス解決に対処する例については、Hunt 著の『TCP/IP Network Administration』を参照してください。arp は、ローカル・ステーションのイーサネット・アドレスを表示しません。ローカル・ステーションのイーサネット・アドレスを取得するには、netstat コマンドに **-ia** オプションを指定して使用します。arp を使用して断続的な故障を診断する方法については、Stern 著の『Managing NFS and NIS』を参照してください。

rpcinfo(1M)

rpcinfo コマンドは、portmapper でリモート・プロシージャ・コール（RPC: Remote Procedure Call）サーバとそのレジストレーションを照会します。rpcinfo は、リモート・マシンが RPC 要求に応答できるかどうかを確認するという意味で、ping に似ています。rpcinfo を使用すると、応答のないサーバ、RPC クライアント / サーバの不適合、RPC サービス関連のブロードキャスト問題を検出し、デバッグできます。rpcinfo が提供する情報には、rpc ベースのアプリケーション（portmapper、NIS、rstatd など）のリスト、プログラム番号、バージョン番号、プロトコル（TCP/UDP）、および関連するポート番号などがあります。RPC ベースのネットワーク・アプリケーションの実行時に、リモート・ステーションから応答が得られない場合は、*rpcinfo* ツールを使用して、リモート・ステーションが問題のアプリケーションをサポートしているかどうかを確認します。RPC メカニズムの詳細については、Stern 著の『Managing NFS and NIS』を参照してください。使用できるオプションの詳細については、rpcinfo(1M) マン・ページを参照してください。

ping(1M)

ping コマンドは、リモート・ホストが稼動していて、自分のシステムからアクセスできるかどうかをテストします。ping はインターネット制御メッセージ・プロトコル（ICMP: Internet Control Message Protocol）に基づいており、ECHO_RESPONSE を要求する ECHO_REQUEST を送信することによって、双方向のストリームを確立します。ping は、パケット損失と往復時間に関する全体的な情報を表示します。

ping を使用して基本的な接続性をテストする方法については、Hunt 著の『TCP/IP Network Administration』を参照してください。ping を使用してネットワーク状態を診断するケース・スタディと、結果の分析方法については、Stern 著の『Managing NFS and NIS』を参照してください。

また、100 ページの「ping によるネットワーク接続のテスト」と ping(1M) マン・ページも参照してください。

spray(1M)

`spray` ユーティリティは、固定長の連続パケットを使用して、ステーションに対して一方向のパケット・ストリームを送信します。そして、リモート・ホスト上の `rpc.sprayd` デーモンに対してリモート・プロシージャ・コールを行い、受信したパケット数と伝送速度を報告します。`spray` は、個別のホストと、ホスト間のネットワーク・ハードウェアの両方について、ネットワーク・インタフェース容量を大まかに判断します。ネットワークの全体的なパフォーマンスについては限られた情報しか提供されませんが、`spray` を使用してイーサネット・インタフェース容量を判断する例については、Stern 著の『Managing NFS and NIS』を参照してください。spray(1M) のマン・ページも参照してください。

rtquery(1M)

特定のステーションに対して、そのステーションのネットワーク経路テーブル（`routed` または `gated`）に関する情報を求める要求を送信します。このツールは、経路に関する問題を解決するのに効果的です。rtquery は、返された経路応答パケット数を表示します。詳細については、『RFC 1058-Routing Information Protocol』と rtquery(1M) マン・ページを参照してください。

traceroute(1M)

ネットワークを通じて送受信されるパケットをトラッキングします。このツールは、大規模な異機種ネットワーク上でのネットワーク障害やルータ障害を解決するのに効果的です。`traceroute` は、インターネット・プロトコルの有効期間（TTL: Time-To-Live）フィールドをサポートする、すべての中間ルータの名前とアドレスを表示します。また、パケットがルータに到達し、ルータを経由し、ルータを離れるまでの時間も表示します。*traceroute* を使用するとネットワークの負荷が増加するので、`traceroute` でネットワークを調べる場合はこの点を考慮します。このコマンドのオプションとサンプル出力については、traceroute(1M) マン・ページを参照してください。

route(1M)

ネットワーク・ルーティング・テーブルにエントリを追加または削除するには、`route` コマンドを使用します。通常、ルーティング・テーブルは、`routed` デーモンまたは `gated` デーモンによって自動的に処理されます。しかし、

route を使用すると、静的なルーティング・テーブルの作成、保守、削除に加えて、ルーティング・テーブルのクリア、経路についての距離情報の表示を行うことができます。起動時に静的経路を組込むには、/etc/gateways ファイルと /etc/config/routed.options ファイルを変更します。

route コマンドを使用して静的ルーティング・テーブルを作成する例については、Hunt 著の『TCP/IP Network Administration』を参照してください。

rup(1C)

Sun RPC ブロードキャストを使用して、リモート・ステーションに関するステータス情報を表示します。この情報には、アップタイムやロード・アベレージが含まれます。特定のステーションが指定されていない場合は、ブロードキャストを使用してローカル・ネットワーク上のステーションに関する情報を表示します。この場合、ブロードキャストはルータを通りません。このツールは、ステーションやネットワークに関する物理的な問題を特定するのに効果的です。

ttcp(1)

伝送制御プロトコル（TCP: Transmission Control Protocol）とユーザ・データグラム・プロトコル（UDP: User Datagram Protocol）のパフォーマンスをテストします。このツールは、標準的なテスト（spray、rup、ping）よりも、パフォーマンスを現実的に測定します。また、伝送経路のローカル終端とリモート終端の両方でパフォーマンスを測定します。101 ページの「ttcp によるネットワーク・スループットの測定」を参照してください。

netsnoop(1M)

パケットを取得し、ネットワーク・トラフィックをデコードし、場合によってはネットワーク・スニッファとして動作します。ファイルに対するトラフィックを取得することも、2 台のマシン間のリアルタイム・パケットを監視することもできます。2 台のマシン間のすべてのトラフィックを取得、または監視するには、3 番目のマシン上で netsnoop を実行する必要があります。このように 3 番目のマシンから監視することで、送信側が回線に対して送り出したと考えていて、実際には送り出していないという状況や、受信側が回線からの受信に失敗したという状況による誤差を避けることができます。

netsnoop は、リアルタイムでいくつものフィルタと共に使用できます。また、保存されているトレースファイルを netsnoop コマンドで再起動できます。ローカル・ネットワーク・インタフェース上で netsnoop を実行するには、特権ユーザでなければなりません。netsnoop は、損傷を受けたパケットを分析したり、問題をトレースする上で効果的なツールです。ネットワークの

過負荷状態が指摘された場合は、NetVisualyzer（下記参照）に含まれる netcollect などのツールを使用します。これらのツールは、サブネット化が必要かどうかを判断することがその目的です。

NetVisualyzer は、Silicon Graphics システム上でオプションとして使用できるネットワーク管理ツールです。これは受動的なネットワーク管理製品で、グラフィック形式のトラフィック・モニタリング、診断、計画、およびパフォーマンス分析用のツールが含まれています。そして、イーサネットや FDDI ネットワークに関するネットワーク情報と統計情報を、直観的なビジュアル形式で表示します。NetVisualyzer を構成するツールは、NetLook、NetGraph、NetCPA、Analyzer、RouteQuery、および TraceRoute の 6 種類です。NetVisualyzer によって、ネットワークの表示とモニタ、ネットワーク統計情報の収集とその統計情報に基づくレポートの作成、異質パケットのレイヤごとのデコードを行うことができます。

ネットワーク統計情報の解釈

ネットワーク管理ツールは、ネットワークに関する有用な情報をネットワーク管理者に提供します。しかし、これらの統計情報の表示形式は分かりにくい場合があります。ここでは、代表的な 3 つの管理ツールの使い方と、それらのツールによって作成されるネットワーク統計情報の解釈の方法について説明します。

- 「ping によるネットワーク接続のテスト」（100 ページ）
- 「ttcp によるネットワーク・スループットの測定」（101 ページ）
- 「netstat によるネットワーク統計情報の収集」（103 ページ）

ping によるネットワーク接続のテスト

オプションが指定されていない場合、`ping` ツールは、マシンがアクティブで、ネットワークを介してアクセスできるかどうかを確認します。`ping -s` を使用すると、指定サイズのデータグラム、伝送時間（往復時間）、パケット損失率を表示できます。`ping` を使用してネットワーク問題を特定する例については、Stern 著の『Managing NFS and NIS』を参照してください。

カウントを表す **`-c`** オプションを指定しないかぎり、`ping` は停止されるまで連続してデータグラムを送信します。このため、これをスクリプトの中では使用しないでください。`ping` を使用してトラブルシューティングを行う場合は、まずローカル・ホストから始めてローカル・ネット

ワーク・インタフェースが正しく動作することを確認します。次に、テストの範囲をリモート・ホストやゲートウェイに広げます。

以下の例では、**-c** オプションを指定して ping を実行することで、送受信するデータグラムの個数を制限しています。ping ツールは、ローカル・ステーションと *testcase* というステーションとの間のトラフィックをテストし、測定します。ほかの ping オプションの詳細については、ping(1M) マン・ページを参照してください。

```
/usr/etc/ping -c5 testcase

PING testcase (192.55.43.4): 56 data bytes

64 bytes from 192.55.43.4: icmp_seq=0 ttl=249 time=160.314 ms
64 bytes from 192.55.43.4: icmp_seq=1 ttl=249 time=47.057 ms
64 bytes from 192.55.43.4: icmp_seq=2 ttl=249 time=28.129 ms
64 bytes from 192.55.43.4: icmp_seq=3 ttl=249 time=48.596 ms
64 bytes from 192.55.43.4: icmp_seq=4 ttl=249 time=131.894 ms

----testcase PING Statistics----

5 packets transmitted, 5 packets received, 0% packet loss

round-trip min/avg/max = 28.129/83.198/160.314 ms
```

ping が何も出力しない場合は、ping の対象であるホストが切断されているか、停止しています。パケット損失率が一貫して 0.1% を超える場合は、詳しく調べる必要があります。パケットの損傷は、場合によってはハードウェアの障害が原因です。ケーブル接続、コネクタ、端子をよく調べてください。パケット損失はときどき発生するので、指定の間隔で ping を実行するスクリプトをファイルに保存しておくと便利です。ping が応答しても、依然として ftp や telnet を使用できない場合は、TTL エラーが発生している可能性があります。-T オプションでデータグラムの有効期間（TTL: Tme-To-Live）を拡張できます。詳細については、ping(1M) マン・ページを参照してください。

ttcp によるネットワーク・スループットの測定

ttcp ツールは、ネットワーク・スループットを測定します。ttcp は、伝送経路のローカル終端とリモート終端の両方でパフォーマンスを測定できるので、そのネットワーク・パフォーマンスの測定は現実的です。すべてのネットワーク管理ツールと同様、統計情報は、ネットワークの構成とアプリケーションの実行状況を念頭に置いて解釈する必要があります。たとえば、ttcp による統計は、ルータで中継された 2 台のステーション間のスループットは、その 2 台のステー

ションが同じネットワーク上にある場合のスループットよりも低くなります。同様に、大規模なデータ構造体を伝送するアプリケーションを使用しているユーザのスループットの方が、小規模なデータ構造体を伝送するアプリケーションを使用しているユーザのスループットより低くなります。

ttcp を使用するには、以下のコマンドをまず受信側から実行します。

```
ttcp -r -s -l 32768
```

次に、送信側から以下のコマンドを実行します。

```
ttcp -t -s -l 32768 -n XXX host
```

送信側の前に受信側を起動する必要があります。**-l** オプションは、書込み／読込みサイズの長さを 32K 単位で設定します。ページ・サイズの倍数であるかぎり、これを 64K または 128K に変更することで、パフォーマンスを多少向上できます。`ttcp` は、エンド・システム間でメモリツーメモリの TCP データ伝送を行い、エンドツーエンドの TCP スループットを測定します。

10baseT 半二重ネットワークでは 800KB ／秒以上、10baseT 全二重ネットワークでは 1000KB ／秒の処理能力が求められます。100baseT ネットワークの場合は、半二重に対しては 8500KB ／秒、全二重に対しては 11500KB ／秒の処理能力が求められます。

半二重と全二重の速度の違いは、その大部分がイーサネットの取得効果によるものです。`netstat -i` 入力または出力エラーはありません。一部のネットワーク・デバイス・ドライバは、`ifconfig xxx debug` モードをサポートしています。このモードでは、エラーが発生すると、より詳細な診断エラー・メッセージがコンソールに表示されます。このモードを再び無効にするには、`ifconfig xxx -debug` を使用します。

以下の例は、セキュリティの確立されたネットワーク上にある、*sheridan* と *longstreet* という 2 台のステーション（ワークステーション）間で、簡単な `ttcp` テストを実行して得られる統計情報を示しています。`ttcp` オプションの詳細については、ttcp(1) マン・ページを参照してください。

sheridan 上で以下のコマンドを入力します。

```
ttcp -r -s
```

以下の情報が表示されます。

```
ttcp-r: buflen=8192, nbuf=2048, align=16384/0, port=5001 tcp
ttcp-r: socket ttcp-r: accept from 192.102.108.4
ttcp-r: 16777216 bytes in 19.99 real seconds = 819.64 KB/sec +++
ttcp-r: 10288 I/O calls, msec/call = 1.99, calls/sec = 514.67
ttcp-r: 0.1user 3.4sys 0:19real 17%
```

longstreet 上で以下のコマンドを入力します。

```
ttcp -t -s sheridan
```

以下の情報が表示されます。

```
ttcp-t: buflen=8192, nbuf=2048, align=16384/0, port=5001 tcp  -> sheridan
ttcp-t: socket
ttcp-t: connect
ttcp-t: 16777216 bytes in 19.98 real seconds = 820.02 KB/sec +++
ttcp-t: 2048 I/O calls, msec/call = 9.99, calls/sec = 102.50
ttcp-t: 0.0user 2.3sys 0:19real 12%
```

スループットに関する統計情報は太字で強調表示され、単位は KB/ 秒です。*sheridan* ステーションのスループットは 819.64KB ／秒で、*longstreet* ステーションのスループットは 820.02 KB ／秒です。両方の値とも、ステーション間のネットワーク・パフォーマンスが良好であることを示しています。

netstat によるネットワーク統計情報の収集

`netstat` ツールは、アクティブなソケット、ルーティング、およびトラフィックを識別することによってネットワークのステータスを表示します。また、選択したオプションに基づいて、インタフェースのネットワーク・アドレスと、最大伝送単位 (MTU: Maximum Transmission Unit) を表示できます。あるシステムにおいて、出力衝突率が一貫してほかのシステムの 5% を超える場合、不良なタップやトランシーバ、端子の緩みなどの物理的な障害の可能性があります。ネットワークワイドの衝突率については、後で説明します。

衝突とネットワークの集中度を計測する上で、`netstat` を使用して分析を行う方法については、Stern 著の『Managing NFS and NIS』を参照してください。以下の例は、`netstat` によってステーション上に表示される統計情報です。また、`netstat` の各種オプションの詳細については、netstat(1) マン・ページを参照してください。

```
netstat -i
Name  Mtu    Network  Address      Ipkts   Ierrs    Opkts  Oerrs   Coll
enp0  1500   central  thumper     498690     937  1066135      3   4858
lo0  32880  loopback localhost   1678915       0  1678915      0      0
```

衝突率は約 0.45% で、これは許容範囲です。

ネットワーク・インタフェースに IP エリアスが割当てられている場合、`netstat -i` の出力は以下のようになります。

```
netstat -i
Name   Mtu   Network    Address   Ipkts    Ierrs    Opkts  Oerrs   Coll
enp0  1500   central    thumper  498690      937  1066135      3   4858
enp0  1500    bldg-2     hopper  584280      163   137245      0  40717
lo0  32880  loopback  localhost 1678915        0  1678915      0      0
```

ネットワークワイドの衝突率が一貫して 10% を超える場合、ディスクレス・ワークステーション上でのメモリの不足などさまざまな原因が考えられますが、ネットワークの分割を検討してください。109 ページの「ネットワーク・デーモンのトラブルシューティングによるネットワーク・パフォーマンスの改善」を参照してください。スループットは、ネットワーク・パフォーマンスを知る上で信頼性の高い目安です。101 ページの「ttcp によるネットワーク・スループットの測定」を参照してください。

ネットワークのチューニング情報

通常、IRIX TCP/IP サービスは、一般のネットワーク・オペレーティング環境で最適なパフォーマンスを実現するように工場出荷時にチューニングされているので、ネットワークを設定した後でチューニングする必要はありません。カーネル TCP/IP チューニング・パラメータは、`systune` コマンドまたは `/var/sysgen/master.d/bsd` ファイルで制御します。

注意：製品のマニュアルで特に指示されていないかぎり、またはネットワーク管理の経験が豊富でないかぎり、このファイル内の工場出荷時の設定値を変更することはお薦めしません。

ここでは、ネットワークのチューニングに関する以下の内容について説明します。

MTU サイズの設定

IRIX 6.2 以降のバージョンでは、MTU Discovery が実装されています。この機能によってホストは、一般的に使用される MTU サイズのテーブルを使用して、最大伝送単位（MTU: Maximum Transmission Unit）サイズを計算できます。MTU Discovery は、`/var/sysgen/master.d/bsd` の中の *tcp_mtudisc* 変数によって指定されます。*tcp_mtudisc* がオンのときに、ルータが MTU サイズ情報を指定せずにパケットの断片化を要求すると、*tcp_mssdflt* で指定されるデフォルトの最大セグメント・サイズではなく、計算されたセグメント・サイズを使用してパケットの断片化が行われます。MTU Discovery フラグはデフォルトでオン（1）に設定されています。

パケット転送の設定

デフォルトでは、複数のネットワーク・インタフェースを備えている IRIX ホストはルータと見なされ、システムの起動時に IP パケット転送がオンになります。また、工場出荷時の構成によって ICMP リダイレクトもオンになります。これによってルータは、特定のホストを宛先とするメッセージを、別の代替ルータに転送できます。*ipforwarding* と *icmp_dropredirects* という 2 つのチューニング・カーネル・パラメータが、パケット転送と ICMP リダイレクトのオンとオフを切替えます。

ファイアウォール・システムの場合は、パケットがセキュリティの範囲を侵害することがないように、*ipforwarding* と *icmp_dropredirects* をオフにする必要があります。ファイアウォール・アプリケーションのマニュアルを参照し、IP 転送機能の設定に関して特別な指示がないかどうかを確認します。詳細については、systune(1M) マン・ページも参照してください。

ウィンドウ・サイズの設定

送受信のウィンドウ・サイズは、`/var/sysgen/master.d/bsd` の中の *tcp_sendspace* と *tcp_recvspace* という変数で指定されます。デフォルトでは、両変数とも 60KB に設定されています。ワイド・エリア・ネットワーク・トラフィックの場合、ウィンドウ・サイズを 4 ～ 8KB の範囲に縮小すると、対話式のセッションの中で応答時間が向上します。4 ～ 8KB を超える TCP ウィンドウを使用しているシステムで大容量の転送トラフィックが大量に発生すると、ワイド・エリア・ネットワーク上の低速ルータや中速ルータのバッファがオーバーフローするおそれがあります。WAN ルータ・バッファがオーバーフローすると大容量転送の速度が低下し、対話式のトラフィックは通常、経路待ち行列の中でこのような大量のバーストの後に待機しなければなりません。

メモ：高速 WAN リンク（T3 など）上で小さなウィンドウ・サイズを使用すると、TCP 速度が大幅に低下します。

ローカルなイーサネット・ネットワークの場合、ウィンドウ・サイズを縮小するとパフォーマンスが向上することがあります。しかし、FDDI ネットワークの場合は、パフォーマンスが低下します。

HTTP に関する検討事項

場合によっては、`/var/sysgen/master.d/bsd` の中の変数を、ハイパーテキスト伝送プロトコル（HTTP: HyperText Transport Protocol）サーバに合わせて調整する必要があります。これらの変数は一般に、接続数と接続タイムアウトに対する変更に影響します。特定のチューニング手順については、使用している HTTP アプリケーションのマニュアルを参照してください。

低いネットワーク・パフォーマンスの改善

多くのネットワークはきわめて複雑であるために、ネットワーク・パフォーマンスが低い場合にこれを改善するのはしばしば困難な作業です。ネットワーク・パフォーマンスに影響する要因としては、ハードウェア障害、大きくなりすぎたネットワーク構成、特定のネットワーク・アプリケーションの多用、過大なパケット・サイズが考えられます。通常、`/usr/adm/SYSLOG` から診断メッセージを調べるのが適当です。以下の節では、ネットワーク・パフォーマンスを改善する方法についてそれぞれ説明します。

ハードウェア障害のトラブルシューティングによるネットワーク・パフォーマンスの改善

ハードウェアに障害があると、ネットワークの処理速度が低下したり操作不能に陥ったりします。通常、これらの障害はパケットの損失やデータの破壊といった形で現れます。パケットの損失やデータの破壊が発生すると、ネットワーク・トラフィックが増大し、管理不要な状態に陥る危険性があります。ネットワークをまずいくつかの部分に分けて、その部分ごとに診断を開始します。物理レベルで確認する主な項目は、以下のとおりです。

コントローラ・ボード

ネットワーク・メディアの帯域幅はネットワーク・トラフィックの負荷を処理するのに十分であっても、各ステーションがトラフィックを処理できない場合があります。これは、ネットワーク・インタフェース上のトラフィック量がはっきりした理由もなく増大することで分かります。このようなトラフィック量の増大は、`gr_osview` ツールで確認できます。ネットワーク・トラフィックの統計情報を表示するオプションについては、gr_osview(1) マン・ページを参照してください。ネットワーク・インタフェース上でのトラフィックが異常に多い場合は、コントローラに障害があるか、またはコントローラの処理速度が低すぎてトラフィックを処理できない可能性があります。この場合は、Efast カードなどの高速コントローラが必要です。

トランスミッタとコントローラ

トランスミッタとコントローラの両方で、SQE（Signal Quality Error、ハートビートとも呼ばれる）がオフになっていることを確認します。SQE は、ローカル・ステーションとトランシーバ間で不要なネットワーク・トラフィックを引起こす場合があります。SQE をオフに設定する方法については、ネットワー

ク・コントローラやトランシーバのインストール・マニュアルを参照してください。デフォルトでは、Silicon Graphics のすべてのネットワーク・コントローラ・ボードは、SQE がオフの状態で出荷されています。

メディアの物理的障害

ケーブル、タップ、その他のハードウェアは、ときどき壊れることがあります。イーサネット・ケーブルに関する問題を解決するには、TDR（Time Domain Reflectometer）が重要な役割を果たします。ネットワークの物理的な問題を特定するには、適正なアナライザを使用することも強くお薦めします。Silicon Graphics の NetVisualyzer は、メディアの物理的障害を検出する理想的なビジュアル・ネットワーク・アナライザです。ディスク・アクセスの障害も、ネットワーク・パフォーマンスの低下をもたらします。通常のネットワークは、ディスク・アクセスを求める NFS 要求が迅速に処理されないと、処理速度が遅いと感じます。

ネットワーク構成のトラブルシューティングによるネットワーク・パフォーマンスの改善

ネットワークの構成やトポロジが、ネットワーク・パフォーマンスに悪い影響を与える場合もあります。ネットワークの動作状態が良好のときに、これをベンチマークとしてネットワークの利用状況をモニタします。これが、変化を測定する上での基準となります。チェックする内容は、以下のとおりです。

- 物理パス — EMI と crosstalk の回線（該当する場合）。
- 連結ポイント — ハブ、および容量と接続の集信装置。
- リピータとブリッジの個数と位置 — その年数と性能。
- ルータとゲートウェイの構成 — トラフィック負荷とスループットを考慮して、特に大規模なサブネットを接続する場合は、ネットワークへの出入口を少なくとも 2 か所は確保しておく必要があります。
- ワーク・グループの加入とリソース（NFS ファイルシステム、NIS ドメイン、電子メール、財務データベース）の日常的な共有。
- クライアント／サーバ構成と、これが作成するトラフィック。
- 各ワークステーションの利用状況、特定のディスク・アクティビティ、ディレクトリ名検索の負荷量。

ネットワーク構成はきわめて複雑なので、91 ページの「ネットワーク管理についての参考文献」に示したすべての参考文献を理解していることが必要です。

ネットワーク・デーモンのトラブルシューティングによるネットワーク・パフォーマンスの改善

rwhod や rtnetd など、ネットワーク・デーモンの中には、ネットワークやネットワーク・インタフェースに好ましくない影響を与えるものがあります。たとえば、ワークステーションがマルチプロセッサの場合や、ワークステーション上でリアルタイム・プロセスが動作している場合は、ステーション上で rtnetd が実行されることがあります。このデーモンは、リアルタイム・プロセスに迅速な応答を提供するために、到着するネットワーク・パケットを先取りする役割を果たします。これは、ユーザがネットワーク処理とリアルタイム処理のトレードオフを認識している場合は、特に問題ありません。したがって、ネットワーク・デーモンは個別に評価します。

ルータや、ネットワークが集中するステーション（メール・サーバ、NIS サーバ、DNS サーバなど）では、rtnetd ソフトウェアをロードしないでください。

パケット・サイズの縮小によるネットワーク・パフォーマンスの改善

イーサネットでは、データの最大伝送単位（MTU）は 1500 バイトです。ネットワークのパフォーマンスと効率性は、メディアの MTU までは、パケット・サイズに伴って向上します。メディアの MTU よりも大きいパケットは、メディアの MTU の範囲内で小さいパケットに分割（断片化）する必要があります。IRIX 6.2 以降のバージョンでは MTU Discovery を使用して断片化を実行します。詳細については、105 ページの「MTU サイズの設定」を参照してください。

カーネル構成によるネットワーク・パフォーマンスの改善

いくつかのパラメータを変更することによって、ネットワークの動作をローカル構成に合わせてカスタマイズできます。以下に示すパラメータは、/var/sysgen/master.d/bsd 設定ファイルで定義されています。このファイルを変更した後にカーネルを再構成する方法については、『IRIX Admin: System Configuration and Operation』を参照してください。

カーネル・チューニング・オプションの変更によるネットワーク・パフォーマンスの改善

カーネルには *tcp_sendspace*、*tcp_recvspace*、*udp_sendspace*、および *udp_recvgrams* という 4 つのパラメータがあり、これらはネットワーク・パフォーマンスに直接影響します。

これらのパラメータは、TCP（SOCK_STREAM）ソケットと UDP（SOCK_DGRAM）ソケットによって使用されるデフォルトのバッファ領域を決定します。*tcp_sendspace* パラメータと *tcp_recvspace* パラメータは、ソケットに割当てられる初期バッファ領域を定義します。*udp_sendspace* パラメータは、送信できる UDP データグラムのデフォルトの最大サイズを決定します。*udp_recvgrams* パラメータは、UDP ソケットの受信用バッファに格納できる最大サイズの UDP データグラムの個数を決定します。各 UDP ソケットの受信用バッファの総バイト数は、`udp_sendspace` と `udp_recvgrams` の積になります。プログラムで `setsockopt` システム・コールのオプションである SO_SNDBUF と SO_RCVBUF を使用すると、ソケットの送信用バッファと受信用バッファのサイズを増減できます。旧式の TCP の実装では、TCP *sendspace/recvspace* 値が大きいと、多くの問題があります。旧式のステーションで通信に関する問題が発生する場合は、値を 60 から 24 に減らしてください。

4.2BSD との互換性を保つために、IRIX システムでは TCP シーケンス番号の初期値が正の値になるように制限しています。

TCP パラメータの変更による PC との接続性の改善

TCP/IP を実装した業界標準のパーソナル・コンピュータを Silicon Graphics のワークステーションやサーバに接続すると、正常に動作しない場合があります。これは、IRIX の `/var/sysgen/master.d/bsd` ファイルで設定されている *tcp_sendspace* 変数と *tcp_recvspace* 変数のサイズが大きいことが原因です。

パーソナル・コンピュータを正常に接続するためには、上記のパラメータの値をデフォルトの（60 * 1024）から（24 * 1024）に変更し、lboot コマンドでカーネルを再構成します。これらの値を再構成する方法については、『IRIX Admin: System Configuration and Operation』を参照してください。

SLIP と PPP

この章では、シリアル回線インターネット・プロトコル（SLIP: Serial Line Internet Protocol）と Point-to-Point プロトコル（PPP）の Silicon Graphics バージョンについて説明します。SLIP と PPP は、シリアル回線を介した TCP/IP 用のプロトコルであり、リモート・システムと LAN 間の接続、2 つのネットワーク間の接続に使用します。

この章では、以下について説明します。

SLIP と PPP について

Silicon Graphics の SLIP および PPP では、RFC 1144 データ圧縮法と、Silicon Graphics 独自のデータ圧縮法（ヘッダ・フレミング、チェックサム、TCP/IP 情報を 3 バイトに圧縮する方法）の両方を提供しています。SLIP は *eoe* ソフトウェア・サブシステムの一部であり、*eoe.sw.slip* パッケージとしてインストールされます。システムに SLIP がインストールされているかどうかを確認するには、`versions` コマンドを使用します。

PPP は、ほぼ SLIP とほぼ同じ機能ですが、より柔軟で強力な機能を備えています。そのため通常、SLIP よりも PPP が使用されます。ただし、接続するシステムが PPP ではなく SLIP をサポートしている場合には、SLIP を使用する必要があります。電話回線やモデムを介したダイアルアップ接続で PPP を使用する以外に、ISDN 接続で PPP を使用することもできます。ただしこのガイドでは、通常のシリアル回線接続で PPP を使用することを前提に説明します。

メモ：ISDN を使用してリモート・ホストに接続している場合には、この章で説明する手順のかわりに、『ISDN User's Guide』を参照することをお薦めします。isdn(7m) マン・ページを参照してください。ただし、この章を読むことにより、PPP の操作およびメンテナンスに関して理解することができます。

PPP は *eoe* ソフトウェア・サブシステムの一部であり、*eoe.sw.ppp* パッケージからインストールされます。システムに PPP がインストールされているかどうかを確認するには、`versions` コマンドを使用します。

通常、SLIP または PPP は次のような目的に使用されます。

- 1 つのリモート・システムをネットワークに接続するために使用します。クライアント／サーバ構成のシステムの場合、一方のシステムがクライアントになり、直接ネットワークに接続しているシステムがサーバになります。サーバがすべての経路制御を処理します。
- 2 つのネットワークを接続するために使います。この場合、2 つのシステムは通常は対等な関係になります。

SLIP リンクまたは PPP リンクは、一方のシステムがもう一方のシステムにダイヤルすることにより、常に一方のシステムが接続の動作を開始するように設定したり、どちらのシステムからも接続の動作を開始できるように設定することができます。クライアント／サーバ接続の場合は、通常はクライアントがサーバにダイヤルして接続の動作を開始します。この場合も、両方向から

の接続が可能です。2 つのネットワークを接続した場合も、一方のシステムから接続の動作を開始するように設定するか、または必要に応じてどちらのシステムからも接続の動作を開始するように設定できます。

SLIP または PPP 接続の設定：一般的な手順

各システムを SLIP クライアントまたは PPP クライアントとして設定するには、次の操作を行います。

1. SLIP および PPP ソフトウェアがインストールされていることを確認します。ソフトウェアがインストールされているかどうかを調べる場合には、114 ページの「SLIP および PPP ソフトウェアの確認」を参照してください。
2. 必要に応じてソフトウェアをインストールします。ソフトウェアをインストールする場合には、115 ページの「SLIP および PPP ソフトウェアのインストール」を参照してください。
3. モデムまたはケーブルなど、適切なハードウェアを選択し両方のシステムにインストールします。詳しい説明については 115 ページの「モデムの選択」を参照してください。
4. IP アドレスを選択します。IP アドレスの割当てについては 116 ページの「SLIP および PPP クライアントの IP アドレスの割当て」を参照してください。
5. 発信用にソフトウェアを設定します。116 ページの「発信用にシステムを設定」 を参照してください。
6. ルート動作を設定します。ルート割当てについては 125 ページの「SLIP および PPP の経路制御とアドレス割当て」を参照してください。

システムを SLIP サーバまたは PPP サーバとして設定するには、次の操作を行います。

1. SLIP および PPP ソフトウェアがインストールされていることを確認します。114 ページの「SLIP および PPP ソフトウェアの確認」を参照してください。
2. 必要に応じてソフトウェアをインストールします。115 ページの「SLIP および PPP ソフトウェアのインストール」を参照してください。
3. モデムまたはケーブルなど、適切なハードウェアを選択しインストールします。詳しい説明については 115 ページの「モデムの選択」を参照してください。

4. 着信用にソフトウェアを設定します。各クライアントに対して構成情報を設定します。122 ページの「着信用にシステムを設定」を参照してください。
5. ルート動作を設定します。経路制御については 125 ページの「SLIP および PPP の経路制御とアドレス割当て」を参照してください。

SLIP または PPP を使用して 2 つのネットワークを接続するには、次の操作を行います。

1. SLIP および PPP ソフトウェアがインストールされていることを確認します。114 ページの「SLIP および PPP ソフトウェアの確認」を参照してください。
2. 必要に応じて両方のシステムにソフトウェアをインストールします。115 ページの「SLIP および PPP ソフトウェアのインストール」を参照してください。
3. モデムまたはケーブルなど、適切なハードウェアを選択しインストールします。115 ページの「モデムの選択」を参照してください。
4. 一方または両方のシステムで発信用にソフトウェアを設定します。116 ページの「発信用にシステムを設定」を参照してください。
5. 一方または両方のシステムで着信用にソフトウェアを設定します。122 ページの「着信用にシステムを設定」を参照してください。
6. ルート動作を設定します。経路制御については 125 ページの「SLIP および PPP の経路制御とアドレス割当て」を参照してください。

SLIP および PPP ソフトウェアの確認

SLIP と PPP を使用するには、適切なソフトウェアがインストールされていなければなりません。SLIP を使用するには、*eoe.sw.slip* サブシステムと *eoe.sw.uucp* サブシステムをインストールする必要があります。PPP を使用するには、*eoe.sw.ppp* サブシステムと *eoe.sw.uucp* サブシステムをインストールする必要があります。インストールしたサブシステムのマン・ページ（*eoe.man* サブシステム）もインストールします。これらのパッケージがインストールされているかどうかを確認するには、`versions` コマンドを使用します。たとえば、PPP がインストールされているかどうかを確認するには、次のコマンドを実行します。

```
% versions eoe.\*.{ppp,uucp}
```

PPP と UUCP がインストールされている場合には、`versions` コマンドを実行すると次のように出力されます。

```
I = Installed, R = Removed

Name                    Date      Description

I  eoe                 09/24/96 IRIX Execution Environment, 6.3
I  eoe.man             09/24/96 IRIX Execution Environment Man Pages
I  eoe.man.ppp         09/24/96 Point-to-Point Protocol Man Pages
I  eoe.man.uucp        09/24/96 UNIX-to-UNIX Copy Man Pages
I  eoe.sw              09/24/96 IRIX Execution Environment Software
I  eoe.sw.ppp          09/24/96 Point-to-Point Protocol Software
I  eoe.sw.uucp         09/24/96 UUCP Utilities
```

SLIP および PPP ソフトウェアのインストール

ソフトウェアのインストール方法については、『IRIX Admin: Software Installation and Licensing』を参照してください。SLIP ソフトウェアまたは PPP ソフトウェアをインストールした後は、システムを再起動してカーネルを構築してください。

モデムの選択

SLIP または PPP で使用するモデムは、伝送速度として最低 9,600 ビット／秒（bps：ボー）が要求されます。ただしこの速度ではリンクがかなり低速になるため、V.32 bis または V.34 standard に準拠した、14,400 bps 以上の伝送速度をサポートするモデムを使用してください。半二重モデムも使用できますが、対話型のタスクにはあまり向いていません。

Silicon Graphics ではこのタイプのモデムを製造していません。このため、このモデムの変更に関しては保証していません。また、SLIP または PPP で使用する電話回線の性能やモデムの互換性についても保証していませんが、独立したデータグレードの電話回線を推奨しています。

高速モデムを使用する場合は、RTS/CTS によるハードウェア・フロー制御がサポートされていることを確認してください。高速モデムには、デバイス名として `ttyf*` を使用します。

メモ：Silicon Graphics は、DSI™、Intel®、Telebit®、ZyXEL™、U.S. Robotics®、Hayes® および大半の Hayes 互換モデムをサポートしています。ほかのモデムも使用できますが、設定が複雑であり、その動作についても保証されません。

モデムのインストールおよび設定方法については、『IRIX Admin: Peripheral Devices』の第 1 章「モデムのインストール」を参照してください。

SLIP および PPP クライアントの IP アドレスの割当て

SLIP クライアントまたは PPP クライアントでは、SLIP または PPP インタフェースに対する IP アドレスとホスト名が必要です。ローカル・エリア・ネットワークにも接続している場合は、ほかのネットワークとのインタフェースのアドレスとホスト名も必要になります。システム管理者またはネットワーク・サービス・プロバイダによりアドレスとホスト名が割当てられている場合は、その IP アドレスとホスト名を使ってシステムを設定します。

一部のネットワーク・サービス・プロバイダでは、クライアントがネットワークに接続したときに、動的に IP アドレスを割当てています。IRIX PPP ではこの動的アドレス割当てがサポートされていますが、IRIX SLIP ではサポートされていません。動的アドレス割当てについては、129 ページの「PPP の動的アドレス割当ての使い方」を参照してください。

LAN の SLIP サーバまたは PPP サーバを設定する場合や、シリアル・リンクで 2 つのネットワークを接続する場合は、125 ページの「SLIP および PPP の経路制御とアドレス割当て」を参照してください。

発信用にシステムを設定

発信用に SLIP または PPP を設定するには、次の操作を行います。

1. `/etc/uucp/Systems` ファイルに接続を記述した行を追加します。
2. `/etc/uucp/Devices` ファイルにモデムを記述した行を追加します。
3. `/etc/uucp/Dialers` ファイルにモデムがリストされていない場合は、そのモデムのエントリを記述した行を追加します。

4. モデム行が /etc/inittab ファイルで正しく設定されているかどうか確認します。

PPP を設定する場合は、これらの操作に加え、/etc/ppp.conf ファイルを設定します。

この節では、発信用に SLIP または PPP を設定する方法を説明します。

- 「発信用のファイル設定」（117 ページ）
- 「発信用 SLIP の設定例」（120 ページ）
- 「発信用 PPP の設定例」（121 ページ）

発信用のファイル設定

ここでは、次に示す SLIP と PPP の設定ファイルについて説明します。

- /etc/uucp/Systems については、117 ページの「発信用の /etc/uucp/Systems の設定」を参照してください。
- /etc/uucp/Devices については、118 ページの「発信用の /etc/uucp/Devices の設定」を参照してください。
- /etc/uucp/Dialers については、119 ページの「発信用の /etc/uucp/Dialers の設定」を参照してください。
- /etc/uucp/inittab については、119 ページの「発信用の /etc/inittab の設定」を参照してください。
- /etc/ppp.conf については、120 ページの「発信用の /etc/ppp.conf の設定」を参照してください。

発信用の /etc/uucp/Systems の設定

/etc/uucp/Systems ファイルには、リモート・ステーションを呼出すための情報が記述されています。このファイルには、ローカル・ステーションのモデムの速度とパスワード、リモート・ステーションのノード名と電話番号が記述されています。これらの情報を使用してリモート・ステーションにログインします。SLIP または PPP を使用した接続は、次のように記述します。

system Any *type speed phone login-script*

system にはリモート・システム名を指定します。*type* には、`Devices` ファイルの回線とモデムのタイプを示すエントリ名を指定します。*speed* にはシステムとモデムの間の接続速度を指定します。*phone* にはリモート・システムの電話番号を指定します。*login-script* は、SLIP または PPP にリモート・システムへのログイン方法を指定します。

次の例は、ローカル・ステーションの wenders が、`/etc/uucp/Systems` ファイルのこの記述を使用して、ステーション lynch を呼出しています。この接続は 38,400 bps で行われ、パスワードは hopper です。SLIP は、ログイン・プロンプトに対しては slip-wenders、パスワード・プロンプトに対しては hopper と応答してリモート・ステーションにログインします。

```
lynch Any ACUSLIP 38400 5551212 "" \r\c ogin:--ogin: slip-wenders \
asswd: hopper SLIP
```

この情報は、1 行で入力します。最後の文字列 `SLIP` は、リモート・ステーションから SLIP プロトコルを起動するという通知を受けるまでローカル・ステーションを待機させます。

Systems ファイルの 3 番目のフィールド（この例では、`ACUSLIP`）は、リモート・ステーションを呼出すモデム行を指定します。このフィールドには、`/etc/uucp/Devices` ファイルのエントリと一致するものが少なくとも 1 つはなければなりません。

`/etc/uucp/Systems` ファイルに関する詳細は、202 ページの「UUCP Systems ファイル」を参照してください。

発信用の /etc/uucp/Devices の設定

IRIS ステーションで SLIP を使用するためにデバイス、モデム速度、ダイヤル・プログラムを `/etc/uucp/Devices` に設定します。`/etc/uucp/Devices` には、次の形式で SLIP 行を記述します。

type device `null` *speed* `212 x` *dialer*

最初のフィールドにある *type* には任意の文字列を指定できますが、Systems ファイルで指定されているタイプと一致している必要があります。使用できるモデムが複数ある場合は、すべてに同じ *type* 名を指定します。呼出しを行うときに、システムが自動的にいずれかのモデムを選択します。SLIP または PPP で使用するモデムを指定するには、通常は *type* 名を ACUSLIP にします。

device は、現在使用されていないポートのフロー制御デバイスになります。高速モデムの場合、ハードウェア・フロー制御およびサウンド・ケーブルを推奨します。*speed* は、システムとモデ

ム間の接続速度になり、*dialer* は `/etc/uucp/Dialers` にリストされてる任意のダイアル・プログラムになります。

複数のモデム速度を設定したい場合、複数のポートを使用したい場合、または異なるコマンドをサポートしているモデムを使用したい場合には、新しい行を追加してそれらの内容を定義します。

例えば、次の行では、Telebit™ T2500 モデムを 38,400 bps で、シリアル・ポート 2 のハードウェア・フロー制御デバイスで使用するために SLIP を設定しています。

```
ACUSLIP ttyf2 null 38400 212 x t25slip
```

`/etc/uucp/Devices` ファイルに関する詳しい情報は、194 ページの「UUCP Devices ファイル」を参照してください。

発信用の /etc/uucp/Dialers の設定

`/etc/uucp/Dialers` には、各種のモデムを指定します。`Devices` ファイルの *dialer* フィールドには、これらのモデムのいずれかと同じものが指定されている必要があります。

IRIX でサポートされているモデムをインストールする場合は、dialers ファイルにエントリを追加する必要はありません。詳細については、199 ページの「UUCP Dialers ファイル」を参照してください。

発信用の /etc/inittab の設定

`/etc/uucp/Devices` で指定したポートは、`/etc/inittab` ファイルで発信用または発信 / 着信用に設定します。『IRIX Admin: Peripheral Devices』の第 1 章「モデムのインストール」の説明に従ってモデムを発信用または発信 / 着信用に設定する場合は、`/etc/inittab` ファイルが正しく設定されている必要があります。

たとえば、`ttyf2` を発信用の SLIP として使用する場合は、`/etc/inittab` の `ttyf2` 行を次のように記述します。

```
t2:23:off:/etc/getty ttyf2 co_38400         # port 2
```

この例では、ポート 2 で `getty` プログラムをオフにしています。

どちらのステーションからも SLIP リンクを起動できるようにするには、uugetty をオンにします。たとえば、Telebit T2500 モデムを使用して対称リンクを設定するには、前述の行を次のように変更します。

```
t2:23:respawn:/usr/lib/uucp/uugetty -Nt 60 -it25in,conn ttyf2
dx_38400
```

/etc/inittab ファイルに加えた変更は、次回 init が /etc/inittab ファイルを読込んだときに有効になります。変更直後、/etc/inittab の内容を有効にするには、次のコマンドを実行します。

```
/etc/telinit q
```

詳細については、inittab(4) マン・ページを参照してください。

発信用の /etc/ppp.conf の設定

/etc/ppp.conf ファイルは、PPP 接続のオプションを指定します。このファイルの各エントリは、ホスト名とオプションから構成されます。次に示すのは、このファイルのエントリの例です。

```
salad     out remotehost=dial-in.salad.com
          localhost=caesar.salad.com
          quiet add_route
```

この例では、ローカル・ホスト caesar.salad.com とリモート・ホスト dial-in.salad.com 間の接続を指定します。**out** キーワードは、外部への接続を示します。**quiet** キーワードは、デマンド・ダイヤルによる接続を指定します。デマンド・ダイヤルについては、130 ページの「デマンド・ダイヤルについて」を参照してください。**add-route** キーワードは、dial-in.salad.com を介して PPP にデフォルトのルート経路を設定させます。

ppp.conf の詳細については、ppp(1M) マン・ページを参照してください。

発信用 SLIP の設定例

ここでは、SLIP の発信用の設定例を示します。この例では、tuna.salad.com と dial-in.salad.com を接続します。つまり、salad.com 社内ネットワークにサーバをセットアップします。

/etc/uucp/Systems

```
salad Any ACUSLIP 38400 5551212 "" \r\c ogin:--ogin: slip-tuna \
asswd: celery SLIP
```

```
/etc/uucp/Devices
```

```
ACUSLIP ttyf2 null 38400 212 x t25slip
```

```
/etc/inittab
```

```
t2:23:off:/etc/getty ttyf2 co_38400         # port 2
```

この例の設定を使用して SLIP を起動するには、次のコマンドを実行します。

```
% /usr/etc/slip -o -p comp -r salad
```

発信用 PPP の設定例

PPP の設定ファイルは `/etc/ppp.conf` です。このファイルの詳細については、ppp(1M) マン・ページを参照してください。次に、`ppp.conf` ファイルの例を示します。

```
salad     out remotehost=dial-in.salad.com
          localhost=caesar.salad.com
          quiet add_route
```

このエントリでは、spice.com ネットワークに対するゲートウェイとして機能するスタンドアロン・システムから、リモート・ホスト dial-in.salad.com への外部接続を指定しています。このエントリでは、デマンド・ダイヤル（**quiet**）モードが指定されています。通常、スタンドアロン・クライアントには、PPP サーバを介してデフォルトの経路を設定するように **add_route** キーワードを指定します。エントリ *salad* は、次のように `/etc/uucp/Systems` ファイルのエントリと一致している必要があります。

```
salad Any ACUSLIP 38400 555-1212 "" @\r\c ogin--ogin: ppp-caesar \
ssword: mypasswd PPP
```

IRIX PPP は、プロトコルを起動する前に、`starting PPP` メッセージを送信します。このため、上記のチャット・スクリプトではログインが成功したことを示す PPP 文字列が返されるまで待ちます。IRIX 以外のシステムに接続している場合は、この例から PPP 文字列を削除します。

`/etc/uucp/Devices` ファイルには、ACUSLIP のエントリが少なくとも 1 つなければなりません。

```
ACUSLIP ttyf2 null 38400 212 x t25slip
```

このエントリでは、ttyf2 を Telebit T2500 モデムに接続することを指定しています。

/etc/inittab ファイル内では、モデム・ポートの速度（この例では 38,400 bps）を設定し、getty または uugetty をオフにします。/etc/inittab ファイルの詳しい編集方法と telinit の起動方法については、『IRIX Admin: Peripheral Devices』を参照してください。上記で示した PPP のエントリには、次のエントリを使用できます。

```
t2:23:off:/etc/uucp/uugetty ttyd2 dx_38400  # ppp modem
```

上記のエントリでは発信用 PPP が使用できますが、着信用 PPP については、回線に応答するように uugetty を設定する必要があります。詳細については、『IRIX Admin: Peripheral Devices』を参照してください。

上記の設定は、特定のサイトにおける一例です。サイトの構成やモデムのメーカーやモデルが異なれば、ここで示した例のように正しく動作しないことがあります。たとえば、ダイヤル先の PPP サイトでは /etc/ppp.conf ファイルの設定が異なっている場合や、Devices ファイル内のエントリでモデムのメーカーやモデルが指定されている場合には、正しく動作しません。また、この例では『IRIX Admin: Peripheral Devices』の第 1 章「モデムのインストール」の説明に従ってモデムが設定されていることが前提となっています。

ファイルを設定した後、*root* で ppp コマンドを実行します。上記の例では、次のコマンドを実行します。

```
ppp -r salad
```

ppp コマンドとそのオプションについては、ppp(1M) マン・ページを参照してください。

着信用にシステムを設定

システムを着信用に設定するには、少なくとも 1 つのポートを着信用（または、着信／発信用）に設定します。着信用のシステムごとに /etc/passwd ファイルにエントリが必要です。また、SLIP 接続の場合は /usr/etc/remoteslip ファイルに、PPP 接続の場合は /etc/ppp.conf ファイルにそれぞれエントリが必要です。

ここでは、着信用に SLIP および PPP を設定する方法について説明します。

SLIP による着信のための /etc/passwd の設定

SLIP では、ログインするには `/etc/passwd` ファイルにエントリが必要です。ユーザ ID とグループ ID は、どちらもゼロにします。`/etc/passwd` のエントリの最後にシェルが指定されるのに対し、SLIP では `/usr/etc/remoteslip` ファイルが指定されます。tuna.salad.com を slip-tuna としてログインするには、dial-in.salad.com の `/etc/passwd` に次の行が必要です。

```
slip-tuna:3RsB768WRAN2.:0:0:slip for tuna:/:/usr/etc/remoteslip
```

SLIP で呼出されるステーションごとに、このようなエントリが必要です。セキュリティ上の理由により、各 SLIP ユーザのホーム・ディレクトリの書込み許可は、特権ユーザのみが所有するようにしてください。`/tmp` などのような公用ディレクトリでは、システムのセキュリティを保証できません。

メモ：この例の暗号化されたパスワードは、実際のパスワードを表しているわけではありません。SLIP ログインのパスワードは、`passwd` コマンドで設定します。`passwd` の使い方については、passwd(1) マン・ページを参照してください。

SLIP による着信のための /usr/etc/remoteslip の設定

`/etc/passwd` の slip-tuna エントリでは、ログイン・シェルが `/usr/etc/remoteslip` ファイルとして指定されています。このファイルを使用してリモート・ステーション上の SLIP を呼出します。`/usr/etc/remoteslip` では、*slip* コマンドでリモート・ステーション名や接続に必要なその他のオプションを指定します。

`/usr/etc/remoteslip` は、Bourne シェルのスクリプト・ファイルです。ほかの Bourne シェルのスクリプトと同様に、case 文を追加することができます。オンラインの sh(1) マン・ページでは、シェルのスクリプト・プログラムを詳しく説明しています。SLIP 接続ごとに、次の形式でエントリを記述します。

```
slip-nodename )
    exec /usr/etc/slip options
    ;;
```

nodename はリモート・ステーションの名前です。`slip` のオプションの詳細については、slip(1M) マン・ページを参照してください。

dial-in.salad.com ステーションでは、`/usr/etc/remoteslip` ファイルに次のエントリがあります。

```
# Edit the case statement as required.

case $USER in

       slip-tuna)
              exec /usr/etc/slip -p comp -i -r tuna
              ;;
       *)
              exec /usr/etc/slip -i -r $USER
              ;;
esac
```

この例では、**`-p comp`** オプションが指定されているので、dial-in.salad と tuna の接続では Silicon Graphics 独自のヘッダ予測／圧縮オプションを使用して高速にデータを転送しています。RFC 1144 圧縮オプションを使用する場合は、**`-p cslip`** を代わりに指定します。このオプションは、相手方のステーションからセッションが入力されることを SLIP に知らせます。`slip` のオプション **`-r tuna`** は、リモート・ステーション名を指定します。また、この例ではデフォルトの case が記述されていることに注意してください。これにより、すべての SLIP クライアントで同じパラメータを使用している場合は、クライアントごとにエントリを追加する必要がなく、デフォルトの case を変更するだけです。

PPP による着信のための /etc/passwd の設定

SLIP と同様、PPP でログインするには `/etc/passwd` ファイルにエントリが必要です。ユーザ ID とグループ ID はどちらもゼロ（0）にします。`/etc/passwd` の通常のエントリでは最後にシェルが指定されるのに対し、PPP では `/usr/etc/ppp` コマンドが指定されます。caesar.salad.com システムを ppp-caesar としてログインするには、dial-in.salad.com の `/etc/passwd` に次の行が必要です。

```
ppp-caesar:3RsB768WRAN2.:0:0:PPP for caesar:/:/usr/etc/ppp
```

PPP で呼出されるステーションごとにこのようなエントリが必要です。セキュリティ上の理由により、PPP アカウントのホーム・ディレクトリの書込み許可は、特権ユーザのみが所有するようにしてください。`/tmp` などの公開されているディレクトリでは、システムのセキュリティを保証できません。

メモ：この例の暗号化されたパスワードでは、実際のパスワードを表しているわけではありません。PPP ログインのパスワードは、`passwd` コマンドで設定します。`passwd` の使い方については、passwd(1) マン・ページを参照してください。

デフォルトのパラメータを使用する場合は、クライアントに対する `/etc/ppp.conf` エントリは不要です。次のように、最低限のエントリを追加してください。

```
ppp-caesar     in remotehost=caesar.salad.com
```

メモ：サーバ・システムでは、**`add-route`** キーワードを使用しないでください。

SLIP および PPP の経路制御とアドレス割当て

SLIP リンクまたは PPP リンクを介して経路制御を行うには、次の 3 つの方法があります。これらは、状況に応じて使分けます。

- メインのネットワークで、SLIP を使用してサーバに接続しているスタンドアロン・クライアントが少数の場合、サーバのネットワークからクライアント・アドレスを割当て、proxy ARP 経路制御（PPP は、自動的に ARP テーブル・エントリを処理します）を使用します。proxy ARP 経路制御に関する詳しい説明は、126 ページの「SLIP 接続のための Proxy ARP 経路制御」を参照してください。
- サーバに接続しているスタンドアロンのクライアントが多数の場合は、SLIP および PPP クライアント・アドレスのためにサブネットを確保します。この場合、サーバはメイン・ネットワークに対して SLIP/PPP ネットワークのゲートウェイになります。これに関しては、128 ページの「クライアント・アドレスのための SLIP/PPP サブネットの設定」で説明されています。

- SLIP または PPP を使用して 2 つのネットワークを接続している場合は、各ネットワークには固有のネットワーク番号が必要であり、リンク先であるシステムでは両方とも `routed` を実行します。詳しい情報は、129 ページの「SLIP または PPP によるネットワーク接続」を参照してください。

いずれの場合でも、小規模なネットワークで静的な経路制御を使用している場合を除き、サーバで経路制御デーモン `routed` を実行する必要があります。詳細については、routed(1M) および route(1M) マン・ページを参照してください。`routed` は、`chkconfig` コマンドを使用してオンにします。

```
# chkconfig routed on
```

スタンドアロン・クライアントでは、`routed` を実行しないでください。

SLIP 接続のための Proxy ARP 経路制御

SLIP サーバに接続しているスタンドアロン・ホストが少数の場合には、Proxy ARP 経路制御を使用することができます。Proxy ARP 経路制御は、PPP 接続に必須ではありませんが、PPP は必要に応じて自動的に ARP テーブル・エントリをインストールします。このシステムでは、サーバのネットワーク側から各スタンドアロン・ホストにインターネット・アドレスを割当てます。各クライアントはサーバを介してデフォルトの経路制御が設定され、サーバはアドレス解決プロトコル（ARP: Address Resolution Protocol）を使用して各クライアントのアドレスを監視することができます。

SLIP 接続のための Proxy ARP 経路制御の設定

サーバを介してデフォルトの経路制御を設定するには、SLIP 接続を構築した後に `route` コマンドを使用します。SLIP を起動するスクリプトにこのコマンドを追加してください。

`route add net default` *server-address* `1`

各 SLIP クライアントに対して、サーバは `arp` コマンドを実行します。

`arp -s` *client-hostname server--ethernet-address* `pub`

サーバのイーサネット・アドレスは IP アドレスと同じではありません。サーバのイーサネット・アドレスを確認するには、同じイーサネット上の別のシステムから `arp` コマンドを実行します。

```
% arp dial-in.salad.com
dial-in.salad.com (192.70.79.7) at 8:0:69:9:4f:ef
```

Proxy ARP 経路制御の例

コロンで区切られた 16 進数の文字列（8:0:69:9:4f:ef）が、サーバのイーサネット・アドレスです。

ローカル・ネットワークのスクリプトにコマンドを追加することにより、ブート時にサーバが arp コマンドを実行するよう設定することができます。次に、複数のクライアントに対して ARP エントリを設定するローカル・ネットワークのスクリプトの例を示します。

```
#!/bin/sh
#
# starting up local networking stuff
#

IS_ON=/etc/chkconfig
CONF=/etc/config
SERV_ADDR=8:0:69:9:4f:ef

if $IS_ON verbose ; then
        ECHO=echo
        VERBOSE=-v
else            # For a quiet startup and shutdown
        ECHO=:
        VERBOSE=
fi

case "$1" in
  'start')
      # setup proxy ARP for the dialin hosts
      # if this host has more than one interface,
      # you will need to hard-code the Ethernet MAC address
      # instead of letting it be determined at run time.
      # note that 4DDN (among others) may change the MAC address
from default!
      # and some AppleTalk packages change the output of `netstat -ian`!
      arp -s client1 $SERV_ADDR pub
      arp -s client2 $SERV_ADDR pub
      arp -s client3 $SERV_ADDR pub
      ;;
  'stop')
      # be nice and delete the ARP entries
      arp -d client1
```

```
        arp -d client2
        arp -d client3
        ;;
  *)
          echo "usage: $0 {start|stop}"
          ;;
esac
exit 0
#
```

上記のファイル名が /etc/init.d/network.local と仮定し、ネットワークの起動直後にこのファイルを実行する場合には、次のコマンドで起動時とシャットダウン時のリンクを設定します。

```
ln -s /etc/init.d/network.local /etc/rc2.d/S31netlocal
ln -s /etc/init.d/network.local /etc/rc0.d/K39netlocal
```

クライアント・アドレスのための SLIP/PPP サブネットの設定

スタンドアロン・クライアントが多数の場合は、そのすべてのクライアントに対して arp コマンドを実行するのは面倒です。このような場合は、SLIP および PPP クライアント用に確保しておいた特別なサブネットからクライアント・アドレスを割当てます。サーバがこのネットワークへのゲートウェイとなります。サーバで routed を実行し、**-F** オプションを指定することにより不要なネットワーク・トラフィックを減らしてください。たとえば、SLIP/PPP サブネットが 128.70.80 の場合、-F 128.70.80 という文字列を /etc/config/routed.options ファイルに追加します。

クライアントは、proxy ARP 経路制御と同じように、サーバを介してデフォルトの経路を設定します。また、クライアント間で通信を行う場合は、そのイーサネット・インタフェースをオフにする必要があります。

```
% chkconfig network off
```

クライアントでイーサネット・インタフェースが有効になっている場合は、サーバではなくイーサネットを介して同じネットワーク上の別のクライアントにアクセスします。

SLIP または PPP によるネットワーク接続

経路を制御する上で、SLIP または PPP を使用して 2 つのネットワークを接続するのが最も簡単な方法です。各ネットワークには固有のネットワーク番号があり、サーバとクライアントのアドレスはそれぞれのネットワークから割当てられます。クライアントとサーバの両方で `routed` を実行してください。

メイン・ネットワークのルータは、新しいネットワークを認識してそのネットワークに対して経路制御を行うように設定します。

PPP の動的アドレス割当ての使い方

動的アドレス割当てが設定された PPP を使用するサービス・プロバイダに接続している場合は、`/etc/ppp.conf` ファイルに **`localhost=0,0`** と **`add_route`** の 2 つのキーワードを指定することにより、リモート・システムで IP アドレスの割当てが可能になり、リモート・システムを介してデフォルトの経路を設定できます。

双方向リンクの設定

単純な SLIP または PPP リンクでは、一方のステーションが接続を開始します。この設定は変更することができ、いずれのステーションからも接続を開始するように設定できます。

まず、一方向のリンクを設定します。そのリンクが正しく動作したら、逆方向のリンクを設定します。発信と着信の両方に同じ `ppp.conf` エントリを使用します。

ブート時に SLIP および PPP を自動起動する方法

ネットワーク間の SLIP 接続または PPP 接続を自動的に開始するには、`/etc/init.d/network.local` というローカル・ネットワーク・スクリプトを作成します。このスクリプトはリンクを開始するステーションに置き、`/etc/rc2.d` ディレクトリと `/etc/rc0.d` ディレクトリの適切なファイルにリンクします。たとえば、121 ページの「発信用 PPP の設定例」で説明したデマンド・ダイヤル PPP リンクを自動的に起動するには、caesar.salad.com にローカ

ル・ネットワーク・スクリプトを作成します。このスクリプトでは、引数 **start** が指定されて呼出されたときに PPP を起動し、引数 **stop** が指定されて呼出されたときに PPP を終了します。

```
#!/bin/sh
# ppp boot startup script
case $1 in
   start)
       /etc/killall ppp
       if /etc/chkconfig ppp && test -x /usr/etc/ppp -a -s /etc/ppp.conf
       then
           /usr/etc/ppp -r salad &
       fi
       ;;
   stop)
       /etc/killall -TERM ppp
       ;;
   *)
       echo “usage: $0 {start|stop}”
       ;;
esac
```

このスクリプトは、`/etc/rc2.d` ディレクトリと `/etc/rc0.d` ディレクトリの適切なファイル名にリンクします。

```
ln -s /etc/init.d/network.local /etc/rc2.d/S31netlocal
ln -s /etc/init.d/network.local /etc/rc0.d/K39netlocal
```

デマンド・ダイヤルについて

不定期ではあるが頻繁に SLIP と PPP リンクを使用する場合は、デマンド・ダイヤルでリンクした方が経済的です。デマンド・ダイヤルでは、デフォルトで送信するネットワーク・トラフィックがある場合に電話回線を接続し、送信するトラフィックがない場合は接続を切ります。デフォルト設定では、ネットワーク・トラフィックが必要でない場合、ネットワークには接続しません。たとえば、タイム・デーモン（`timed`）の生成するトラフィックでは呼出しを行いませんが、ファイル転送が要求されるとネットワークに接続します。

デマンド・ダイヤルの設定

デマンド・ダイヤルを使用するには、接続を開始するシステムで **-q** オプションを指定して `slip` コマンドを実行します。デマンド・ダイヤル・モードは、quiet モードとも呼ばれます。`slip` コマンドとそのオプションについては、slip(1M) マン・ページを参照してください。

`ppp.conf` ファイルにキーワード quiet を追加すれば、PPP にデマンド・ダイヤルが設定されます。

デマンド・ダイヤルを使用している場合は、ブート時に SLIP および PPP が自動的に呼出されるように設定できます。129 ページの「ブート時に SLIP および PPP を自動起動する方法」を参照してください。

リンクが開始されるのは、接続の動作を開始した方のシステムにトラフィックがある場合だけです。

SLIP および PPP を介した NFS

NFS は SLIP リンクまたは PPP リンクで実行できます。NFS トランザクションで送信するデータ量は多く、処理速度が遅くなります。しかし、次の方法によって処理速度を改善することができます。

- 9600 bps 以上のモデムを使用します。
- SLIP ヘッダ予測／圧縮オプションを使用します。

  SLIP の圧縮オプション **comp** と **cslip** については、slip(1M) マン・ページを参照してください。
- NFS オプションの **rsize**、**wsize**、**timeo**、**retrans** を再設定してから、NFS ファイル・システムをマウントします。

  パフォーマンスを改善するには、読込みと書込みをするブロックを小さくし、タイムアウト値を長く設定します。NFS ファイル・システムの詳細については、fstab(4) マン・ページを参照してください。

SLIP および PPP を介したファイル転送

ほかのデマンド・ユーティリティがリンクを共有している場合にシリアル回線を介したファイル転送を行うと、遅くなる可能性があります。ファイル転送速度を速くするには、uucp を使用してください。uucp は、ほかのユーティリティと回線を共有したくない場合には効果的です。

SLIP および PPP リンクのトラブルシューティング

SLIP リンクまたは PPP リンクが接続しているのに、リモート・ネットワークのシステムにアクセスできない場合は、経路制御に問題がある可能性があります。ping コマンドでリモート・システムにアクセスしてみます。

```
% ping -c 10 dial-in.salad.com
```

正常に接続している場合は、次のように出力されます。

```
PING dial-in.salad.com (128.70.79.52): 56 data bytes
64 bytes from 128.70.79.52: icmp_seq=0 ttl=255 time=2 ms
64 bytes from 128.70.79.52: icmp_seq=1 ttl=255 time=1 ms
64 bytes from 128.70.79.52: icmp_seq=2 ttl=255 time=1 ms
64 bytes from 128.70.79.52: icmp_seq=3 ttl=255 time=1 ms
64 bytes from 128.70.79.52: icmp_seq=4 ttl=255 time=1 ms
64 bytes from 128.70.79.52: icmp_seq=5 ttl=255 time=1 ms
64 bytes from 128.70.79.52: icmp_seq=6 ttl=255 time=1 ms
64 bytes from 128.70.79.52: icmp_seq=7 ttl=255 time=1 ms
64 bytes from 128.70.79.52: icmp_seq=8 ttl=255 time=2 ms
64 bytes from 128.70.79.52: icmp_seq=9 ttl=255 time=1 ms


----dial-in.salad.com PING Statistics----
10 packets transmitted, 10 packets received, 0% packet loss
```

接続に問題がある場合は、次のように出力されます。

```
PING dial-in.salad.com (128.70.79.52): 56 data bytes


----dial-in.salad.com PING Statistics----
10 packets transmitted, 0 packets received, 100% packet loss
```

リモート・ホストにアクセスできるにもかかわらず、ネットワーク上のほかのシステムにアクセスできない場合は、経路制御に問題があります。125 ページの「SLIP および PPP の経路制御とアドレス割当て」で説明したとおりに経路制御が設定されているかどうか確認します。

どのシステムにも接続できない場合は、別のユーティリティで通信回線を調べます。uucp を使い慣れている場合は、uucp でステーション間の接続を確認します。それ以外は、cu で接続を確認します。

メモ：cu を使用するには、/etc/uucp/Devices ファイルに *direct* エントリが必要です。詳しい説明は、194 ページの「UUCP Devices ファイル」を参照してください。

SLIP リンクをデバッグする場合は、各ステーションを別々に調べます。まず、cu コマンドで次のようにローカル・ステーションのポートとモデムを調べます。

```
cu -d -s speed -l port
```

たとえば、ステーション tuna.salad.com にインストールされているポートとモデムをテストするには、次のように cu コマンドを実行します。

```
cu -d -s 38400 -l ttyf2
```

このコマンド実行する前に、まず /etc/inittab ファイルにある port 行の **respawn** に **off** を指定し、(uu)getty をオフにします。

これで、モデムが応答するはずです。大部分のモデムは、AT を出力して応答します。モデムが応答しない場合は、ローカル・ステーションの SLIP の設定手順を見直し、マニュアルを参照してモデムの設定を確認します。モデムが応答した場合は、次のようにチルダ記号とドットを入力して cu を切断します。

```
~.
```

次に、SLIP でリンクするのと同じように相手のステーションと接続します。ローカル・ステーション上で cu を実行し、cu ですでに検証したポートを介してリモート・ステーションを呼出します。

cu コマンドを次のように実行し、リモート・ステーションとの接続を調べます。

```
cu -d -sspeed telno
```

たとえば、tuna と dial-in.salad.com の接続をテストするには、cu コマンドで tuna から特定の番号を呼出してみます。

```
cu -d -s38400 5552002
```

これで、ローカル・ステーションがモデムに指示を与えてリモート・ステーションを呼出すかどうか確認します。呼出しが正しく行われると、ログイン・プロンプトが表示されます。expect 文字列に対して、/etc/uucp/Systems ファイルの *send* 文字列を入力します。

BIND ネーム・サーバ

Berkeley インターネット・ネーム・ドメイン（BIND: Berkeley Internet Name Domain）サーバは、IRIX オペレーティング・システム環境でインターネット・ドメイン・ネーム・サービス（DNS: Domain Name Service）を実現するソフトウェアです。ネーム・サーバは、クライアントがネットワーク上の資源およびオブジェクトの名前を指定し、これらの情報をほかのネットワーク・オブジェクトを共有できるようにします。つまり、ネーム・サーバは、コンピュータ・ネットワーク上のオブジェクトに対する分散データベース・システムになります。すべての IRIX ネットワーク・プログラムは、BIND を利用して、ステーション名やアドレスを格納したり検索することができます。また、BIND は、従来の `/etc/hosts` ファイルを用いたホスト・テーブルの検索の代わりに使用することができます。

BIND は、2 つのソフトウェアから構成されています。1 つは、ネーム・サーバ・プログラムの `named` であり、もう 1 つは、サーバにアクセスする C ライブラリのリゾルバ・ルーチンです。BIND をネーム・サーバとして設定するには、`named` をインストールする必要があります。詳細については、141 ページの「BIND 設定ファイル」を参照してください。`named` は、バックグラウンドで動作するデーモンであり、UDP と TCP による問い合わせに応答します。リゾルバ・ルーチンは、標準 C ライブラリの `libc.a` に含まれています。ホスト・アドレスの検索ルーチン `gethostbyname`、`gethostbyaddr`、および `sethostent` はリゾルバ・ルーチンを使ってネーム・サーバに問い合わせます。*resolver* に記述されているリゾルバ・ライブラリ・ルーチンは、問い合わせパケットを構築し、それらをネーム・サーバとの間で交換します。

この章では、以下について説明します。

ドメイン・ネーム・サービス

`/etc/hosts` ファイルなどを使用するホスト・テーブル検索ルーチンでは、ネットワーク全体に対するマスター・ファイルを集中管理する必要があります。この方法は、ステーション数が少なく、ステーションを管理する各グループが互いに協力できる小規模なネットワークには向いていますが、ステーションが組織の外側とも通信できるような大規模なネットワークには向いていません。

DNS を使用すると、すべての名前を 1 か所で集中管理する必要がなくなります。名前情報の管理は、ネットワーク上の組織に委任できます。

DNS は、IRIX ファイルシステム同様、ドメインを階層構造で構成しています。図 6-1 に階層の一部を示します。階層内のサブツリーはそれぞれドメインと呼ばれ、ラベルが付いています。階層の最上位はルート・ドメインであり、空のラベルが付いています。ドメイン名は、ルートから現在のドメインまでのすべてのドメイン・ラベルを連結したものです。ラベルは右から左に連結され、ドットで区切られます。図 6-1 で fruit というラベルが付いているドメインの名前は、fruit.salad.com になります。同じドメインの中では、ラベルは一意である必要があります。

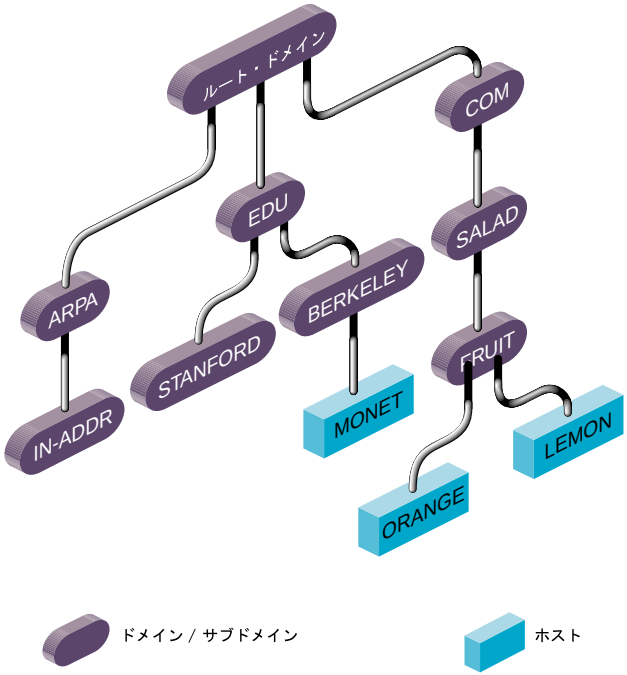

図 6-1　　ドメインの階層

ルート・ドメインのすぐ下がトップレベル・ドメインです。これらのトップレベル・ドメインは比較的静的なドメインであり、ネットワーク・インフォメーション・センター（NIC: Network Information Center）が管理しています。NIC に現在登録されているトップレベル・ドメインは、次のとおりです。

arpa	ステーションの一時ドメイン。アドレスと名前の対応付けでトップレベルのドメインとしても使われます。
com	企業、商業組織
edu	大学やその他教育機関
gov	政府機関
mil	軍関係機関
net	各種のネットワーク型組織、ネットワーク管理関連組織（インフォメーション・センターやオペレーション・センターなど）

org　　技術サポート組織、学会、その他の団体

このほかに、日本を表す JP、ドイツの DE、フランスの FR など、国を表す多数のドメインがあります。

ドメインの名前空間は、ゾーンと呼ばれる領域に分割され、各ゾーンが重複することはありません。各ゾーンの情報は、そのゾーンのマスター・ネーム・サーバによって処理されます。各ゾーンは、特定のドメインからリーフ・ドメインまで、またはほかのゾーンの始まるドメインまでがその範囲です。通常、ゾーンは管理上の境界を表します。

たとえば、カリフォルニア大学バークレー校にあるステーションのドメイン名は次のようになります。

monet.berkeley.edu

この例では、edu が教育機関のトップレベル・ドメインであり、berkeley は edu のサブドメイン、monet はステーション名を表しています。

BIND サーバとクライアント

BIND はサーバ／クライアント・モデルをベースとしています。サーバには、その権限の度合いによっていくつかのクラスに分類されます。ここでは、各種のサーバ間およびサーバとクライアント間の関係について説明します。表 6-1 に BIND のサーバについてまとめます。

表 6-1 は、各種 BIND のサーバ構成の特徴をまとめています。

表 6-1 BIND サーバ構成

プライマリ・サーバ	セカンダリ・サーバ	キャッシュ専用サーバ	フォワード・サーバ	スレーブ・サーバ
ドメインの管理サーバ	プライマリ・サーバから権限を委任されたサーバ	非管理サーバ	非管理サーバ	非管理サーバ
ローカル・ファイルからデータをロードします。	プライマリ・サーバからデータをロードします。	問い合わせに応えるか、問い合わせを管理サーバに転送します。	再帰的要求に応えるか、ほかのネーム・サーバと通信してから要求に応えます。	指定したサーバ（フォワード）リストからデータにアクセスします。

すべてのサーバ構成で、`named` サーバ・デーモンを起動する必要があります。設定フラグ **`named`** をオンに設定すると、ステーションの起動時に `named` デーモンが自動的に起動します。詳細については、chkconfig(1M) マン・ページを参照してください。

クライアントは、`resolv.conf` ファイルに指定されているネーム・サーバのデータにアクセスします。クライアントがドメイン・サーバ `named` を起動することはありません。

メモ：IRIX 6.5 オペレーティング・システム環境下で起動すると、統一ネーム・サービス（UNS: Unified Name Service）がデーモン `nsd` を実行し、`/etc/nsswitch.conf` ファイルを使って BIND クライアントの解決順序を制御します。詳細については、171 ページの「BIND に対する UNS の動作」を参照してください。

BIND マスター・サーバ

ドメインのマスター・サーバは、そのドメインに対する権限を持っています。このマスター・サーバは、そのドメインに対応するすべてのデータを管理します。各ドメインには、最低 2 つのマスター・サーバが必要です。つまり、1 つはプライマリ・サーバ、もう 1 つはセカンダリ・サーバです。セカンダリ・サーバは、プライマリ・サーバを利用できない場合や負荷がかかりすぎた場合にバックアップ・サービスを行います。1 台のサーバを、複数のドメインのマスター・サーバ

として設定できます。あるドメインのプライマリ・サーバとして機能しながら、別のドメインのセカンダリ・サーバとして機能することもできます。

プライマリ・サーバは、ディスク上のファイルからデータをロードするサーバです。ドメイン内のほかのサーバに権限を委任することもできます。セカンダリ・サーバは権限を委任されたサーバであり、プライマリ・サーバから特定のドメインに関するデータを受取ります。起動時、セカンダリ・サーバはドメインに関するすべてのデータを提供するようにプライマリ・サーバに要求します。以降、セカンダリ・サーバは定期的にプライマリ・サーバを調べ、データを更新する必要があるかどうか確認します。

ルート・サーバは、ルート・ドメインのマスター・サーバであり、インターネット・ドメインのトップ・レベルになります。ルート・サーバは *root.cache* ファイルに登録されています。root.cache の詳細については、146 ページの「BIND の root.cache ファイル」を参照してください。

BIND スレーブ・サーバとフォワード・サーバ

スレーブ・サーバは、ローカルに解決できない問い合わせをルート・ドメインやほかのドメインのマスター・ネーム・サーバには問い合わせずに、常に特定のリスト内にあるフォワード・サーバに転送します。フォワード・サーバは複数ある場合もあり、リストの終わりに達するまで順番に試みられます。

スレーブ・サーバとフォワード・サーバの組合わせは、インターネットに直接アクセスできない環境を構築する場合に有用です。管理上、ステーションの中にはインターネットにアクセスできないように設定されているものもあります。そのようなステーションがインターネットのドメイン・システムにアクセスできるようにするには、これらのステーションをゲートウェイ・ステーション上のフォワード・サーバに対するスレーブ・サーバに設定します。ゲートウェイのフォワード・サーバは、インターネット上のほかのネーム・サーバと通信することによって、問い合わせを解決して応答します。フォワード・サーバを用いることの長所は、より多くの情報をキャッシュとして保持できることです。スレーブ・モードおよびフォワード・サーバの使用については、148 ページの「BIND 環境の構築」を参照してください。

フォワード・サーバを使用する理由は 2 つあります。使用しているステーションに完全なネットワーク・アクセス権がない場合に使用します。IP パケットをネットワーク上のほかのステーションに転送できないため、ネットワークにアクセスできるフォワード・サーバを利用する必要があります。もう 1 つは、フォワード・サーバはすべての問い合わせがサーバを通過するときにそれ

らを認識できるので、通常のステーション・ネーム・サーバのキャッシュよりも完全なデータ・キャッシュを蓄えることができるからです。実際には、フォワード・サーバはステーションが使用できるメタキャッシュとなるので、そのサイトからネットワーク上のそれ以外のステーションへの問い合わせの総数を減らすことができます。

BIND キャッシュ専用サーバ

キャッシュ専用サーバは、どのドメインに対する権限も持っていません。このサーバは、キャッシュで解決できなかった問い合わせを権限のあるほかのサーバに問い合わせます。キャッシュ専用サーバは問い合わせの回答を得るとそれをキャッシュするので、権限のあるサーバへのトラフィックが減少します。問い合わせの回答には、有効期間（*TTL: time-to-live*）フィールドが含まれており、これがキャッシュを保持する期間を示しています。

BIND クライアント

BIND クライアントは、ネットワーク上のほかのステーションで実行中のネーム・サーバにアクセスします。`named` サーバは、クライアント・ステーション上では実行されません。

BIND 設定ファイル

ここでは、各種 BIND 設定ファイルについて説明します。このようなファイルの例については、148 ページの「BIND 環境の構築」を参照してください。

IRIX では、`named` データベース・ファイルは `/var/named` ディレクトリに保存されており、`named` はデフォルトではインストールされません。このため、インストール・メディアから `eoe.sw.named` を選択する必要があります。`eoe.sw.named` がインストールされたことを確認するには、次を実行します。

```
versions eoe.sw.named
```

README ファイルには、設定手順をまとめたものと、BIND クライアントと BIND サーバのリストが記述されています。通常、サーバはクライアントにもなります。BIND クライアントには `/etc/resolv.conf` ファイルが必要です。

/var/named/Examples サブディレクトリには、named データベース・ファイルのサンプルがあります。この *Example* ディレクトリ内のファイルをユーザ独自の設定条件に合わせて変更します。これらのファイルは、付録 A「BIND 標準リソース・レコードの形式」で説明するレコード形式になっています。次に示すのは、BIND 環境の設定に必要なデータベース・ファイルです。

- named.boot
- root.cache
- named.hosts
- named.rev
- localhost.rev

メモ：ネットワークに複数のドメインがある場合は、named.hosts、named.rev、localhosts.rev の各ファイル名にドメイン名を組込み、ユーザ独自のファイルを作成します。

データベース・ファイルの数と構成は、サーバのタイプによって異なります。

表 6-2 にサーバのタイプごとに必要なデータベース・ファイルをまとめます。

表 6-2 named データベース・ファイル

ファイル名	プライマリ・サーバ	セカンダリ・サーバ	キャッシュ専用サーバ	フォワード・サーバ	スレーブ・サーバ
named.boot	必須	必須	必須	必須	必須
localhosts.rev	必須	必須	必須	必須	必須
named.hosts	必須	不要	不要	不要	不要
named.rev	必須	不要	不要	不要	不要
root.cache	必須	必須	必須	必須	必須

BIND のブート・ファイル

BIND のブート・ファイルは、named 起動時に最初にに読込まれ、サーバのタイプは何か、どのゾーンの権限を持つか、どこで初期データを得るかなどをサーバに指示します。ブート・ファイ

ルのデフォルト名は、/etc/named.boot です。ブート・ファイルのテンプレートは、/var/named/Examples/named.boot.master（プライマリ・サーバ用）と named.boot.slave（セカンダリ・サーバ用）です。

デフォルトのブート・ファイルとは異なるファイルを使用するには、/etc/config/named.options ファイルを作成または変更して次のエントリを挿入します。

-b *other-bootfile-name*

次に、ブート・ファイルの構造について説明します。

BIND ブート・ファイルのディレクトリ指定

directory 行には、ネーム・サーバを実行するディレクトリを指定します。ブート・ファイルにあるほかのファイル名は、このディレクトリからの相対パスになります。

```
directory /var/named
```

このエントリは必須です。named は、$INCLUDE 文の相対パスで指定できます。また、named のコア・ダンプもこのディレクトリに生成されます。

BIND ブート・ファイルのプライマリ・マスター指定

ブート・ファイルでは、プライマリ・サーバのゾーンが次のように指定されています。

```
primary   Berkeley.EDU   named.hosts
```

第 1 フィールドは、このサーバが第 2 フィールドで指定したゾーンのプライマリ・サーバであることを示します。第 3 フィールドは、データを読込むファイルの名前です。

BIND ブート・ファイルのセカンダリ・サーバ指定

セカンダリ・サーバを指定する行は、データを受取るサーバがプライマリ・サーバでない点を除けば、プライマリ・サーバを指定する行と同じです。たとえば、次のように指定します。

```
secondary  Berkeley.EDU 128.32.0.10 128.32.0.4 ucbhosts.bak
```

第 1 フィールドは、このサーバが第 2 フィールドで指定したゾーンのセカンダリ・サーバであることを示します。第 3 フィールド、第 4 フィールドのネットワーク・アドレスは、ゾーンのプラ

イマリ・サーバのアドレスです。セカンダリ・サーバは、これらのプライマリ・サーバからネットワークを介してデータを受取ります。データの受取りが成功するまで、ここに挙げた順番でデータの受取りを試みます。

第 3 フィールド、第 4 フィールド（プライマリ・サーバのリスト）の後ろにファイル名フィールドがあれば、データがそのバックアップ・ファイルに保存されます。サーバを起動する場合は、可能であればこのバックアップ・ファイルからデータをロードし、そのデータが最新のものかどうかをプライマリ・サーバに問い合わせます。

BIND ブート・ファイルのキャッシュ専用サーバ指定

すべてのサーバのブート・ファイルには、ネーム・サーバのキャッシュを準備するために次のような行を記述する必要があります。

```
cache    .    root.cache
```

ここに記述されたすべてのキャッシュ・ファイルが `named` の起動時に読込まれます。有効な値がキャッシュにリストアされ、初回の問い合わせに対しては、キャッシュ・ファイルのルート・ネーム・サーバ情報が常に使われます。

ネーム・サーバは、ネットワークのルート・ドメインに対して権限のあるネーム・サーバを認識する必要があります。`root.cache` ファイルは、そのような高い権限を持つネーム・サーバのアドレスでキャッシュを初期化します。このファイルは、付録 A「BIND 標準リソース・レコードの形式」で説明する標準リソース・レコード形式（マスター・ファイル形式）になっています。

サーバがキャッシュ・サーバであることを指定する行を特に記述する必要はありません。*secondary* や *primary* などの行がブート・ファイルになければ、キャッシュ専用サーバになります。

BIND ブート・ファイルのフォワード・サーバ指定

どのサーバでもフォワード・サーバを利用できます。たとえば、再帰的問い合わせを処理できるサーバは、ほかのステーションに代わって問い合わせを解決することができます。フォワード・サーバを設定するには、次のように `forwarders` コマンドでインターネット・アドレスを指定します。

```
forwarders    128.32.0.10     128.32.0.4
```

BIND ブート・ファイルのスレーブ・モード指定

ネットワーク・アクセスが制限されているために、問い合わせを解決する唯一の手段がフォワード・サーバである場合にスレーブ・モードを使用します。ネーム・サーバがリストされているほかのサーバではなくフォワード・サーバを使用することを禁止する場合にも、スレーブ・モードを使用できます。スレーブ・モードをオンにする場合は、ブート・ファイルに次のコマンドを記述します。

```
slave
```

`slave` を使用する場合は、フォワード・サーバも指定する必要があります。スレーブ・モードでは、問い合わせが解決されるか、フォワード・リストの終わりに達するまで、各フォワード・サーバに順次問い合わせを転送していきます。

BIND の named.hosts ファイル

このファイルには、ドメインのホスト／アドレス・データベースが記述されています。このファイルは、プライマリ・サーバには必須です。

BIND の named.rev ファイル

このファイルは、IN-ADDR.ARPA ドメインを指定します。これは、IP アドレスを名前に変換します。インターネット・アドレスはドメインの境界を超えるので、この特殊なドメインにより逆のマッピングが可能になります。IN-ADDR.ARPA ドメインには、その前に 4 つのラベルが付きます。これらのラベルは、インターネット・アドレスの 4 つのオクテットを逆の順序で並べたものです。オクテットは、値がゼロであっても 4 つ指定する必要があります。

たとえば、インターネット・アドレス 128.32.130.12 は、ドメイン 12.130.32.128.IN-ADDR.ARPA に対応付けられます。アドレスを逆にすることによって、ネットワーク上にあるステーションの自然なグループ分けができます。

IN-ADDR.ARPA ドメインは、ネットワークを表すこともできます。たとえば、ARPANET がネットワーク 10 であれば、10.IN-ADDR.ARPA というドメインになります。

BIND の localhost.rev ファイル

このファイルは、ローカル・ループバック・インタフェースのネットワーク・アドレスである 127.0.0.1 の IN-ADDR.ARPA ドメインを指定します。このアドレスは、`localhost` アドレスとして知られています。重要なネットワーク・プログラムには、このドメイン内の情報に依存しているものが多くあります。このファイルはすべてのサーバに必須です。

BIND の root.cache ファイル

このファイルは、デフォルトでルート・ドメイン・サーバの初期キャッシュ・データを保持しています。形式を問わず、このファイルはすべてのサーバに必須です。

BIND の /etc/config/named.options ファイル

このファイルはオプションです。このファイルは、ステーションの起動時に `named.restart` スクリプトにより読込まれます。このファイルには、`named` のコマンド行で使用する引数を指定します。指定できるオプションについては、named(1M) マン・ページを参照してください。

/etc/resolv.conf によるホスト名の検索の設定

BIND クライアントに必要な設定ファイルは `/etc/resolv.conf` だけです。このファイルは、`gethostbyname` または `gethostbyaddr` 呼出し時に最初に読取られます。`resolv.conf` には次の役割があります。

- デフォルトのドメインまたはデフォルトのドメイン検索リストを定義します。
- `gethostbyname`および`gethostbyaddr`が使用するホスト検索方法の順序を指定します。
- ネーム・サーバのインターネット・アドレスをリストします。

最初の 2 つは、クライアント・ステーションとサーバの両方に対して行います。3 番目は、クライアント・ステーションだけに行います。ファイルの形式については、resolver(4) マン・ページを参照してください。

ステーションをリモート・サーバのクライアントとして設定するには、ネーム・サーバのインターネット・アドレスを示す **nameserver** エントリを /etc/resolv.conf に追加します。たとえば、次のように入力します。

```
nameserver 128.32.130.12
```

この nameserver エントリは 3 つまで指定できます。通常、ローカル・サーバを実行している場合はこのファイルを作成する必要はありません。ローカル・サーバを明示的に指定する場合は、インターネット・アドレスに 0（使用しているステーションを意味します）を指定する必要があります。

クライアントとサーバの両方において、/etc/sys_id の名前をフル・ドメイン名で指定します。たとえば、次のように入力します。

```
monet.Berkeley.EDU
```

フル・ドメイン名を使用しない場合は、キーワード domain とステーションのドメイン名を記述した行を resolv.conf ファイルに追加します。たとえば、次のように入力します。

```
domain berkeley.edu
```

通常、ライブラリ・ルーチン gethostbyname と gethostbyaddr は、次の順序でステーション情報にアクセスするように設定されています。

1. NIS
2. BIND
3. ローカルの /etc/hosts ファイル

IRIX 6.5 よりこの順番は、/etc/nsswitch.conf の *hosts* キーワードで変更することができます。詳細については、171 ページの「BIND に対する UNS の動作」を参照してください。

シングル・ユーザ・モードの場合にシステム管理者がほかのステーションからファイルをコピーできるようにするには、ローカル・ステーションのネットワーク・インタフェースと *localhost* のエントリのほかに、重要なステーションのエントリを /etc/hosts ファイルに記述します。ファイルの形式については、hosts.equiv(4) マン・ページを参照してください。

BIND 環境の構築

ここでは、BIND 環境をどのように構築するのか例を挙げて説明し、また、各種のサーバやクライアントを設定する手順についても説明します。ここで挙げる例では、インターネットに接続していることを前提としています。ユーザ独自の環境を設定する場合は、ここでの各変数をユーザ独自の BIND 環境変数に置換えます。ここでは、次の変数を使用します。

- ドメインの名前は *fruit.com*、ネットワーク・アドレスは 128.70.10 です。ネットワークはインターネットに接続されています。
- プライマリ・サーバの名前は *apples.fruit.com*、インターネット・アドレスは 128.70.10.1 です。設定方法の詳細については、149 ページの「BIND プライマリ・サーバの設定」を参照してください。
- セカンダリ・サーバは *oranges.fruit.com*、インターネット・アドレスは 128.70.10.2 です。設定方法の詳細については、153 ページの「BIND セカンダリ・サーバの設定」を参照してください。
- フォワード・サーバの名前は、*banana.fruit.com*、インターネット・アドレスは 128.70.10.3 です。設定方法の詳細については、155 ページの「BIND フォワード・サーバの設定」を参照してください。
- キャッシュ専用サーバの名前は *guava.fruit.com*、インターネット・アドレスは 128.70.10.4 です。設定方法の詳細については、154 ページの「BIND キャッシュ専用サーバの設定」を参照してください。
- スレーブ・サーバの名前は *pineapple1.fruit.com* と *pineapple2.fruit.com*、インターネット・アドレスは 128.70.10.8 と 128.70.10.9 です。設定方法の詳細については、156 ページの「BIND スレーブ・サーバの設定」を参照してください。
- クライアントの名前は *plum1.fruit.com*、*plum2.fruit.com* と *plum3.fruit.com*、インターネット・アドレスは 128.70.10.5、128.70.10.6 および 128.70.10.7 です。設定方法の詳細については、157 ページの「BIND クライアントの設定」を参照してください。

メモ：BIND の現バージョンでは、システムのコンポーネントまたはドメイン名に対して、下線（_）を使用することはできません。

図 6-2 に上記の変数を使用した BIND 環境の例を示します。表記の都合上、図ではステーションに省略名を用いています。

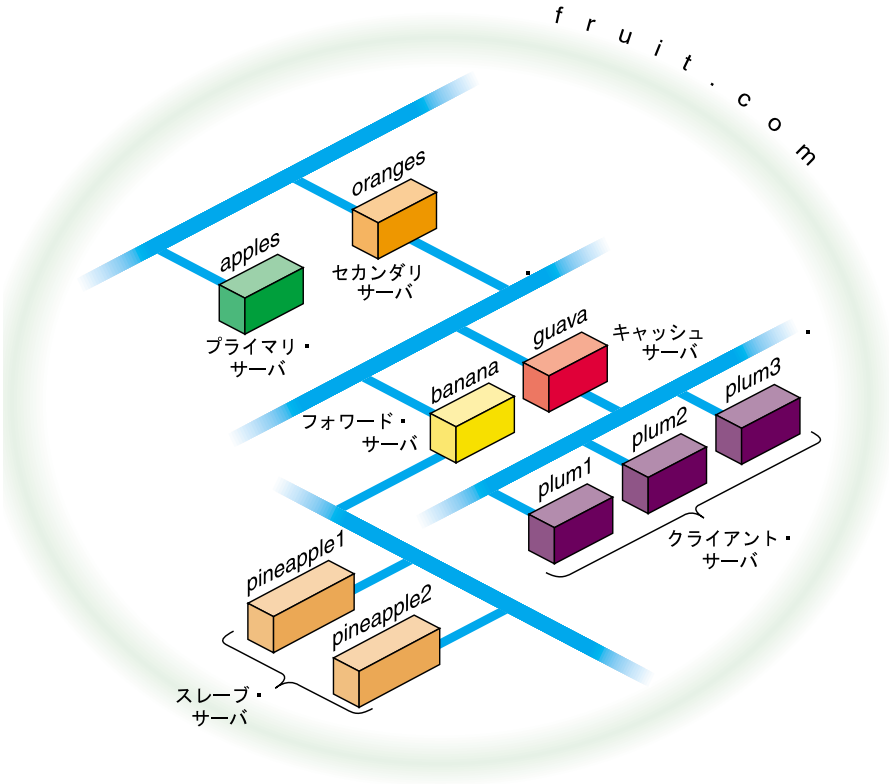

図 6-2 BIND の構築例

BIND プライマリ・サーバの設定

プライマリ・サーバを設定するには、次の手順に従います。

1. `root` でログインします。
2. named のサンプル・ディレクトリに移動します。

```
cd /var/named/Examples
```

3. テンプレート・ファイルを */var/named* ディレクトリにコピーします。

```
cp named.boot.master root.cache named.hosts \
named.rev localhost.rev /var/named
```

4. named.boot.master をデフォルトのファイル名に変更します。

```
cd ..
mv named.boot.master named.boot
```

5. エディタで named.boot を次のように変更します。

```
;
; Boot file for apple.fruit.com, primary for fruit.com
;
directory /var/named
;type    domain  source host/file  backup file
cache      .          root.cache
primary  fruit.com  fruit.named.hosts
```

6. named.hosts（ここでは *fruit.named.hosts*）ファイルを次のように変更します。

```
; Authoritative data for fruit.com
;
@   IN  SOA  apples.fruit.com. named-mgr.apples.fruit.com.
            (1994021501 ; Serial
             10800      ; Refresh 3 hours
             3600       ; Retry 1 hour
             3600000    ; Expire  1000 hours
             86400 )    ; Minimum 24 hours
; authoritative name servers for fruit.com
          IN    NS     apples.fruit.com.
          IN    NS     oranges.fruit.com.
; address records for all hosts on the net
          IN     A     128.70.10.1
apples    IN     A     128.70.10.1
oranges   IN     A     128.70.10.2
banana    IN     A     128.70.10.3
guava     IN     A     128.70.10.4
plum1     IN     A     128.70.10.5
plum2     IN     A     128.70.10.6
plum3     IN     A     128.70.10.7
pineapple1 IN     A     128.70.10.8
pineapple2 IN     A     128.70.10.9
localhost IN     A     127.0.0.1
; canonical or alias name for localhost
loghost   IN        CNAME localhost
```

7. localhost.rev ファイルを次のように変更します。

```
;localhost.rev  -- PTR record for 127.1
;
@  IN  SOA  apples.fruit.com. named-mgr.apples.fruit.com.
                            (1994021501 ;Serial
                             10800      ;Refresh 3 hours
                             3600       ;Retry   1 hour
                             3600000    ;Expire 1000 hours
                             86400 )    ;Minimum 24 hours
; authoritative name servers for fruit.com
   IN      NS      apples.fruit.com.
   IN      NS      oranges.fruit.com.
0  IN      PTR     loopback.fruit.com.
1  IN      PTR     localhost.
```

8. named.rev（ここでは fruitnamed.rev）ファイルを次のように変更します。

```
;
;       @(#)named.rev   1.1     (Berkeley)      86/02/05
;
@  IN  SOA  apples.fruit.com. named-mgr.apples.fruit.com.
                             (1994021501 ; Serial
                              10800 ; Refresh 3 hours
                              3600  ; Retry   1 hour
                              3600000 ; Expire  1000 hours
                              86400 ); Minimum 24 hours
;authoritative name servers for fruit.com
               IN      NS     apples.fruit.com.
               IN      NS     oranges.fruit.com.
;named.rev addresses, by default, are the last two numbers
;of the internet addresses in reverse order, if Class B
;address.  If Class C address, then it's the last number.
1              IN    PTR      apples.fruit.com.
2              IN    PTR      oranges.fruit.com.
3              IN    PTR      banana.fruit.com.
4              IN    PTR      guava.fruit.com.
5              IN    PTR      plum1.fruit.com.
6              IN    PTR      plum2.fruit.com.
7             IN    PTR    plum3.fruit.com.
8             IN    PTR    pineapple1.fruit.com.
9             IN    PTR    pineapple2.fruit.com.
```

9. プライマリ・サーバがインターネットに接続されている場合は、デフォルトの `root.cache` ファイルを使用します。必要に応じて、anonymousFTP で rs.internic.net から最新リストを入手します。このリストは、ディレクトリ `domain/named.root` にあります。

```
;       This file holds the information on root name servers needed to
;       initialize cache of Internet domain name servers
;       (e.g. reference this file in the “cache  .  <file>”
;       configuration file of BIND domain name servers).
;
;       This file is made available by InterNIC registration services
;       under anonymous FTP as
;           file                /domain/named.root
;           on server           FTP.RS.INTERNIC.NET
;       -OR- under Gopher at    RS.INTERNIC.NET
;           under menu          InterNIC Registration Services (NSI)
;              submenu          InterNIC Registration Archives
;           file                named.root
;
;       last update:    Sep 1, 1995
;       related version of root zone:   1995090100
;
;
; formerly NS.INTERNIC.NET
;
.                        3600000  IN  NS    A.ROOT-SERVERS.NET.
A.ROOT-SERVERS.NET.      3600000      A     198.41.0.4
;
; formerly NS1.ISI.EDU
;
.                        3600000      NS    B.ROOT-SERVERS.NET.
B.ROOT-SERVERS.NET.      3600000      A     128.9.0.107
;
; formerly C.PSI.NET
;
.                        3600000      NS    C.ROOT-SERVERS.NET.
C.ROOT-SERVERS.NET.      3600000      A     192.33.4.12
;
; formerly TERP.UMD.EDU
;
.                        3600000      NS    D.ROOT-SERVERS.NET.
D.ROOT-SERVERS.NET.      3600000      A     128.8.10.90
;
; formerly NS.NASA.GOV
;
.                        3600000      NS    E.ROOT-SERVERS.NET.
```

```
E.ROOT-SERVERS.NET.      3600000      A     192.203.230.10
;
; formerly NS.ISC.ORG
;
.                        3600000      NS    F.ROOT-SERVERS.NET.
F.ROOT-SERVERS.NET.      3600000      A     39.13.229.241
;
; formerly NS.NIC.DDN.MIL
;
.                        3600000      NS    G.ROOT-SERVERS.NET.
G.ROOT-SERVERS.NET.      3600000      A     192.112.36.4
;
; formerly AOS.ARL.ARMY.MIL
;
.                        3600000      NS    H.ROOT-SERVERS.NET.
H.ROOT-SERVERS.NET.      3600000      A     128.63.2.53
;
; formerly NIC.NORDU.NET
;
.                        3600000      NS    I.ROOT-SERVERS.NET.
I.ROOT-SERVERS.NET.      3600000      A     192.36.148.17
; End of File
```

10. 次のコマンドを実行し、named をオンに設定してステーションを再起動します。

    chkconfig named on
    reboot

BIND セカンダリ・サーバの設定

セカンダリ・サーバを設定するには、次の手順に従います。

1. *root* でログインします。
2. named のサンプル・ディレクトリに移動します。

   cd /var/named/Examples

3. テンプレート・ファイルを /var/named ディレクトリにコピーします。

   cp named.boot.slave root.cache localhost.rev /var/named

4. named.boot.slave をデフォルトのファイル名に変更します。

```
cd ..

mv named.boot.slave named.boot
```

5. named.boot を次のように変更します。

```
more named.boot

;
; Boot file for orange.fruit.com, secondary for fruit.com
;
directory /var/named
; type     domain     source host/file  backup file
cache      .          root.cache
secondary  fruit.com  128.70.10.1       fruithosts.bak
```

6. プライマリ・サーバにインストールした localhost.rev ファイルを使用します。
7. プライマリ・サーバにインストールした root.cache ファイルを使用します。
8. 次のコマンドを実行し、named をオンに設定してステーションを再起動します。

```
chkconfig named on

reboot
```

BIND キャッシュ専用サーバの設定

キャッシュ専用サーバを設定するには、次の手順に従います。

1. root でログインします。
2. named のサンプル・ディレクトリに移動します。

```
cd /var/named/Examples
```

3. テンプレート・ファイルを /var/named ディレクトリにコピーします。

```
cp named.boot.master root.cache /var/named
```

4. named.boot.master をデフォルトのファイル名に変更します。

```
cd ..

mv named.boot.master named.boot
```

5. named.boot を次のように変更します。

   more named.boot

```
;
;Boot file for guava.fruit.com,caching-only server for
;fruit.com
;Note that there should be one primary entry for each SOA
;record.
;
;
directory /var/named
;type    domain  source host/file  backup file
cache      .          root.cache
```

6. プライマリ・サーバにインストールした localhost.rev ファイルを使用します。
7. プライマリ・サーバにインストールした root.cache ファイルを使用します。
8. 次のコマンドを実行し、named をオンに設定してステーションを再起動します。

   chkconfig named on

   reboot

BIND フォワード・サーバの設定

フォワード・サーバを設定するには、次の手順に従います。

1. root でログインします。
2. named のサンプル・ディレクトリに移動します。

   cd /var/named/Examples

3. テンプレート・ファイルを /var/named ディレクトリにコピーします。

   cp named.boot.master root.cache localhost.rev /var/named

4. named.boot.master をデフォルトのファイル名に変更します。

   cd ..

   mv named.boot.master named.boot

5. named.boot を次のように変更します。

   more named.boot

   ```
   ;
   ;Boot file for banana.fruit.com, forwarder server
   ;for fruit.com
   ;Note that there should be one primary entry for each
   ;SOA record.
   ;
   ;
   directory       /var/named

   ;type       domain        source host/file   backup file
   cache       .             root.cache
   forwarders  128.70.10.1   128.70.10.2
   ```

6. プライマリ・サーバにインストールした localhost.rev ファイルを使用します。
7. プライマリ・サーバにインストールした root.cache ファイルを使用します。
8. 次のコマンドを実行し、named をオンに設定してステーションを再起動します。

   chkconfig named on

   reboot

BIND スレーブ・サーバの設定

1. root でログインします。
2. named のサンプル・ディレクトリに移動します。

   cd /var/named/Examples

3. テンプレート・ファイルを /var/named ディレクトリにコピーします。

   cp named.boot.slave root.cache localhost.rev /var/named

4. named.boot.master をデフォルトのファイル名に変更します。

   cd ..

   mv named.boot.slave named.boot

5. named.boot を次のように変更します。

```
;
;Boot file for pineapple1.fruit.com, slave server for
;fruit.com
;
directory       /var/named
;type       domain      source host/file   backup file
cache       .           root.cache
forwarders  128.70.10.3
slave
```

6. プライマリ・サーバにインストールした localhost.rev ファイルを使用します。
7. プライマリ・サーバにインストールした root.cache ファイルを使用します。
8. 次のコマンドを実行し、named をオンに設定してステーションを再起動します。

```
chkconfig named on
reboot
```

BIND クライアントの設定

BIND クライアントを設定するには、次の手順に従います。

1. root でログインします。
2. resolv.conf ファイルを作成または変更し、デフォルトのドメイン名、ホスト名を検索する順序、ネーム・サーバのリストを記述します。次のように記述します。

```
domain fruit.com
nameserver 128.70.10.4
nameserver 128.70.10.2
nameserver 128.70.10.1
hostresorder bind local
```

3. 必須ではありませんが、ここでクライアントを再起動することをお薦めします。

BIND 環境の管理

ここでは、BIND データベースの管理方法について説明します。ドメインにステーションを追加したり削除する方法や、新しいサブドメインを追加する方法について説明します。また、BIND データベースを管理するスクリプトについても説明します。

新しい BIND ステーションの追加

新しいステーションをゾーン・ファイルに追加するには、次の手順に従います。

1. ステーションのドメインの該当するゾーン・ファイルを編集します。
2. ステーションの各アドレスごとに *A* レコードを追加します。
3. *CNAME*、*HINFO*、*WKS*、*RP*、および *MX* の各レコードを追加します（オプション）。
4. ステーションが接続しているネットワークのゾーン・ファイルに、ステーション・アドレスごとに逆 IN-ADDR エントリを追加します。

BIND ステーションの削除

ゾーン・ファイルからステーションを削除するには、次の手順に従います。

1. ステーションのドメインのゾーン・ファイルから、削除するステーションに関するすべてのリソース・レコードを削除します。
2. ステーションが接続していたネットワークごとに IN-ADDR ゾーン・ファイルからステーションのすべての *PTR* レコードを削除します。

BIND ドメインの追加

新しいサブドメインをドメインに追加するには、次の手順に従います。

1. ドメイン・サーバまたは新しいゾーン・ファイル、あるいはその両方を設定します。
2. 新しいドメインの各サーバごとに、親ドメインのゾーン・ファイルに *NS* レコードを追加します。
3. 必要なグルー・アドレス・レコードを追加します。グルー・レコードについては、付録 A「BIND 標準リソース・レコードの形式」を参照してください。

named を再ロードするスクリプト

このシェル・スクリプトは、HUP 信号を `named` に送信します。この信号は `named.boot` を読込み、データベースを再ロードします。これにより、以前にキャッシュされていたデータは失われます。このスクリプトは、`named` の実行中に、`named` の内部データベースに変更内容を反映させる場合に使用します。`/usr/sbin/named.reload` スクリプトを使用すると簡単に実行できます。

named を再起動するスクリプト

このシェル・スクリプトは実行中の `named` を終了し、新しい `named` を起動します。*named.boot* ファイルを変更した場合、またはサーバに通知する必要がある場合にこのスクリプトを使用します。`/usr/sbin/named.restart` スクリプトを使用すると簡単に `named` を起動することができます。

named のデバッグ

`named` が正しく動作しない場合は、まず `/var/adm/SYSLOG` に何かメッセージがないかどうか調べます。それ以外の情報を得るには、`SIG` に `INT`、`ABRT`、`USR1` または `USR2` を指定し、`killall(1M)` でその信号を `named` に送信します。

```
/etc/killall -SIG named
```

INT	現在のデータベースとキャッシュを/var/tmp/named_dump.dbにダンプします。このダンプ内容からデータベースのロードが正しかったかどうかを判断します。
ABRT	統計データを /var/tmp/named.stats にダンプします。この統計データは、ファイルに追加されます。
USR1	デバッグ機能をオンにします。USR1 を指定するたびにデバッグ・レベルが順次高くなります。レベルは 10 まであり、レベルを上げるとより詳細な情報が表示されます。検索要求のデバッグには、レベル 5 を指定します。この出力は /var/tmp/named.run に格納されます。
USR2	デバッグ機能を完全にオフにします。

SYSLOG エラー・メッセージ

named は syslog を用いて /var/adm/SYSLOG にエラーを記録します。ここでは、重要なエラー・メッセージとその意味について説明します。

- *dname* has CNAME and other illegal data

  エリアスに *CNAME* レコード以外のデータがあります。たとえば、monet だけに *A* レコードがある場合には、次の内容は正しくありません。

  ```
  ucbmonet IN CNAME monet
  ucbmonet IN A 128.32.0.1
  ```

- Attempted to query myself on ipaddr as name server for *dname*

  named.boot ファイル内で、ステーション自身がフォワード・サーバとして登録されています。

- zoneref: Masters for secondary zone *dname* unreachable

  このステーションは、*dname* のセカンダリ・サーバです。プライマリ・サーバが無効なデータ応答を返したか、またはネットワーク上に問題があるためにプライマリ・サーバと通信できず、ゾーンの現在の状態を得ることができませんでした。

- Lame delegation to *dname1* received from *ipaddr*(purported server for *dname2*) on query on name [*dname3*]

指定したアドレスのリモート・サーバがドメイン *dname2* に対して権限がある場合に、権限がないことを示す応答が返されました。このリモート・サーバまたはその親ドメインのサーバの設定が正しくありません。このエラー・メッセージは、**-L lamedel** オプションを指定して *named* を起動すると、出力されません。

- `MAXQUERIES exceeded, possible data loop in resolving` *dname*

  指定のレコードに対してネーム・サーバが問い合わせたサーバの数が多すぎます。このエラー・メッセージは、2 つのリモート・サーバがお互いに相手のサーバが *dname* の情報を持っていると応答した場合に出力されます。

- `Malformed response from` *ipaddr*

  指定したアドレスのリモート DNS サーバが間違った形式のパケットを返しました。このエラー・メッセージは、リモート・サーバにエラーが発生したことを示します。

- `Bogus root NS` *dname1* `received from` *ipaddr* `on query on name[`*dname2*`] --rejected`

  指定したアドレスのネーム・サーバが間違ってルート・ドメインの NS レコードを返しました。このレコードは無視されます。このエラー・メッセージは、**-L rootns** オプションを指定して *named* を起動すると、出力されません。

- `Root NS` *dname1* `received from` *ipaddr* `on query on name [`*dname2*`]`

  指定したアドレスのネーム・サーバがルート・ドメインの NS レコードを返しました。このエラー・メッセージは、**-L rootns** オプションを指定して *named* を起動すると、出力されません。

nslookup コマンドによるネーム・サーバのデバッグ

nslookup コマンドは、ローカル・ネーム・サーバとリモート・ネーム・サーバに問い合わせるデバッグ用のツールです。nslookup では、対話モードでの問い合わせと非対話モードでの問い合わせが可能です。対話モードでは、各種のステーションやドメインに関する情報をネーム・サーバに問い合わせたり、ドメインにあるステーションのリストを出力したりすることができます。非対話モードでは、単一のステーションまたはドメインの名前や指定した情報を出力します。次の nslookup の例では、ステーション monet.berkeley.edu のアドレス・レコードを出力しています。

```
Default Server:  ucbvax.berkeley.edu
Address:  128.32.133.1
> monet
Server:  ucbvax.berkeley.edu
Address:  128.32.133.1
Name:    monet.berkeley.edu
Address:  128.32.130.6
```

終了するには、**<Ctrl>-D** を押すか **exit** と入力します。その他の機能を知りたい場合は、*help* コマンドを使用します。nslookup コマンドの便利なオプションに、**set type=Any**、**set type=Mx** および **set debug** があります。コマンドの詳細については、nslookup(1C) マン・ページを参照してください。

統一ネーム・サービス

統一ネーム・サービス（UNS: Unified Name Service）とは、Silicon Graphics IRIX オペレーティング・システムが備えるネーム・サービス・レイヤで、ネーム・サービス要求を変換し、簡易化するものです。ネーム・サーバはネットワーク・サービスを提供し、クライアントがネットワーク上のリソースまたはオブジェクトに名前を付けて、その情報をほかのネットワーク・オブジェクトと共有できるようにします。

従来は、新しいネーム・サービスはすべて標準 C ライブラリのコードを使ってアプリケーションに組込まれていました。ネットワークに新しいネーム・サービスが追加されるたびに、システム・リソースとアカウントに関する情報が格納された設定ファイルが、多数のライブラリ・ルーチンと共に追加されました。分散型ネーム・スペース管理という概念の導入により、このプロセスはより複雑になりました。

これを簡易化するため、UNS が統一レイヤを提供する方式が開発されました。IRIX にネットワーク接続されたすべてのプログラムは、UNS を使ってほかのネーム・サービスを実装できます。

この章では、以下について説明します。

UNS について

統一ネーム・サービス（UNS: Unified Name Service）は、多数の異なるプロトコルから発行されるネーム・サービス要求の結果を単一のファイル・ベースのプロトコルに変換します。プロトコルとは、ネットワークのコンポーネント同士の間でデータを転送する方法を規定する規則、データ・フォーマット、規約の集合です。IRIX オペレーティング・システムが備える標準プロトコルの中には、ネーム・サービス要求を扱うプロトコルの DNS と NIS があります。複数のプロトコルが存在する状況では、ほかを無効にする 1 つのプロトコルがあると、ネーム・サービス要求が簡易化されます。

UNS には 3 つの主要な構成要素があります。

- ネーム・サービス・デーモンの `nsd`
- C ライブラリ内の個々のアプリケーション・プログラミング・インタフェース（API）ルーチン
- 数個のプロトコル・ライブラリ

ネーム・サービス API は、ライブラリ・レベルの互換性を保つため、IRIX オペレーティング・システムの前のリリースと同じにしてあります。UNS のコンポーネントを利用するためにアプリケーションを再コンパイルする必要はありません。

UNS の動作の概要

システムの起動時、`nsd` デーモンは 2 つの操作を行います。まず `/ns` をルートとするファイル・システム・ネームスペースを初期化します。その後、サポートされている個々のテーブルとプロトコルの解決順を指定する UNS 設定ファイル `/etc/nsswitch.conf` を読込みます。

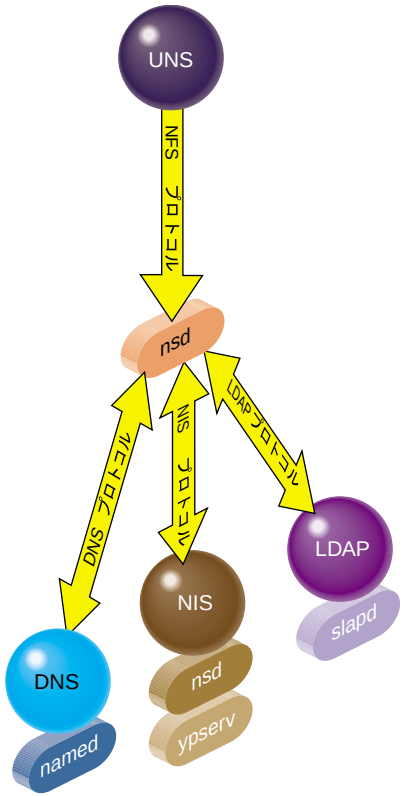

図 7-1 ネーム・サービス・プロトコルに対する nsd デーモンの動作

nsd デーモンは直接プロトコルを処理しません。nsd はいくつかのネーム・サービス・のフロントエンド・サービスを設け、その間の受渡しをします。たとえば、NIS からパスワードをシステム名に解決するような要求を受取ると、これを要求のパス名から決定される、適切なライブラリに渡します。たとえば、NIS が発行した要求 /etc/passwd が /ns/.local/.nis/passwd に渡されるというように、UNS はエリアスとして機能します。

nsd デーモンは呼出されると、プロトコルとテーブルのリストが格納された設定ファイル nsswitch.conf を読込みます。テーブルは検索可能な行と列から構成されています。hosts.byname と hosts.byaddr から構成されるホスト・テーブルによって、ホスト名またはホスト・アドレスのいずれを基準にしても効率的に検索が行えます。

nsd によって起動されるファイル・システム・ネームスペースは、/ns をルートとし、基底にあるインタフェースは、それを使ってネーム・システム・データを取得します。これは 30 秒でタイムアウトする動的ファイル・システムなので、/ns から ls コマンドを実行してもファイルはリストされません。

すべての名前検索の結果は /ns ネームスペースの .local キャッシュ・ファイルに格納されます。.local ファイルは、ローカル・ユーザから見たネームスペースです。/ns ディレクトリでファイルがアクセスされると、/var/ns/cache の下のキャッシュ・ファイルにエントリが追加され、実質的なシャドー・ファイルが作成されます。ネーム・サービス・デーモンがサポートしている個々のテーブルには、それぞれのキャッシュ・ファイルがあります。たとえば、NIS が呼出された場合は、まずローカル・ホスト上の NIS データベース・ファイルが検索されます。この方式によって、ネットワーク・トラフィックが軽減され、検索が高速になります。

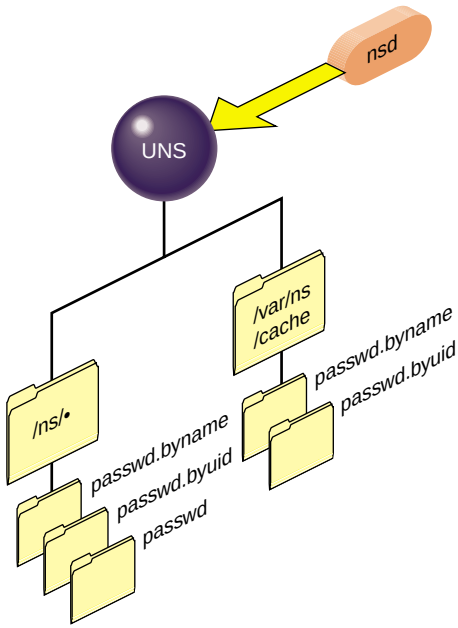

図 7-2　動的な UNS ファイルの部分表示

これらのファイルのフォーマットは、これに置換えられる従来の設定ファイルのフォーマットと同じです。たとえば passwd の場合、/ns/*/passwd.byname ディレクトリの下の各々のファイルは、復帰改行文字で区切られた次のフォーマットの行で構成されます。

```
login:password:uid:gid:gecos:directory:shell
```

`/etc/nsswitch.conf` ファイルがホスト・テーブルを含むサポートされている各々のテーブルの解決順を指定しているため、`/etc/resolv.conf` の hostresorder 行は無視されます。

次の表に、IRIX 6.2 以降のオペレーティング・システムの一部としてサポートされているネーム・サービスと、ネーム・サービスに対応するクライアントとサーバのサービス、クライアントとサーバをリンクするバインディングを示します。

表 7-1 従来からサポートされているサービスのプロトコル

プロトコル	クライアント	バインディング（リンキング）	サーバ
DNS	解決ライブラリ	resolv.conf	named
NIS		ypbind（現在は nsd）	ypserv（現在は nsd）
Files	getx by y() getx by ent()		

初期の UNS リリースで提供されるプロトコルとテーブルは、クライアントサイド DNS、クライアントサイド NIS、ファイル、MDBM、NIS データベース・ファイル、NDBM、BerkeleyDB、`ypserv` の代わりの Nisserv、X500 の簡易版の LDAP です。これらのライブラリの動作については、177 ページの「UNS プロトコル・ライブラリ」で説明します。

次の節では、UNS のさまざまな側面について説明します。

- 「NIS に対する UNS の動作」（168 ページ）
- 「UNS と NIS データベースについて」（170 ページ）
- 「BIND に対する UNS の動作」（171 ページ）
- 「NFS に対する UNS の動作」（172 ページ）
- 「LDAP に対する UNS の動作」（173 ページ）

NIS に対する UNS の動作

NIS ネットワーク検索サービスは、サービスに参加しているシステムに対して、ネットワークに関する情報の集中データベースを提供します。ネットワーク・アプリケーションに対しては多数の情報ソースが提供されていますが、デフォルトの検索順は、通常、最初が NIS、2 番目が DNS (BIND)、3 番目が適切なローカル・ファイルです。

UNS は、NIS の基本的な動作を変更することはありませんが、サーバ・デーモンの `ypserv` とバインダ・デーモンの `ypbind` の動作は変更されます。

NIS デーモンの `ypserv` は、クライアントの問い合わせに応答し、データベースを更新するデータベース・サーバの役割を果たしています。`ypserv` は完全な NIS データベースを備えた NIS サーバ・マシン上でのみ動作します。NIS バインダ・デーモンの `ypbind` は、すべての NIS クライアント上で動作し、`ypserv` との間の通信に必要とされる情報を記憶する役割を果たしていました。システムは同時にサーバとしてもクライアントとしても機能するため、システムでは `ypbind` と `ypserv` の両方を実行できます。

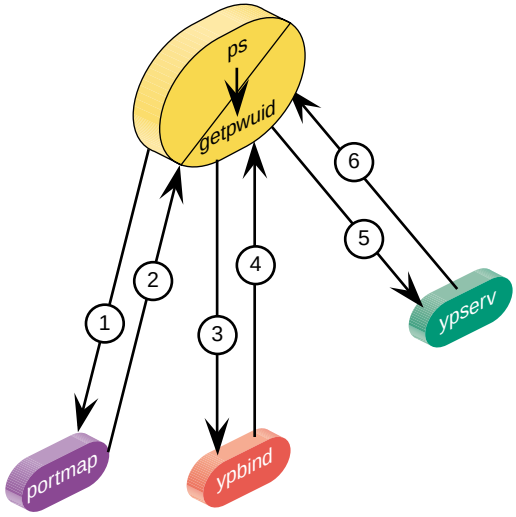

図 7-3 NIS ネーム検索の従来の動作

IRIX オペレーティング・システムの初期のバージョンでは、NIS デーモン・フラグが「on」にセットされていると、マスター・ネットワーク・スクリプト `/etc/init.d/network` が NIS デーモンを起動します。IRIX 6.5 オペレーティング・システムからは、UNS ファイルの `/etc/nsswitch.conf` が NIS クライアントの解決順を制御し、以前の `ypserv` の動作が無効になります。

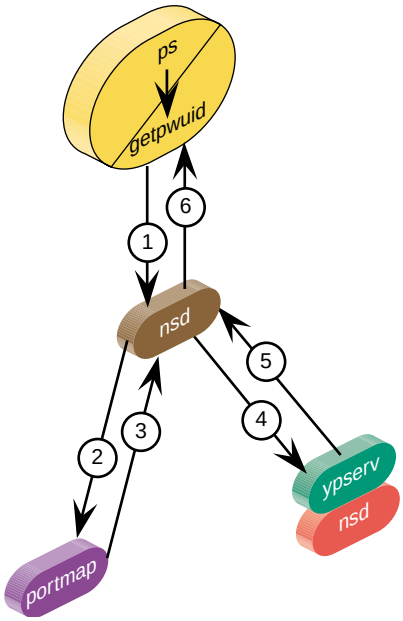

図 7-4 NIS に対する nsd デーモンの動作

`nsd` デーモンは UNS 設定ファイル `/etc/nsswitch.conf` を読込みます。UNS 設定ファイルのデフォルト・コピーには、次の形式の行が格納されています。

```
hosts:nis dns files
```

この例では、NIS ホスト・テーブルは、DNS またはローカル・ファイルの前に参照されます。次に `nsd` デーモンは、`/var/ns/lib` ディレクトリに格納されている共有ライブラリ内の NIS プロトコルを実行します。これについては、177 ページの「UNS プロトコル・ライブラリ」で詳しく説明します。

このほか、NIS サーバ・ユーティリティが変更された点は、`ypinit`、`ypmake`、`make.script` が `mdbm_parse` という perl スクリプトになったことです。makefile やバイナリを使う代わりに、これらのスクリプトが実際の分析を行います。これによってシステムの動作が変化することはありません。

NIS 環境の詳細については、『NIS Administrator's Guide』を参照してください。`nsd` デーモンの動作の詳細については、175 ページの「UNS 設定ファイル」を参照してください。

UNS と NIS データベースについて

NIS データベースは、マップと呼ばれるファイルのグループで構成されています。マップはキーと値で構成されています。たとえば、`hosts.byname` というマップでは、キーは個々のシステムの名前です。

マップを作成するときは、入力ファイル（通常は標準 ASCII ファイル）をデータベース・レコード形式のファイルに変換します。従来は、`dbm` と呼ばれるバックエンド・データベース・ファイルには、`.dir` または `.pag` の拡張子が付く出力ファイルがあり、それら 1 組でマップを構成します。DBM フォーマットのファイルでは、エントリの数と各エントリの相対サイズの両方が限定されます。

UNS 環境のデータベース・バックエンド・ファイルは `mdbm` と呼ばれ、通常出力ファイルの拡張子は `.m` です。データベース自体は 1 つのファイルのみを必要とするため、`.dir` ファイルと `.pag` ファイルという別々のファイルはありません。MDBM フォーマットのファイルは、長さが 64KB（デフォルトは 4KB）までの何百万という数のエントリを扱うことができます。この単純な単一キーのファイル・フォーマットによって、応答が速くなります。

UNS では、NIS バックエンド・データの新しい保存場所は `/var/ns/domains` です。各テーブルは、各キーに対して 1 つのファイルという複数ファイルの形式で格納されています。検索されるキーは、`/var/ns/cache/` にあるローカル・ハッシュ・ファイルにキャッシュされます。

BIND に対する UNS の動作

Berkeley インターネット・ネーム・ドメイン（BIND: Berkeley Internet Name Domain）は、IRIX オペレーティング・システムにインターネットのドメイン・ネーム・サービス（DNS: Domain Name Service）を実装します。すべての IRIX ネットワーク・プログラムは、元のホスト・テーブルの代わりに BIND を使って、`/etc/hosts` ファイルにある情報を検索できます。

BIND サーバ構成は `named` サーバ・デーモンを実行します。`named` サーバ・デーモンは、`named` 構成フラグが「on」に設定されていれば、ステーションの起動時に自動的に起動します。詳細については、chkconfig(1M) マン・ページを参照してください。

通常の BIND サーバ上の解決順は、`resolv.conf` ファイルによって決定されます。以下にその例を示します。

```
noodle% cat resolv.conf
search example1.com example2.com
#hostresorder nis bind local # now defunct under UNS
nameserver 192.99.89.54 #bindle
nameserver 127.0.0.1
```

基本的な BIND クライアントは通常、クライアントの `resolv.conf` ファイルで指定されているネーム・サーバからデータにアクセスするため、`named` ドメイン・サーバは実行しません。

IRIX 6.5 オペレーティング・システム以降では、UNS ファイル `/etc/nsswitch.conf` が BIND クライアントの解決順を制御するため、`resolv.conf` の解決順は使用されません。デフォルトの `/etc/nsswitch.conf` 行は、ホスト・テーブルの参照を示しています。

```
spanker% cat nsswitch.conf
.....
group: files nis
hosts: nis dns files
```

UNS は DNS クライアント機能を備えています。

UNS 環境では、nsd デーモンが /var/ns/lib ディレクトリに格納されている共有ライブラリの DNS プロトコルを実行します。これについては、177 ページの「UNS プロトコル・ライブラリ」で詳しく説明します。DNS の詳細については、第 6 章「BIND ネーム・サーバ」を参照してください。

NFS に対する UNS の動作

NFS は、ユーザ・データグラム・プロトコル（UDP: User Datagram Protocol）または伝送制御プロトコル（TCP: Transmission Control Protocol）が動作しているネットワーク上で使用できるアプリケーション・レイヤ・サービスです。NFS では、ネットワーク上のすべてのシステムから /usr/local のようなファイルの 1 つのコピーにアクセスできます。各ユーザのシステムからはリモート・ファイルがローカル・ファイルのように見えます。NFS サーバ・デーモンの nfsd は、サーバ上で動作し、クライアントからの RPC コールを受付けます。NFS は、一部の操作をわざわざサーバまで動員しなくても実行できるように、クライアント・サイドにファイル属性をキャッシュします。この方式はフロントエンド・ファイル・システムと呼ばれています。

ls -l のようなファイル操作の要求があっても、ファイル内部のデータは使われません。渡される情報は、ファイルの属性（たとえば、サイズ、アクセス時刻、所有者）に関する情報であり、これらの属性は約 3 秒間有効です。クライアントの変更はローカルでキャッシュされたコピーに対して加えられ、ファイルの属性が約 60 秒間変化しないと、属性はキャッシュから消去され、サーバの元の位置に書込まれます。バックエンド・ファイル・システムと呼ばれるサーバは、アーカイブに格納された最終的なデータのコピーを保管します。

IRIX 6.5 オペレーティング・システムより、UNS 環境は /ns/.local に格納されるローカル・ファイルを提供します。これは、ユーザからはネームスペースに見えます。ほかのネーム・サービスでは、/ns/.local ディレクトリでファイルがアクセスされると、/var/ns/cache の下にあるキャッシュ・ファイルにキャッシュ・エントリが追加され、シャドー・ファイルが作成されます。nsd デーモンは、マウントされたファイル・システムの状態を表示します。実際には、リモートの NFS サーバは存在しません。

以前は NFS（Network File System）と呼ばれていた、Sun Microsystems の ONC+（Open Network Computing Plus）分散サービスの Silicon Graphics バージョンの詳細については、『ONC3/NFS Administrators Guide』で説明しています。

LDAP に対する UNS の動作

LDAP（Lightweight Directory Access Protocol）は、TCP/IP で実行される X.500 Directory Access Protocol の簡易版です。LDAP の魅力は、ユーザがデータベースの方式を定義し、情報をどのように整理し、データベースから引出すかを指定できる点です。もちろん、データベースを使うには LDAP クライアントがこの方式を知っている必要があるので、この情報は LDAP クライアント設定ファイル `/etc/ldap-ns.conf` に格納されます。

システムの起動時に `nsd` デーモンは、サポートされている個々のテーブルとプロトコルの解決順を指定する設定ファイル `/etc/nsswitch.conf` を読込みます。リストの中に LDAP があれば、LDAP クライアント設定ファイル `/etc/ldap-ns.conf` が読込まれます。

このファイルの中には、LDAP プロトコル・ライブラリを使用している個々のドメイン専用のセクションがあり、そのセクションでは、使用する LDAP サーバ、各サーバ上で検索するパラメータ、LDAP 検索フィルタに各ネーム・サービス要求をマップする方法、応答をファイル・フォーマットに変換する方法が指定されています。LDAP はこのように柔軟性が高いので、構成情報の保管にはどんな方式でも使用できます。LDAP の技術的な情報については、RFC 1777 を参照してください。設定ファイルの詳細については、ldap-ns.conf(4) マン・ページを参照してください。

ネームスペースのフォーマット

ネームスペースのフォーマットは /ns/domain/table/protocol/key です。プロトコル・ディレクトリは省略できます。プロトコル・ディレクトリを省略すると、/etc/nsswitch.conf で指定されている順に従ってすべてのプロトコルが検索されます。.local ドメインは、ローカル・システムから見たネームスペースを表しています。特殊ファイル .all には、テーブルのすべてのデータがリストされます。表 7-2 にファイルとファイルの目的を示します。

表 7-2 UNS ファイルとファイルの目的

ファイル名	目的
/ns/.local/passwd.byname/root	サポートされている任意のプロトコルを使うローカルの passwd のルート・エントリ
/ns/.local/passwd.byname/.nis/root	NIS プロトコルのみを使うローカルの passwd のルート・エントリ
/ns/.local/passwd.byname/.nis/.all	ローカル・ドメインの NIS passwd テーブルのすべてのパスワード・エントリ
/ns/engr/passwd.byname/root	任意の使用可能なプロトコルを使う engr ドメインの passwd マップのルート・エントリ
/ns/sgi.com/hosts.byname/sgi	sgi.com ドメインから sgi マシンのホスト・アドレスを見つける（sgi.com はローカルでなければならない）。
/ns/.local/hosts.byname/sgi.sgi.com	ホスト・マップ内で sgi.sgi.com マシンのホスト・アドレスを見つける（データはどこにあってもよい）。

たとえば、engr.example1.com ドメインのルート・ユーザのパスワード・エントリを検索するには、次のコマンドを実行します。

```
# cat /ns/engr.example1.com/passwd.byname/root
```

.local ディレクトリは、ローカル・ドメインに対して作成されるため、ローカル・ドメインの root パスワード・エントリは常に /ns/.local/passwd.byname/root にあります。

各テーブル・ディレクトリの .all ファイルにはすべてのパスワード・テーブルがリストされます。nsswitch.conf にリストされているすべてのライブラリ・ルーチンを使用するローカル・ドメインの各パスワード・エントリのリストが必要な場合には、次のコマンドを実行します。

```
# cat /ns/.local/passwd.byname/.all
```

このコマンドを実行することにより、ローカル・ドメイン上のすべてのユーザの情報を表示することができます。

各テーブル・ディレクトリの下に、nsswitch.conf にリストされているそのテーブルの各ライブラリに対して、.nis にような特殊ディレクトリの *.library* が作成されます。

ns_lookup() ライブラリ・ルーチンは、ネーム・サービス検索からの要求を満たすときに、常に /ns にマウントされている .local ドメイン・ネームスペースの下にあるファイルを開くので、これを変更することはできません。

UNS 設定ファイル

UNS 設定ファイルは /etc/nsswitch.conf の 1 つだけで、次のフォーマット行で構成されています。

```
map: library library library
```

例

```
hosts: nis dns files
```

このファイルは、ライブラリとライブラリの使用順を指定します。パス要素のどれかが欠けていると、nsd デーモンはその要素が見つかるまで、ネーム・サービス・ライブラリ・ルーチンを指定された順に呼出します。

システム管理者は nsd が動作していることだけを確認するのみです。183 ページの「nsd のトラブルシューティング」を参照してください。構成フラグが「on」にセットされていれば、通常、nsd デーモンは /etc/init.d/network によりシステムの起動時に起動されます。chkconfig(1M) を参照してください。nsd 構成フラグが「on」に設定されていない場合には、サポートされるネーム・サービスは、ローカル・ファイルのみです。nsd デーモンは各ネーム・サービス要求をパス名に変換します。

nsd は起動時に /ns をルートとする動的ファイル・システム・ネームスペースを作成し、基底にあるインタフェースはそれを利用してネーム・システム・データを取出します。

これらのファイル・フォーマットは、これに置換えられる従来の設定ファイルのフォーマットと同じです。たとえば passwd の場合、/ns/*/passwd.byname ディレクトリの下のすべてのファイルは、復帰改行文字で区切られた以下のフォーマットの行で構成されます。

```
login:password:uid:gid:gecos:directory:shell
```

/etc/nsswitch.conf ファイルがホスト・テーブルを含むサポートされているすべてのテーブルの解決順を指定しているため、/etc/resolv.conf の hostresorder 行は無視されます。

このデーモンがサポートしている各ドメインごとに 1 つの nsswitch.conf ファイルがあります。各マシンにはローカル・ドメインの .local があり、ローカル・ドメインの設定ファイルは /etc/nsswitch.conf にあります。サーバ・マシンは複数ドメインをサポートしており、各ドメイン名の設定ファイルは /var/ns/domains/domainname/nsswitch.conf にあります。

UNS 構成の設定

IRIX オペレーティング・システムをインストールすると、自動的に UNS が構成されます。nsd デーモンが動作し、/etc/nsswitch.conf ファイルを使って、要求されているネーム・サービスの構成とプロトコルを特定します。このデーモンがサポートする各ドメインごとに 1 つの nsswitch.conf ファイルがあります。各マシンにはローカル・ドメインの .local があり、ローカル・ドメインの設定ファイルは /etc/nsswitch.conf にあります。サーバ・マシンは複数ドメインをサポートしており、各ドメイン名の設定ファイルは /var/ns/domains/domainname/nsswitch.conf にあります。

ただし、システムを NIS サーバとして設定するには、さらに次の操作が必要です。

1. システムを NIS サーバとして設定するには、ypinit を実行します。デフォルトの nsswitch.conf ファイルが所定の位置にコピーされ、システム設定ファイルが分析され、mdbm ハッシュ・ファイルに変換されます。詳細については ypinit(1M) マン・ページを参照してください。
2. NIS データベースを再構築し、配布するには、ypmake を実行します。詳細については ypmake(1M) マン・ページを参照してください。
3. mdbm ハッシュ・データベースの現在の内容を表示するには、mdbm_dump を実行します。詳細については mdbm_dump(1M) マン・ページを参照してください。

UNS プロトコル・ライブラリ

UNS の最初のリリースには、以下のプロトコル・ライブラリが提供されます。

- ファイル。178 ページの「ファイル・ライブラリ」を参照してください。
- ネットワーク・インフォメーション・サービス（NIS: Network Information Service）。179 ページの「NIS プロトコル・ライブラリ」を参照してください。
- ドメイン・ネーム・サービス（DNS: Domain Name service）。179 ページの「DNS プロトコル・ライブラリ」を参照してください。
- ローカル・ハッシュ・ファイル（MDBN）。179 ページの「MDBM プロトコル・ライブラリ」を参照してください。
- NDBM（MDBN の初期バージョン）。180 ページの「NDBM プロトコル・ライブラリ」を参照してください。
- Berkeley DB（Berkely ハッシュ・ファイル）。180 ページの「Berkeley DB プロトコル・ライブラリ」を参照してください。
- Nisserve（ypserv の代わり）。180 ページの「Nisserv プロトコル・ライブラリ」を参照してください。
- LDAP（X500 の代わりの簡易版）。181 ページの「LDAP プロトコル・ライブラリ」を参照してください。

図 7-5 に示すように、nsd デーモンはプロセスのパス名の解決に基づいて、特定のプロトコル・ライブラリに要求を送ります。

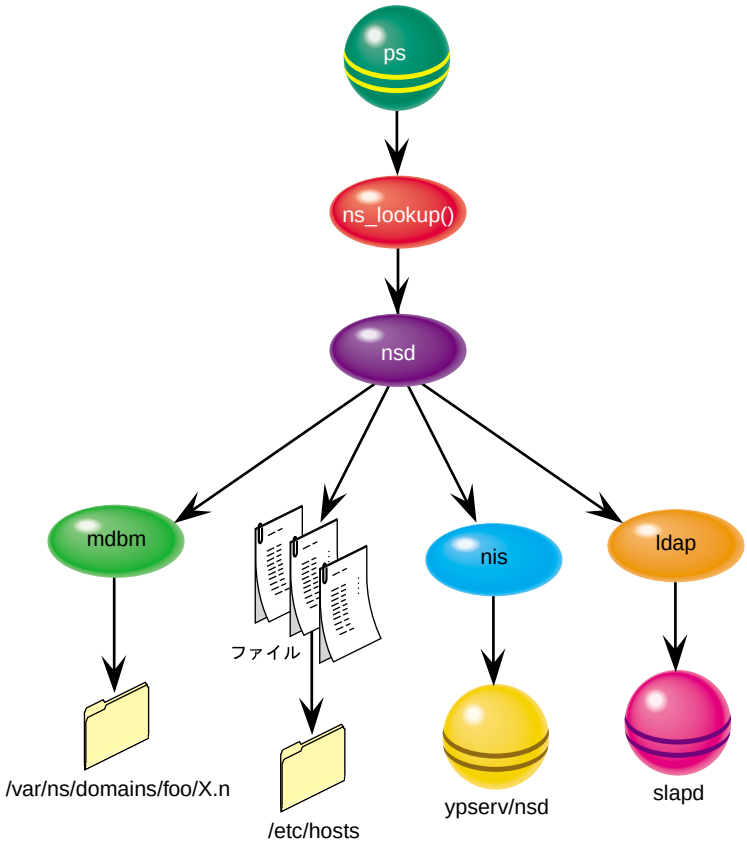

図 7-5 ns_lookup がプロトコル・ライブラリを選択するときの動作

ファイル・ライブラリ

ファイル・ライブラリは、passwd や hosts などの従来のフォーマットの設定ファイルを分析します。.local ドメイン内で検索が行われると、/etc 内のファイルが開かれ、検索されます。ほかのドメインで検索が行われると、要求先のドメインの /var/ns/domains/domainname の下の同じ名前のファイルが開かれます。属性の詳細については、181 ページの「UNS ファイル構造属性」を参照してください。また files(7P) マン・ページも参照してください。

NIS プロトコル・ライブラリ

NIS プロトコル・ライブラリは、NIS プロトコルのバージョン 2 を使ってリモートの `ypserv` デーモンに要求を送ります。NIS プロトコル・ライブラリは、IRIX オペレーティング・システムでは `ypbind` デーモンの代わりになります。NIS プロトコルをサポートしているテーブルが 1 つでもあれば、ライブラリは新しいサービス・ソケットを作成し、そのサービス・ソケットが `yp_match()` や `yp_all()` などの旧スタイルの NIS API ルーチンからの NIS バインド要求に応答します。

.local ドメインでキーが検索されると、`domainname` コマンドで指定されるシステムの現在のドメイン名が使用されます。通常、これは `/var/yp/ypdomain` ファイルで設定されます。ドメインが指定されていない場合には、`ypbind` プロトコルを使用してサービスが直接提供されます。マルチキャスト・バインディングで使用できる hops の最大数は、**`nis_multicast`** 属性を使用して指定できます。この属性のデフォルト値は、32 です。182 ページの「ドメイン属性の設定」を参照してください。

DNS プロトコル・ライブラリ

DNS プロトコル・ライブラリは、DNS プロトコルを使ってリモートの `named` デーモンに要求を送ります。DNS ライブラリは `/etc/resolv.conf` ファイルを分析してデフォルトの検索パス、ネーム・サーバ・リスト、オプションを特定します。.local ドメインでキーが検索された場合には、検索パスに従い、それ以外の場合は特定のドメインが使用されます。これらの属性を設定する方法については、182 ページの「ドメイン属性の設定」を参照してください。

MDBM プロトコル・ライブラリ

MDBM プロトコル・ライブラリは、ローカル・ホストの `mdbm` ハッシュ・データベース・ファイルのデータを検索します。これらのファイルの名前は、テーブルと同じ名前に `.m` 拡張子が付きます（例：`hosts.byname.m`）。これらのファイルは `.local` ドメインでは `/etc` ディレクトリにあり、これ以外は各ドメイン名の `/var/ns/domains/domainname` ディレクトリにあります。フラット・ファイルは nisserv ライブラリと共に提供される `mdbm_parse` スクリプトを使って分析し、`mdbm` ハッシュ・ファイルに変換できます。詳細については、mdbm(3B) マン・ページおよび 182 ページの「ファイル属性の設定」を参照してください。

NDBM プロトコル・ライブラリ

NDBM プロトコル・ライブラリは、ローカル・ホストの `ndbm` ハッシュ・データベース・ファイルのデータを検索します。これらのファイルの名前は、テーブルと同じ名前に `.pag` および `.dir` 拡張子が付きます（例：`hosts.byname.pag` および `hosts.byname.dir`）。これらのファイルは `.local` ドメインでは `/etc` ディレクトリにあり、これ以外は各ドメイン名の `/var/ns/domains/domainname` ディレクトリにあります。ndbm(7P) マン・ページを参照してください。また 182 ページの「ファイル属性の設定」の説明も参照してください。

Berkeley DB プロトコル・ライブラリ

Berkeley DB プロトコル・ライブラリは、ローカル・ホストの `db` ハッシュ・データベース・ファイルのデータを検索します。これらのファイルの名前は、テーブルと同じ名前に `.db` 拡張子が付きます（例：`hosts.byname.db`）。これらのファイルは `.local` ドメインでは `/etc` ディレクトリにあり、これ以外は各ドメイン名の `/var/ns/domains/domainname` ディレクトリにあります。berkeleydb(7P) マン・ページを参照してください。また 182 ページの「ファイル属性の設定」の説明も参照してください。

Nisserv プロトコル・ライブラリ

IRIX オペレーティング・システムでは、nisserv プロトコル・ライブラリが `ypserv` デーモンを置換えます。このライブラリは、ローカル・ディスク上の MDBM ハッシュ・ファイルのデータを検索し、データを NIS プロトコルのバージョン 2 を使ってリモート・マシンに提供します。ファイルの命名規則は MDBM のファイルと同じです。**`nis_secure`** 属性が設定されている場合、またはマップ・ファイルに YP_SECURE キーがある場合、クライアント・システムは 1024 未満のポート番号を使う必要があります。これにより、機密情報が格納されているマップに対して弱いセキュリティが提供されます。マップ内には、NIS プロトコル・プロシージャにデータを提供するための別の特殊キーが多数存在します。これらのキーには YP_MASTER_NAME、YP_LAST_MODIFIED、YP_INPUT_FILE、YP_OUTPUT_NAME、YP_DOMAIN_NAME が含まれます。これらのキーは前のバージョンの NIS で使われていた従来のキーです。詳しい情報は nisserv(7P) マン・ページを参照してください。

LDAP プロトコル・ライブラリ

LDAP プロトコル・ライブラリは、ローカル・ネーム・サービス要求をリモート LDAP プロトコル・パケットに変換し、結果をフォーマットして、標準ネーム・サービス API が使用できるようにします。すべてのサーバ情報とフォーマット規則は単一の設定ファイル `/etc/ldap-ns.conf` から読込まれます。このファイルはライブラリが初期化されるとき、またはネーム・サーバ・デーモンが SIGHUP 信号を受信するたびに読込まれます。

UNS ファイル構造属性

`nsd` デーモンはメモリ内ファイル・システムを実現します。ファイル構造は各ファイルの状態を表し、各ネーム・サービス・ファイルでは拡張属性がサポートされています。ファイルの属性は、属性を検索したライブラリによって異なりますが、属性には常にドメイン、テーブル、キー、タイムアウト、ソース、バージョン、サーバが含まれます。属性は親ディレクトリから継承されます。**`timeoout`** のような属性がディレクトリ上に存在する場合、特定のファイル用に無効化されないかぎり、そのディレクトリ内のすべてのファイルが同じ **`timeout`** 値をもちます。グローバル属性、タイムアウト、ホスト名、プログラムは、コマンド行引数によって設定できます。

この節では、さまざまな属性を設定したり照会する方法を説明します。

- 「ドメイン属性の設定」（182 ページ）
- 「テーブル属性の設定」（182 ページ）
- 「ライブラリ属性の設定」（182 ページ）
- 「属性の照会」（182 ページ）

`nsd` に対する属性は、`nsd -a key=value` のようなコマンド行を入力するか、`nsswitch.conf` ファイルで指定します。何か変更を行った場合には、次のコマンド行を実行して `nsd` を再起動する必要があります。

```
# killall -HUP nsd
```

ドメイン属性の設定

nsswitch.conf ファイルでは、次の形式でドメインの属性を設定します。

```
(key1=value1, key2=value2)
```

テーブル属性の設定

テーブル属性はファイル内のテーブル名によって設定されます。

```
passwd(timeout=10): nis files
```

ライブラリ属性の設定

テーブルのライブラリ属性はライブラリ名で設定されます。

```
passwd: nis(timeout=10) files
```

ファイル属性の設定

特定のファイル属性はファイル名によって指定されます。

```
/ns/.local/passwd.byname/root(timeout=10)
```

属性はライブラリ・ルーチンによって自由に変更できます。属性が指定されていない場合は、デフォルト値が使われます。

属性の照会

属性は attr コマンドを使ってリストまたは照会することができます。

```
attr -l /ns/.local/passwd.byname/root
attr -g timeout /ns/.local/passwd.byname/root
```

キャッシュ・チューニング

キャッシュのサイズは表 7-3 のパラメータによって制御でき、nsswitch.conf ファイルで設定します。

表 7-3 キャッシュ・チューニング・パラメータ

パラメータ	説明	デフォルト／ソース
size	総ページ数の上限の二乗	8
timeout	キャッシュが無効になるまでの秒単位の時間	300
pagesize	2 の累乗のバイト数	4 Kbytes
mode	許可、変更可能	0755
library	データを提供したライブラリの名前。nsswitch.conf で指定される。	xattr コマンドによって設定

nsd のトラブルシューティング

トラブルシューティングの一般的なアプローチ

多くの場合は、システム・ログの SYSLOG にエラーが記録され、解決法が示されます。システム・ログはデスクトップから取得できます。エラー・メッセージは次のような形で表示されます。

```
NFS3 server (local host) nsd not responding

NFS3 server local host (nsd) not responding

Stale NFS File Handle
```

リモート・サービス・エラー

ping や rlogin などのリモート・サービスで定期的にまたはランダムにエラーが発生する場合は、nsd が動作していない可能性があります。nsd の状態をチェックするには、以下を入力します。

```
ps -ef | grep nsd
```

実行中のプロセスのリストの中に nsd がない場合は、nsd を再起動する必要があります。プロンプトから nsd を入力すれば再起動します。

nsswitch.conf の間違い

発生頻度は少ないですが、次のエラー・メッセージが表示されることもあります。

```
no such protocol
```

この場合は、nsswitch.conf の記述にスペルの間違いがないかどうかをチェックしてください。

nsd 信号の解読

nsd がサポートする信号の中には、分かりやすいものがあります。次のコマンドで **SIG** に **USR1**、**USR2**、**HUP**、あるいは **TERM** を指定して nsd を送信します。

```
kill -SIG nsd
```

USR1 強制的に nsd で状態ファイルの /usr/tmp/nsd.dump を作成します。このファイルは、その時点でのファイルシステムの状態が記録されたバッチファイルです。

USR2 ログ・レベルを増分します。コマンドを繰返すたびに、ログが増えます。レベルは 0 から 6 で、ログの記録頻度が高いのは 6 です。

HUP .config ファイルが変更されると HUP が送信され、システムがその変更を読取る必要があります。

TERM nsd を中止する場合に TERM を送信します。受信可能な信号を送信する必要があります。**SIG TERM nsd** は強制的に終了します。

ファイルシステムにマウントできなくなる可能性があるため、kill -kill、kill -9、kill -all は使用しないでください。

/ns/.local の確認

クライアント・ファイルがインストールされていることを確認するには、次のコマンドを発行します。

```
# cat /ns/.local/hosts.byname/localhost
```

表に示す例と類似した出力が表示されます。

```
127.0.0.1 local host
```

UUCP

UUCP（UNIX-to-UNIX Copy Program）は、UNIX オペレーティング・システム（IRIX など）を使用するコンピュータ間の通信、またはリモート端末との通信を行う各種ユーティリティを指します。これらのユーティリティには、`uucp` や `uuto` などコンピュータ間でファイルをコピーするものから、`uuencode` や `uudecode` など簡単なエンコード化とデコード化を行うもの、また、`cu` や `uux` などリモート・ログインやコマンドを実行するものが含まれます。

この章では、以下について説明します。

UUCP システムは、IRIX 提供ソフトウェアの `eoe.sw.uucp` パッケージの `eoe` サブシステムに含まれています。このサブシステムがインストールされているかどうかは、`versions` コマンドで確認します。

電話回線やモデムを介して UUCP 接続を実行すると、USENET ネットワーク上にある何千ものコンピュータ間で電子メールを交換したり、ネット・ニュースを配布することができます。

システム管理者は、UUCP で使用する管理用ツール、ログ、データベース・ファイルに精通している必要があります。この章では、UUCP ファイル、ディレクトリ、デーモン、コマンドについて詳しく説明します。

TCP/IP または UUCP の選択

ここでは、いろいろな観点から UUCP と TCP/IP プロトコルを比較します。これらのプロトコルの両方を使用するよう設定し、タスクに合わせて使分けることが可能です。UUCP と TCP/IP は、いずれも IRIX オペレーティング・システムの標準ソフトウェアです。TCP/IP ソフトウェアを使用する場合は、次のいずれかを備えている必要があります。

- イーサネット・ネットワークとの接続
- オプションの FDDI ハードウェアとソフトウェア
- シリアル回線インターネット・プロトコル（SLIP: Serial Line Internet Protocol）ソフトウェア

UUCP を使用する場合は、シリアル・ネットワークまたは TCP/IP ネットワークと接続する必要があります。

TCP/IP は信頼性の高い対話型サービスとバッチ・サービスを提供します。UUCP はバッチ・モードのプログラムです。つまり、`uucp` コマンドを実行すると、このコマンドがほかのコマンドと一緒に待ち行列に入れられます。システムは一定の間隔でこの待ち行列を調べ、コマンドが見つかるとそれを実行します。コマンドが実行されると、UUCP はその結果を報告します。リモート・ステーションでコマンド実行に要する時間はステーションによって異なります。表 8-1 は、TCP/IP と UUCP の特徴を比較したものです。

表 8-1 TCP/IP と UUCP の比較

TCP/IP の特徴	UUCP の特徴
Ethernet、FDDI、シリアル回線を介して実行	シリアル回線を介して、または TCP/IP リンクで実行
対話式でファイルを転送	バッチ・モードでファイルを転送
コマンドをリモート・システム上で対話式で実行	コマンドをリモート・システム上でバッチ・モードで実行
メールを対話式またはバッチ・モードで送信	メールをバッチ・モードで送信
シェルをリモート・ステーション上で起動	シェルをリモート・ステーション上で起動
リモート・ログインを `rlogin`/`telnet` で実行	リモート・ログインを `cu` で実行
TCP/IP を実行している任意のステーションにデータを転送	UUCP を実行している任意のステーションにデータを転送

UUCP でのハードウェアの必要条件

UUCP を介してほかのコンピュータと通信するには、ハードウェアを設定して通信リンクを確立します。必要となるケーブルとハードウェアは、コンピュータの接続方法、つまり、直接リンク、電話回線、またはローカル・エリア・ネットワークのいずれを使用しているかにより異なります。

メモ：モデムなどのハードウェアの設定については、『IRIX Admin: Peripheral Devices』と『Personal System Administration Guide』を参照してください。

直接リンク　ほかのコンピュータと直接リンクするには、2 台のコンピュータ上のシリアル・ポート間にケーブルを接続します。2 台のコンピュータが定期的に通信する場合、または通信距離が 50 フィート（約 15 メートル）以内と物理的に近い場合は、直接リンクを採用します。近距離用モデムを使用すると、この距離を多少延長することができます。コンピュータが直接リンクされていると、最高で毎秒 38,400 ビット（bps）の速度でデータを転送することができます。しかし、最近では、ローカル・エリア・ネットワークの方が高速で使いやすくなっているため、直接リンクはあまり採用されていません。

電話回線　電話番号をダイヤルできるモデムを使用すると、標準電話回線を介してほかのコンピュータと通信することができます。モデムは、ネットワーク・ユーティリティによって要求された電話番号をダイヤルします。この場合、通信相手であるコンピュータが、受信した呼出しに応答できるモデムを備えていることが必要です。

UUCP コマンド

UUCP プログラムは、ユーザ・プログラムと管理プログラムの 2 つのカテゴリに分類されます。次に、それぞれのカテゴリに属するプログラムについて説明します。

UUCP ユーザ・プログラム・コマンド

UUCP ユーザ・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パーミッションは必要ありません。各ユーザ・プログラムについては、それぞれのマン・ページを参照してください。

cu	コンピュータをリモート・コンピュータと接続し、ログインします。また、初期リンクを解除せずに、ファイル転送やコマンド実行をどちらのコンピュータ上でも行うことができます。
uucp	あるコンピュータから別のコンピュータにファイルをコピーします。作業ファイルとデータ・ファイルを作成し、転送するジョブを待ち行列に入れ、uucico デーモンを呼出します。このデーモンは、リモート・コンピュータとの通信を試みます。
uuto	あるコンピュータから別のコンピュータのパブリック・スプール・ディレクトリ（/var/spool/uucppublic/receive）にファイルをコピーします。リモート・コンピュータのアクセス可能なディレクトリにファイルをコピーする uucp とは異なり、uuto は適切なスプール・ディレクトリにファイルをコピーし、uupick でメールをピックアップするように要求したリモート・ユーザにメールを送信します。
uupick	uuto を使用しているコンピュータにファイルが転送されると、/var/spool/uucppublic/receive に格納されているファイルを取出します。
uux	リモート・コンピュータでのコマンド実行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します。作業ファイルには、uucp と uuto が作成した作業ファイルと同じ情報が格納されています。実行ファイルには、リモート・コンピュータで実行されるコマンド文字列と、データ・ファイルのリストが格納されています。データ・ファイルはコマンドの実行に必要なファイルです。
uustat	要求された転送（uucp、uuto または uux）のステータスを表示します。このプログラムは、待ち行列に登録されている転送を制御する方法を提供します。

UUCP 管理プログラム

UUCP データベース・ファイルは /etc/uucp にありますが、管理プログラムの大半は /usr/lib/uucp にあります。唯一の例外は uulog であり、これは /usr/bin にあります。これらのコマンドについては、それぞれのマン・ページを参照してください。

UUCP を管理するには、uucp ログイン ID を使用します。この ID が基本的なネットワーク・ファイルとスプール・データ・ファイルを所有しているからです。uucp ログイン ID のホーム・ディレクトリは、/usr/lib/uucp です。もう 1 方の UUCP ログイン ID は nuucp であり、この ID は独自のログイン ID を持たないリモート・コンピュータがローカル・コンピュータにアクセスするときに使います。nuucp でログインするコンピュータは、シェルとして uucico を使います。

次に示すのは UUCP の管理ユーティリティです。

uulog　指定されたコンピュータのログ・ファイルの内容を表示します。ログ・ファイルは、ローカル・コンピュータが通信するリモート・コンピュータごとに作成されます。ログ・ファイルには、実行した uucp、uuto、uux がそれぞれ記録されます。

uucleanup　スプール・ディレクトリをクリアします。通常、このコマンドはシェル・スクリプト uudemon.cleanup から実行されます。このスクリプトは cron によって起動します。

Uutry　呼出し機能をテストし、適度なレベルのデバッグを行います。このコマンドはデバッグ・モードで uucico デーモンを起動し、ローカル・コンピュータと指定したリモート・コンピュータ間に通信リンクを確立します。

uucheck　UUCP のディレクトリ、プログラム、およびサポート・ファイルの有無を調べます。また、Permissions ファイルに明らかな構文の誤りがないか調べます。

genperm　各リモート・ステーションに独自のログイン ID を割当てるステーションに対し、Permissions ファイルを作成します。

次のプログラムは、各種モデムの初期化に使用されます。これらのプログラムの使用方法については、『IRIX Admin: Peripheral Devices』の第 1 章「端末とモデム」を参照してください。

fix-dsi　このプログラムは DSI モデムを初期化します。

fix-hayes　このプログラムは Hayes モデムを初期化します。

`fix-intel` このプログラムは Intel モデムを初期化します。

`fix-telebit` このプログラムは Telebit モデムを初期化します。

`fix-usr` このプログラムは U. S. Robotics モデムを初期化します。

`fix-zyxel` このプログラムは Telebit モデムを初期化します。

UUCP デーモン

UUCP には次のようなデーモンがあり、ファイルを転送し、コマンドを実行します。これらのデーモンは、シェルから手作業で起動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・コンピュータとのリンクを確立し、必要なログイン・シーケンスとパーミッション・チェックを行い、データを転送し、ファイルを実行し、結果を記録し、`mail` で転送が完了したことをユーザに通知します。また、要求されたコマンドを実行するために、`uuxqt` を起動します。ローカル `uucico` デーモンは、リモート・コンピュータを呼出すとそのセッションの間、リモート・コンピュータの `uucico` デーモンと対話します。

`uucico` デーモンは、リモート・コンピュータとの通信に必要なすべてのファイルが作成された後に、`uucp`、`uuto`、および `uux` プログラムによって実行されます。このデーモンは、`uusched` や `Uutry` プログラムからも実行されます。

`uuxqt` リモート・システムから要求されたジョブを実行します。このデーモンはスプール・ディレクトリを検索し、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`）を見つけます。`X.file` が見つかると、`uuxqt` はそのファイルを開き、実行に必要なデータ・ファイルのリストを読込みます。次に、必要なデータ・ファイルが存在し、それがアクセス可能かどうか確認します。ファイルが存在し、アクセス可能であれば、`uuxqt` が `Permissions` ファイルを調べ、要求されたコマンドを実行する権限があるどうか確認します。`uuxqt` デーモンは、`cron` で起動する `uudemon.hour` シェル・スクリプトによって実行されます。

`uusched` スプール・ディレクトリに格納されている待ち行列内のジョブ・スケジュールを管理します。`uusched` は、`uucico` デーモンを起動する前にリモート・コンピュータを呼出す順番をランダム化します。`uusched` は、`cron` で起動する `uudemon.hour` シェル・スクリプトによって実行されます。

`uugetty` このプログラムは、回線（ポート）が双方向で使用できることを除けば、`getty` と同じです。ポートを双方向で使用したい場合は、`uugetty` を `/etc/inittab` ファイルのポートに割当てます。`uugetty` については、uugetty(1M) マン・ページを参照してください。

UUCP のサポート・データベース

UUCP サポート・ファイルは、`/etc/uucp` ディレクトリにあります。

`Devices` 自動呼出し装置（モデム）と直接リンクのデバイスおよび回線速度に関する情報が記述されています。詳細については、194 ページの「UUCP Devices ファイル」を参照してください。

`Dialers` リモート・コンピュータと接続する際に、自動呼出し装置（ACU: automatic call unit）またはモデムと設定するために必要な文字列が記述されています。詳細については、199 ページの「UUCP Dialers ファイル」を参照してください。

`Systems` `uucico` デーモンと `cu` プログラムがリモート・コンピュータと接続するのに必要な情報が記述されています。たとえば、リモート・コンピュータの名前、リモート・コンピュータの接続デバイスの名前、コンピュータと通信する日時、電話番号、ログイン ID、およびパスワードなどが記述されています。詳細については、202 ページの「UUCP Systems ファイル」を参照してください。

`Dialcodes` `Systems` ファイルの電話番号フィールドに入力するダイヤル・コードの省略形が記述されています。詳細については、206 ページの「UUCP Dialcodes ファイル」を参照してください。

`Permissions` リモート・コンピュータのユーザが `uucp` または `uux` を使用してファイルを転送したりリモートでコマンドを実行する際に、そのリモート・ユーザに与えるアクセス権が定義されています。詳細については、207 ページの「UUCP Permissions ファイル」を参照してください。

`Poll` ローカル・ステーションによってポーリングするコンピュータと、いつポーリングするかが定義されています。詳細については、216 ページの「UUCP Poll ファイル」を参照してください。

`Sysfiles` `uucico` および `cu` が使用するファイル、たとえば `Systems`、`Devices`、`Dialers` などを指定します。詳細については、216 ページの「UUCP Sysfiles ファイル」を参照してください。

リンクの確立とファイル転送のプロセスには直接関係しないものの、サポート・データベースとして考えられるファイルがほかにもいくつかあります。たとえば、`Maxuuxqts`、`Maxuuscheds`、`remote.unknown` などがそうです。これらのファイルについては、217 ページの「その他の UUCP ファイル」を参照してください。

UUCP Devices ファイル

`Devices` ファイル（`/etc/uucp/Devices`）には、自動呼出し装置、直接リンク、ネットワーク接続など、リモート・コンピュータとのリンクを確立する際に使用するすべてのデバイスに関する情報が記述されています。

メモ：このファイルは、`Dialers`、`Systems`、および `Dialcodes` ファイルと相互に依存して動作します。これらのファイルに変更を加える場合は、これらの 4 つのファイルの関係について熟知しておく必要があります。1 つのファイルのエントリを変更すると、それに関連するファイルのエントリにも変更が必要になることがあります。

`Devices` ファイルのエントリは、次の形式で記述されています。

```
Type Line Line2 Class Dialer-Token-Pairs
```

モデム用のエントリは、次の形式で記述されています。

```
Name device null speed 212 x dialer
```

TCP/IP ネットワーク接続用のエントリは、次の形式で記述されています。

```
TCP - - Any TCP uucp
```

次に、Devices ファイルの各フィールドについて説明します。

UUCP Type フィールド

Type フィールドのキーワードは、Systems ファイルの第 3 フィールドのエントリと一致します。Type フィールドに指定できるキーワードは、**Direct**、**ACU**、またはステーション名のいずれかです。

Direct このキーワードは、ほかのコンピュータまたはスイッチ（cu 接続の場合のみ）との直接リンクを表します。

ACU このキーワードは、リモート・コンピュータとのリンクが自動呼出し装置（自動ダイヤル・モデム）を介して確立されていることを表します。

Sys-Name この値は、特定のコンピュータとの直接リンクを表します。Sys-Name には、コンピュータの名前を入力します。入力した値は、この Devices エントリに対応する回線が Systems ファイルで指定された特定のコンピュータで使用されることを表します。

このフィールド内のデバイスに対して使用するプロトコルは指定することができます。198 ページの「UUCP デバイス・プロトコル」を参照してください。

UUCP Line フィールド

Line フィールドには、`Devices` エントリに対応する回線（ポート）のデバイス名を入力します。たとえば、特定のエントリの自動ダイヤル・モデムを `/dev/ttyf5` 回線に接続する場合、このフィールドには *ttyf5* を入力します。

モデムを使う場合は、必ず `ttyf` デバイスを使用します。これらのデバイスはハードウェア・フロー制御をサポートしており、V.32 または V.32*bis* をサポートする全モデムで使用されます。

UUCP Line2 フィールド

Type フィールドにキーワード ACU が指定されていて ACU が 801 型ダイヤラである場合、Line2 フィールドには 801 ダイヤラのデバイス名を入力します。801 型 ACU にはモデム機能がないので、別のモデムが必要となります。このモデムは、Line フィールドに定義されている回線とは別の回線に接続します。つまり、1 方の回線をモデムに、もう 1 方の回線をダイヤラに使います。通常、非 801 型ダイヤラはこのような構成を使用しないため、Line2 を無視します。ただし、Line2 にはプレース・ホルダとしてハイフン（-）またはキーワード **`null`** が必要です。大半のモデムにはこのプレース・ホルダの入力が必要です。

UUCP Class フィールド

Devices ファイルの Class フィールドのキーワードは、Systems ファイルの第 4 フィールドのエントリと一致します。

```
Devices: ACU ttyf5 null D9600 212 x telebit
Systems: eagle Any ACU D9600 14155551212 login:nuucp password:Oakgrass
```

どの転送速度でも使用可能なデバイスに対しては、Class フィールドにキーワード **`Any`** を指定します。**Any** を指定すると、その回線は Systems ファイルのエントリが要求するどの転送速度にも対応します。このフィールドが Any で、Systems ファイルの Class フィールドも Any である場合、デフォルトの転送速度は 9,600 bps となります。Type フィールドにキーワード **`ACU`** または **`Direct`** を使用する場合、Class フィールドにはデバイスの速度だけを指定します。ただし、この速度の前に 1 文字付加し（たとえば *C9600*、*D9600*）、ダイヤラのクラス（Centrex や DimensionPBX）を区別することもできます。1 つの電話回線網は内部通信専用で、別の電話回線網は外部通信用というように、2 種類以上の電話回線網を備えた大企業では、ダイヤラ・クラスを指定する必要があります。この場合、どの回線が内部通信用で、どの回線が外部通信用かを区別します。

UUCP Dialer-Token-Pairs フィールド

このフィールドではダイヤラとトークンのペアを指定します。ダイヤラ部分は、自動ダイヤル・モデムの名前、または直接リンク・デバイスを表す「Direct」を指定します。指定できる DTP（Dialer-Token-Pair）フィールドの数に上限はありません。トークン部分はダイヤラ部分の直後に指定します。トークン部分を指定しないと、トークン部分は Systems ファイルの関連エントリから検索されます。

このフィールドの形式は次のとおりです。

```
dialer   token   dialer   token
```

DTP の最後のエントリは、接続デバイス（ダイヤラ）に応じて、ペアになっている場合とそうでない場合があります。多くの場合、最後のペアはダイヤラ部分だけで、トークン部分は `Systems` ファイルの Phone フィールドのエントリが適用されます。ダイヤラ部分に指定できるエントリは、`Dialers` ファイルで定義されています。

DTP フィールドには、エントリに対応するデバイスに応じて、いろいろな組合わせを指定することができます。

自動ダイヤル・モデムがコンピュータのポートに直接接続されている場合は、`Devices` ファイルの DTP フィールドはダイヤラ部分だけから構成されます。通常、これはモデムの名前になります。この名前で `Devices` ファイルのエントリと `Dialers` ファイルのエントリとを対応させます。したがって、このファイルの Dialer フィールドは `Dialers` ファイルの第 1 フィールドのエントリと一致させる必要があります。

```
Devices: ACU ttyf2 null 9600 212 x telebit
Dialers: telebit =&-% "" \r\p\r\c $ <K\T%%\r>\c ONLINE!
```

上の例では、`Devices` ファイルの DTP フィールドにはダイヤラ部分（*telebit*）しかありません。つまり、ダイヤラに渡すトークン（この例では電話番号）は、`Systems` ファイルの Phone フィールドに指定されているものが適用されます。エスケープ文字として \T が使われます。

特定のコンピュータとの直接リンクが確立されると、対応する DTP フィールドにはキーワード **`Direct`**が入ります。このキーワードは、*Direct* と *System-Name* のどちらのタイプにも使用します。

自動ダイヤル・モデムをスイッチに接続する場合、まずコンピュータがそのスイッチにアクセスします。その後、スイッチがモデムと接続します。2 つの *Dialer-Token-Pairs* が必要です。各ペアのダイヤラ部分（エントリの第5と第7フィールド）を*Dialers*ファイルのエントリと一致させます。

```
Devices: ACU ttyf2 null 9600 212 x t2500 telebit T25

Dialers: telebit "" "" \pr\ps\c est:\007 \E\D\e \007

Dialers: T25 =&-% "" \r\p\r\c $ <K\T%%\r>\c ONLINE!
```

最初のペアでは、*t2500* がダイヤラで *telebit* がトークンです。このトークンは Develcon スイッチに渡され、コンピュータにどのデバイス（telebit モデム）を接続するのかをスイッチに指示します。スイッチごとに設定が異なるので、このトークンはモデム・スイッチごとに一意になります。telebit モデムが接続されると、2 番目のペアがアクセスされます。このペアでは、*T25* がダイヤラであり、電話番号であるトークンは `Systems` ファイルから検索されます。205 ページの「UUCP Systems ファイルの Phone フィールド」を参照してください。

DTP フィールドには、次の 2 つのエスケープ文字を使えます。

\T　　`Dialcodes` ファイルで Phone（Token）フィールドを変換する必要があることを示します。

\D　　`Dialcodes` ファイルで Phone（Token）フィールドを変換する必要がないことを示します。Devices エントリの最後にエスケープ文字が指定されていない場合は、デフォルトで \D が適用されます。

UUCP デバイス・プロトコル

デバイスごとに使用するプロトコルを定義できますが、ほとんどの場合、その必要はありません。デフォルトの値を使用するか、または呼出している特定のステーションによってプロトコルが定義されているからです。プロトコルの定義については、204 ページの「UUCP Systems ファイルの Type フィールド」を参照してください。プロトコルを指定する場合は、Type、Protocol の形式で定義します。次は、使用可能なプロトコルです。

g　　このプロトコルは **e** よりも低速ですが、信頼性の高いプロトコルです。ノイズの多い電話回線での送信に適しています。これはデフォルトのプロトコルです。

e　　このプロトコルは **g** よりも高速ですが、TCP/IP ネットワーク接続などエラーのないネットワーク上での送信を前提としています。

t **e** と同様に、このプロトコルは TCP/IP ネットワーク接続のようなエラーがない環境で使用します。このプロトコルは、BSD UNIX オペレーティング・システムが動作するシステムで使用します。

UUCP Dialers ファイル

`Dialers` ファイル（`/etc/uucp/Dialers`）は、実際にデータの送信を開始する前に必要な対話を指定します。通常、この対話は送信または受信待ちの一連の ASCII 文字列であり、ASCII ダイヤラ（自動ダイヤル・モデムなど）で電話番号をダイヤルするのに使われます。

前述のように、`Devices` ファイルの第 5 フィールドは `Dialers` ファイルへのインデックスか、または特殊なタイプのダイヤラです。*Devices* ファイルの第 5 フィールドと各 `Dialers` ファイルの第 1 フィールドは一致させるようにします。また、`Devices` ファイルの 7 番目のフィールド以降の奇数フィールドは、`Dialers` ファイルへのインデックスとして使用されます。両ファイルのエントリが一致していると、`Dialers` ファイルのエントリが解釈され、ダイヤラ折衝が行われます。

`Dialers` ファイルのエントリは、次の形式で記述されています。

```
dialer substitutions expect-send ...
```

`Dialer` フィールドは `Devices` ファイルの第 5 フィールドとそれ以降の奇数フィールドと一致します。

substitutions フィールドは変換文字列です。このフィールドの文字列はペアになっており、ペアの最初の文字がペアの 2 番目の文字に対応付けられています。通常、この方法によって、等号（=）文字とハイフン（-）文字がそれぞれ、発信音待ちと休止に変換されます。

expect-send フィールドは文字列です。

次に、`Dialers` ファイルについて詳しく説明します。

- 「UUCP Dialers ファイルのエスケープ文字」（200 ページ）
- 「UUCP Dialers ファイルの例」（201 ページ）

UUCP Dialers ファイルのエスケープ文字

次に、`Dialers` ファイルで使用できるエスケープ文字についてまとめます。表 8-2 は、UUCP で使用するエスケープ・シーケンスです。

表 8-2 UUCP のエスケープ・シーケンス

エスケープ・シーケンス	説明
\N	ヌル文字（ASCII NUL）を送信または受信待ちします。
\b	バックスペース文字を送信または受信待ちします。
\c	文字列の最後にある場合は、通常は送信される改行を送信しません。それ以外の場所にある場合は、無視されます。
\d	次の文字を送信または読取る前に 2 秒間待ちます。
\p	約 0.25 秒から 0.5 秒間待ちます。
\E	エコー・チェックを開始します。この時点から、文字を送信すると、処理を継続する前にその文字が送り返されるのを待ちます。
\e	エコー・チェックをオフにします。
\n	改行文字を送信します。
\r	改行を送信または受信待ちします。
\s	スペース文字を送信または受信待ちします。
\t	タブ文字を送信または受信待ちします。
\\	バックスラッシュ（\）文字を送信または受信待ちします。
EOT	EOT 改行を 2 回送信または受信待ちします。
BREAK	ブレーク文字を送信または受信待ちします。
\K	BREAK と同じです。
ddd	8 進値（*ddd*）で表される文字を送信または受信します。

UUCP Dialers ファイルの例

次に示すのは、Dialers ファイルのサンプル行です。

```
penril  =W-P    "" \d > Q\c : \d- > s\p9\c )-W\p\r\ds\p9\c-) y\c :
\E\TP > 9\c OK
```

Dialers ファイルの *penril* エントリは次のように実行されます。まず、引数としての電話番号が変換され、等号があれば **W**（wait for dial tone：発信音待ち）に、ハイフンがあれば **P**（pause：休止）に置換えられます。

次に、サンプル行の残りの部分のハンドシェイクを示します。

""	何も待ちません。つまり、次のステップに進みます。
\d	2 秒間待ちます。
>	右不等号（>）を待ちます。
Q\c	改行を挿入せずに、Q を送信します。
:	コロン（:）を待ちます。
\d-	2 秒間待ちます。
>	右不等号（>）を待ちます。
s\p9\c	s を送信し、0.5 秒間待ち、9 を送信し、改行は送信しません。
)-W\p\r\ds\p9\c-)	閉じかっこの [)] を待ちます。これが受信されない場合は、ハイフンで囲まれた文字列を次のように処理します。W を送信し、待ち、復帰を送信し、待ち、s を送信し、待ち、改行を挿入せずに 9 を送信し、閉じかっこを待ちます。
y\c	改行を挿入せずに y を送信します。
:	コロン（:）を待ちます。
\E\TP	エコー・チェックをオンにします。この時点から、文字を送信するたびに、同一文字が送り返されてくるのを待ちます。その後、電話番号を送信します。\T は、引数として渡された電話番号に Dialcodes ファイルによる変換を適用し、さらにこのエントリの第 2 フィールドで指定されたモデム機能の文字列変換を適用した結果を送信するものです。電話番号を送信した後、**P** を送信します。

9\c	改行を挿入せずに 9 を送信します。
OK	文字列 **OK** を待ちます。

UUCP Systems ファイル

`Systems` ファイル（`/etc/uucp/Systems`）には、`uucico` デーモンがリモート・コンピュータと通信リンクを確立するために必要な情報が記述されています。このファイルの各エントリは、ローカル・コンピュータを呼出せるコンピュータ、またはローカル・コンピュータから呼出せるコンピュータを表します。UUCP ソフトウェアはデフォルトで、このファイルに登録されていないコンピュータがローカル・コンピュータにログインできないように設定されています。`remote.unknown` ファイルについては、217 ページの「その他の UUCP ファイル」を参照してください。また、1 つのコンピュータに対して、複数のエントリを設定することもできます。この付加的なエントリは代替としての通信経路を表すものであり、順次試みられます。

`Sysfiles` ファイルを使用し、`Systems` ファイルとして使用できるファイルを複数定義することができます。詳細については、216 ページの「UUCP Sysfiles ファイル」を参照してください。

`Systems` ファイルのエントリは、次の形式で記述されています。

```
System-name Time Type Class Phone Login
```

次に、これらのフィールドについて説明します。

- 「UUCP Systems ファイルの System-Name フィールド」（203 ページ）
- 「UUCP Systems ファイルの Time フィールド」（203 ページ）
- 「UUCP Systems ファイルの Type フィールド」（204 ページ）
- 「UUCP Systems ファイルの Class フィールド」（204 ページ）
- 「UUCP Systems ファイルの Phone フィールド」（205 ページ）
- 「UUCP Systems ファイルの Login フィールド」（205 ページ）

UUCP Systems ファイルの System-Name フィールド

このフィールドは、リモート・コンピュータのノード名を表します。

UUCP Systems ファイルの Time フィールド

このフィールドは、リモート・コンピュータを呼出す曜日と時刻を示す文字列です。Time フィールドは、次の形式で記述されています。

day[*time*][;*retry*]

day 部分には、次に示すオプションを指定します。複数のオプションを組合わせて指定することもできます。

Su、Mo、Tu、We、Th、Fr、Sa
: これらの省略形は、日曜日、月曜日、火曜日、水曜日、木曜日、金曜日、土曜日の各曜日を表します。

Wk
: 月曜日から金曜日の平日を指定します。

Any
: 毎日呼出し可能です。

Never
: ステーションがリモート・コンピュータを呼出すことがないことを意味します。Time フィールドに **Never** が指定されていると、ローカル・コンピュータからリモート・コンピュータを呼出すことはありません。呼出しは、必ずリモート・コンピュータが行います。つまり、ローカル・コンピュータはリモート・コンピュータに関して受動モードにあります。パーミッションについては、207 ページの「UUCP Permissions ファイル」を参照してください。

次に示すのは、Time フィールドの例です。

```
Wk1700-0800,Sa,Su
```

この例では、月曜日から金曜日までの午後 5 時から午前 8 時までと、土曜日と日曜日の終日に呼出しを許可しています。この例では、電話料金の安い時間帯に呼出しを行うように設定しており、送信が緊急を要さないかぎり効果的なものです。

time 部分は、たとえば、0800-1230 のような時間帯にします。*time* 部分が指定されていないと、1 日中いつでも呼出しが許可されてしまいます。0000 をまたぐ時間帯も指定できます。たとえば、

0800-0600 は、午前 6 時から午前 8 時までを除くすべての時刻に呼出しを許可していることを意味します。

オプションのサブフィールド *retry* では、呼出しが失敗した後の再試行までの最短時間（分単位）を指定します。デフォルトの待ち時間は、最初の失敗後 5 分、2 度目の失敗後 10 分というように、約 24 時間まで延ばすことができます。retry サブフィールドが指定されている場合は、失敗の後、その待ち時間が毎回適用されます。サブフィールドの区切り文字はセミコロン（;）です。たとえば、「Any;9」は、常時呼出し可能、ただし、失敗した場合は再試行まで最低 9 分待つと解釈されます。

UUCP Systems ファイルの Type フィールド

このフィールドは、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のタイプを指定します。このフィールドのキーワードは、Devices ファイルの第 1 フィールドと一致します。

```
Systems: eagle Any ACU,g D1200 3251 login:nuucp password: Oakgrass
Devices: ACU ttym2 - D1200 penril
```

プロトコルを Type フィールドに追加すると、リモート・ステーションとの通信に使うプロトコルを定義することができます。上記の例は、プロトコル **g** をデバイス・タイプ **ACU** に接続する方法を示しています。直接接続している場合は、接続しているステーションの名前を使用します。198 ページの「UUCP デバイス・プロトコル」を参照してください。

UUCP Systems ファイルの Class フィールド

このフィールドは、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。これには、ダイヤラのクラスを区別するための文字と速度、たとえば *C1200*、*D1200* などを指定します。194 ページの「UUCP Devices ファイル」の Class フィールドに関する説明を参照してください。どの転送速度でも扱えるデバイスには、キーワード *Any* を指定します。このフィールドは、対応する Devices ファイルの Class フィールドと一致する必要があります。

```
Systems: eagle Any ACU D1200 NY3251 login:nuucp password: Oakgrass
Devices:  ACU ttym2 - D1200 penril
```

このフィールドの情報が必要ない場合は、プレース・ホルダとしてこのフィールドにハイフンを入力します。

UUCP Systems ファイルの Phone フィールド

このフィールドは、リモート・コンピュータの電話番号（トークン）を自動ダイヤラに提供します。電話番号はオプションの英字省略形と数字から構成されます。省略形を使用する場合は、`Dialcodes` ファイルにリストされている省略形を使用します。次に例を示します。

```
Systems: eagle Any ACU D1200 NY3251 login:nuucp password: Oakgrass
Dialcodes:  NY 9=1212555
```

この例では、等号（=）は ACU に対して 2 次発信音を待ってから残りの数字をダイヤルするように指示しています。この文字列内にハイフン（-）を指定すると、次の数字をダイヤルするまで 4 秒間休止します。

コンピュータがモデム・スイッチに接続されている場合は、そのスイッチに接続されているほかのコンピュータにもアクセスすることができます。それらのコンピュータに対しては、`Systems` ファイルの Phone フィールドに電話番号ではなくトークンが指定されます。このトークンはローカル・コンピュータが通信するコンピュータを指定するためにスイッチに渡されます。通常トークンは、ステーション名になります。対応する `Devices` ファイルの最後には **\D** が必要です。これは、このフィールドが `Dialcodes` ファイルによって変換されないようにするためです。

UUCP Systems ファイルの Login フィールド

このフィールドには、フィールドとサブフィールドからなるログイン情報が記述されています。このフィールドは、次の形式で記述されています。

expect send

send は *expect* 文字列を受信したときに送信する文字列です。expect フィールドは、次の形式でサブフィールドから構成されます。

expect`[`*-send-expect*`]...`

前回の expect の読取りに失敗すると send フィールドが送信され、send の後の expect が次に受信される文字列となります。たとえば、login-login では、UUCP はまず login を待ちます。UUCP は login を受信すると、次のフィールドに進みます。login が受信されないと何も送信せず、改行の後再び login を検索します。リモート・コンピュータから受信する文字が最初に何もない場合は、expect の第 1 フィールドに `""`（ヌル文字列）を挿入します。また、send 文字列を **\c** で終了しないと、すべての send フィールドは改行付きで送信されます。

次に示すのは、expect-send 文字列を使用する Systems ファイルのエントリ例です。

```
owl Any ACU 1200 NY6013 "" \r login:-BREAK-login: uucpx word: xyzzy
```

この例では、UUCP で復帰を送信し、文字列 login: を待ちます。この文字列を受信しない場合は、ブレーク文字を送信し、再び login: を待ちます。これに対し、login: を受信した場合は、ログイン名 **uucpx** を送信し、文字列 word:（「Password:」のこと）を待ちます。その後、パスワード **xyzzy** を送信します。

エスケープ・シーケンスがログイン操作で送信される文字列の一部である場合は、特定の動作を実行します。これらのエスケープ・シーケンスは Dialers ファイルで使用されるものと同じです。表 8-2 を参照してください。

UUCP Dialcodes ファイル

Dialcodes ファイル（/etc/uucp/Dialcodes）には、Systems ファイルの Phone フィールドで使用するダイヤル・コードの省略形が記述されています。Dialcodes ファイルのエントリは、次の形式で記述されます。

```
abb dial-seq
```

abb は、Systems ファイルの Phone フィールドで使用される省略形であり、*dial-seq* はその Systems ファイルのエントリがアクセスされたときにダイヤラに渡されるダイヤル・シーケンスです。

たとえば、次のエントリは、Systems ファイルの Phone フィールドが jt7867 などに設定されている場合に動作します。

```
jt 9=555-
```

jt7867 を含むエントリを処理する場合、DTP のトークン **\T** に対して、9=555-7867 がダイヤラに送信されます。

UUCP Permissions ファイル

Permissions ファイル（/etc/uucp/Permissions）は、ログイン、ファイルのアクセス、およびコマンドの実行に対してリモート・コンピュータに与えるパーミッションを指定します。オプションを指定して、リモート・コンピュータがファイルを要求したり、ローカル・コンピュータの待ち行列にあるファイルを受信するのを制限することもできます。また、リモート・サイトがローカル・コンピュータ上で実行できるコマンドを指定することもできます。

Systems ファイルからサンプルの、またはデフォルトの Permissions ファイルを作成する場合は、/etc/uucp/genperm プログラムを使います。

次では、Permissions ファイルについて詳しく説明します。

- 「UUCP Permissions ファイルの構成」（207 ページ）
- 「UUCP Permissions ファイルに関する留意事項」（208 ページ）
- 「UUCP Permissions ファイルのオプション」（208 ページ）

UUCP Permissions ファイルの構成

このファイルの各エントリは、複数の行からなる論理行です。バックスラッシュ（\) で終了している行は、次の行へと継続することを示します。論理行は、バックスラッシュ以外の文字で終了している行で終わります。大半の UUCP ファイルでは、このように行を継続することはできません。エントリはオプションから構成され、それぞれ空白で区切られます。各オプションは名前と値で構成され、次の形式になっています。

```
name=value
```

オプションの代入式には、空白を挿入できません。

コメントはシャープ記号（#）から改行文字までの行全体です。空白行は複数行あっても、無視されます。

Permissions ファイルのエントリには次の 2 つのタイプがあります。

LOGNAME　リモート・コンピュータがローカル・コンピュータにログイン（呼出し）したときに有効となるパーミッションを指定します。

MACHINE　ローカル・コンピュータがリモート・コンピュータにログイン（呼出し）したときに有効となるパーミッションを指定します。

LOGNAME エントリは LOGNAME オプションで始まり、MACHINE エントリは MACHINE オプションで始まります。

UUCP Permissions ファイルに関する留意事項

`Permissions` ファイルでリモート・コンピュータに与えるアクセス権を制限する場合は、次の点に注意します。

- リモート・コンピュータが UUCP で通信するためにログインするときのログイン ID は 1 つであり、LOGNAME エントリも 1 つです。
- 呼出されたサイトの名前が MACHINE エントリに存在しない場合は、次のようなデフォルトのパーミッション／制約条件が与えられます。
  - ローカルの送信要求と受信要求が実行できます。
  - 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することができます。
  - リモート・コンピュータから送信されたコマンドをローカル・コンピュータ上で実行する場合、そのコマンドはデフォルトのコマンドでなければなりません。通常は `rmail` です。

UUCP Permissions ファイルのオプション

ここでは、各オプションとその使い方を説明し、デフォルトの値を示します。

REQUEST　リモート・コンピュータがローカル・コンピュータを呼出してファイルの受信を要求した場合、これを許可または拒否することができます。**REQUEST** オプションは、リモート・コンピュータがローカル・コンピュータにファイルを転送するように要求できるかどうかを指定します。

yes を指定すると、リモート・コンピュータがローカル・コンピュータにファイル転送を要求できます。

```
REQUEST=yes
```

no を指定すると、リモート・コンピュータがローカル・コンピュータからファイルを受信することを要求できません。

```
REQUEST=no
```

デフォルト値は no であり、**REQUEST** オプションが指定されていない場合に適用されます。**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・コンピュータを呼出す場合）または MACHINE エントリ（ローカル・コンピュータがリモート・コンピュータを呼出す場合）のどちらにも指定できます。

セキュリティに関する注意：リモート・コンピュータがローカル・コンピュータを呼出した場合、そのコンピュータに一意なログインとパスワードがないかぎり、そのコンピュータを識別することはできません。

SENDFILES リモート・コンピュータがローカル・コンピュータを呼出してその作業を完了すると、ローカル・コンピュータがそのリモート・コンピュータに対して待ち行列に入っている作業を実行しようと試みます。**SENDFILES** オプションは、ローカル・コンピュータがリモート・コンピュータに対して待ち行列に入っている作業を送信するかどうかを指定します。

yes は、リモート・コンピュータに対して待ち行列に入っている作業を送信します。ただし、これはリモート・コンピュータが **LOGNAME** オプションで指定されている名前の 1 つでログインした場合にかぎります。

```
SENDFILES=yes
```

ローカル・コンピュータがリモート・コンピュータに対して受動モードになっている場合は、この文字列を指定する必要があります。

call は、ローカル・コンピュータがリモート・コンピュータを呼出した場合にだけ、ローカル・コンピュータの待ち行列にあるファイルを送信します。

```
SENDFILES=call
```

デフォルト値は call です。このオプションは、LOGNAME エントリがある場合に有効です。リモート・コンピュータを呼出す場合は、MACHINE エントリが適用されるからです。このオプションを MACHINE エントリに対して指定しても無効です。

READ および **WRITE**

この 2 つのオプションは、uucico が読込みまたは書込み可能なファイル・システムのディレクトリを指定します。**READ**と**WRITE**オプションは、MACHINE エントリと LOGNAME エントリのいずれでも使用できます。

READ と **WRITE** オプションのデフォルトは、uucppublic ディレクトリです。

```
READ=/var/spool/uucppublic
```

```
WRITE=/var/spool/uucppublic
```

この文字列は、"other" のパーミッションが与えられたローカル・ユーザがアクセスできるすべてのファイルへのアクセス権を与えます。

```
READ=/  WRITE=/
```

これはセキュリティ上の問題を起こす可能性があるので、必要な場合にだけ指定します。

次に示すエントリの値は、コロンで区切られたパス名のリストです。**READ** オプションはほかのユーザにファイルの読込み権を、**WRITE** オプションはファイルのディレクトリへの書込み権を与えます。この値のいずれかに、受信または送信するファイルのフルパス名をプレフィックスとして記述します。パブリック・ディレクトリのほかに、/usr/news ディレクトリにファイルを置くことを許可する場合には、**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ないため、**READ** または **WRITE** オプションを使用する場合は、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 だけが指定されている場合は、パブリック・ディレクトリにファイルを置くことは許可されません。

リモート・ステーションからの読取り／書込みを許可するディレクトリを決定する場合には注意が必要です。たとえば、リモート・コンピュータが /etc/passwd ファイルを上書きすることがないように、/etc には書込み権を与えないようにします。

NOREAD および **NOWRITE**

NOREAD と **NOWRITE** オプションは、**READ** と **WRITE** オプションまたはデフォルトの値に対する例外を指定します。次に示す文字列は、特定のリモート・コンピュータに対して、/etc ディレクトリおよびそのサブディレクトリ（これ

らはプレフィックスです)にあるファイルを除くすべてのファイルの書込みと、デフォルトのディレクトリ /var/spool/uucppublic に対する書込み権だけを与えます。

```
READ=/ NOR EAD=/etc WRITE=/var/spool/uucppublic
```

NOWRITE は **NOREAD** と同じように動作します。**NOREAD** と **NOWRITE** は LOGNAME と MACHINE エントリのどちらにも使用することができます。

CALLBACK

CALLBACK オプションは LOGNAME エントリで使用し、呼出し側のステーションを呼出された側がコールバックするまで、一切のトランザクションを処理しないようにします。**CALLBACK** を指定する理由は 2 つあります。1 つはセキュリティ上の理由で、ステーションにコールバックすれば、それが呼出し側のステーションであることが確認できるからです。もう 1 つは、長時間のデータ伝送を行う場合に、長時間の呼出しに対して料金を請求するステーションを選択できるからです。

次の例は、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てからファイル転送を行うように指定しています。

```
CALLBACK=yes
```

CALLBACK オプションのデフォルトは no です。

```
CALLBACK=no
```

CALLBACK オプションを指定することはほとんどありません。2 つのサイトが相互にこのオプションを設定した場合は、対話を開始することはできません。

COMMANDS

COMMANDS オプションはステーションのセキュリティを損なう可能性があるので、このコマンドの使用に際しては細心の注意を払います。

uux プログラムはリモート・ステーションに対する実行要求を作成し、それを待ち行列に入れ、リモート・コンピュータに転送します。ファイルとコマンドはターゲット・ステーションに送信され、リモートで実行されます。

COMMANDS オプションを MACHINE エントリで使用すると、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できます。**COMMANDS** は LOGNAME エントリでは使用しません。MACHINE エントリの **COMMANDS** は、ローカルとリモートのどちら側から呼出すかにかかわらず、コマンドの実行権を決定します。

次の例は、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デフォルトのコマンドを示しています。

```
COMMANDS=rmail
```

MACHINE エントリでコマンド文字列を指定すると、デフォルトのコマンドは無効になります。たとえば、次の例では、MACHINE のエントリが **COMMAND** のデフォルトを無効にし、リモート・コンピュータの *eagle*、*owl*、および *hawk* がローカル・コンピュータ上で `rmail` と `rnews` を実行できるようにしています。

```
MACHINE=eagle:owl:hawk REQUEST=yes
COMMANDS=rmail:/usr/bin/rnews
READ=/  WRITE=/
```

この例で指定している名前のほかに、コマンドのフルパス名も指定できます。たとえば、次の例はコマンド `rmail` がデフォルトのパスを使用するように指定しています。

```
COMMANDS=rmail:/usr/bin/rnews:/usr/local/lp
```

ローカル・コンピュータのデフォルトのパスは `/bin`、`/usr/sbin`、`/usr/bsd`、および `/usr/bin` です。リモート・コンピュータが実行するコマンドに対して `rnews` または `/usr/bin/rnews` を指定すると、デフォルトのパスにかかわらず `/usr/bin/rnews` が実行されます。同様に、`lp` コマンドとしては `/usr/local/lp` が実行されます。

メモ：COMMANDS のリストに ALL を指定すると、指定されているリモート・コンピュータがすべてのコマンドを実行できるようになります。つまり、リモート・コンピュータに、ローカル・コンピュータへの完全なアクセス権を与えることになるため十分に注意してください。この値は、通常のユーザが許可されているレベル以上のアクセス権を与えることになります。

次の文字列は、さらに高レベルのアクセス権を与えます。

```
COMMANDS=/usr/bin/rnews:ALL:/usr/local/lp
```

この文字列には注意すべき点が 2 つあります。1 つは文字列のどこにでも ALL を指定することができるという点、もう 1 つは、要求されたコマンドで `rnews` または `lp` のフルパス名が指定されていない場合は、デフォルトの代わりに上記の `rnews` および `lp` に指定されているパス名が使用される点です。

COMMANDS オプションに cat や uucp などの潜在的に危険なコマンドが指定されている場合は、必ず **VALIDATE** オプションに **COMMANDS** オプションを組合わせて使用します。ファイルの読込みや書込み行うコマンドが UUCP リモート実行デーモン（uuxqt）によって実行される場合、それらのコマンドはローカル・ステーションのセキュリティにとって潜在的に危険な存在となります。

VALIDATE **VALIDATE** オプションは、コンピュータのセキュリティにとって潜在的に危険なコマンドを指定するときに、**COMMANDS** オプションと組合わせて使用します。**VALIDATE** を使用すると、呼出し側をある程度確認することができます。**VALIDATE** オプションを使用するには、特権付きのコンピュータが UUCP トランザクションを行うために一意なログインとパスワードを持っている必要があります。呼出し側を確認するときの重要な点は、エントリに対応するログインとパスワードが保護されていることです。部外者がこの情報を入手した場合には、その **VALIDATE** オプションも安全なものではありません。**VALIDATE** は単に **COMMANDS** オプションのセキュリティ・レベルを向上させるものにすぎませんが、コマンド・アクセスを与える方法としては ALL よりも安全です。

リモート・システムに UUCP トランザクションの特権ログインとパスワードを与える際は、慎重に行います。このような特権をほかのシステムに与えることは、そのコンピュータを使用する全ユーザにローカル・コンピュータの通常のログインとパスワードを教えることになります。そのため、リモート・サイトのユーザ全員を信用できない限り、そのシステムに特権ログインとパスワードを提供しないようにします。

LOGNAME **LOGNAME** オプションは、ローカル・コンピュータにログインしようとするリモート・ステーションが、ログインする特権を持っているかどうかを確認します。次の **LOGNAME** エントリは、リモート・コンピュータが *eagle*、*owl* または *hawk* のいずれかであり、ローカル・コンピュータにログインする場合、ログイン *uucpfriend* を使用する必要があることを示しています。

```
LOGNAME=uucpfriend VALIDATE=eagle:owl:hawk
```

ただし、部外者が uucpfriend ログインとパスワードを入手した場合は、簡単に不法侵入される可能性があります。

このことは、**MACHINE** エントリにしか現れない **COMMANDS** オプションと関係があるのでしょうか。**COMMANDS** オプションは **MACHINE** エントリと **COMMANDS** オプションを特権ログインと関連する **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間は、デー

モンが実行されていないからです。実際には、このリンクは実行要求を送ったコンピュータをまったく認識できない非同期プロセスです。このため、実行ファイルがどこから送られてきたのかをローカル・コンピュータが知る方法が必要です。

各リモート・コンピュータは、ローカル・コンピュータ上に独自のスプール・ディレクトリを持っています。これらのディレクトリに書込みができるのは、UUCP プログラムだけです。リモート・コンピュータからの実行ファイルは、ローカル・コンピュータに送られた後にそのスプール・ディレクトリに置かれます。`uuxqt` デーモンを実行すると、このデーモンはスプール・ディレクトリ名を使用して `Permissions` ファイルの MACHINE エントリを見つけ、COMMANDS リストを入手します。また、コンピュータ名が `Permissions` ファイルにない場合は、デフォルトのリストを使用します。

次に、**MACHINE** エントリと **LOGNAME** エントリの関係を示します。

```
MACHINE=eagle:owl:hawk REQUEST=yes  \
COMMANDS=rmail:/usr/bin/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・ユーザがリモート・メールと `/usr/bin/rnews` を実行できることを意味しています。

最初のエントリでは、リストにあるコンピュータの 1 つを呼出す際に、実際に *eagle*、*owl*、または *hawk* を呼出すことを前提としています。したがって、*eagle*、*owl*、または *hawk* のいずれかのスプール・ディレクトリに格納されるファイルは、この 3 つのいずれかのコンピュータによって格納されます。この 3 つのコンピュータのうちの 1 つであるリモート・コンピュータがログインすると、その実行ファイルも特権スプール・ディレクトリに格納されます。このため、そのコンピュータが特権ログイン `uucpz` を持っているかどうかを確認する必要があります。

その他のシステムのための **MACHINE** エントリ

ローカル・コンピュータが呼出すリモート・コンピュータのうち、特定の **MACHINE** エントリに記述されていないコンピュータに対しては、別のオプショ

ン値を指定することができます。これは、ログインしてくるコンピュータが多数存在する場合に、設定コマンドを時々変更するときに必要になります。このようなエントリには、コンピュータ名として **OTHER** を使用します。

```
MACHINE=OTHER \
   COMMANDS=rmail:rnews:/usr/bin/Photo:/usr/bin/xp
```

ほかの **MACHINE** エントリに記述されていないコンピュータに対しても、**MACHINE** エントリで使用できるその他のすべてのオプションを設定することができます。

MACHINE エントリと **LOGNAME** エントリの組合わせ

MACHINE エントリと **LOGNAME** エントリの対応するオプションの値が同じ場合は、それを 1 つのエントリにマージすることができます。たとえば、次に示す 2 つのエントリは、**REQUEST**、**READ**、および **WRITE** オプションが同じです。

```
MACHINE=eagle:owl:hawk REQUEST=yes \
   READ=/  WRITE=/
LOGNAME=uucpz REQUEST=yes SENDFILES=yes \
   READ=/  WRITE=/
```

この 2 つのエントリは次のようにマージできます。

```
MACHINE=eagle:owl:hawk REQUEST=yes \
LOGNAME=uucpz SENDFILES=yes \
READ=/  WRITE=/
```

MYNAME

MYNAME オプションは、ローカル・コンピュータがリモート・コンピュータに対して自らを識別する場合に、そのローカル・コンピュータの名前を無効にします。このオプションは、コンピュータを交換した場合や名前を変更した場合、または、その周辺のコンピュータが古い名前で古いトラフィックを処理しなければならない場合に使います。

UUCP Poll ファイル

Poll ファイル（/etc/uucp/Poll）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記述されています。Poll ファイルの各エントリは、呼出すリモート・コンピュータの名前、<Tab> 文字（スペースは挿入しないでください）、そのコンピュータを呼出す時刻の順で記述します。Poll ファイルのエントリは次の形式で指定します。

```
sys-name hour ...
```

たとえば、次のエントリでは、4 時間おきにコンピュータ *eagle* のポーリングを行います。

```
eagle  0 4 8 12 16 20
```

uudemon.poll スクリプトは、実際にはポーリングを実行せず、uudemon.hour で起動するスケジューラが確認するスプール・ディレクトリのポーリング作業用ファイル（C.file）を設定します。

UUCP Sysfiles ファイル

/etc/uucp/Sysfilesファイルは、uucpとcuで別のファイルをSystemsファイル、Devicesファイル、および Dialers ファイルとして使用できるようにします。これは、次の場合に有用です。

- 別の Systems ファイルを使用し、uucp サービスを要求するのとは別の電話番号でログイン・サービスを要求する場合。
- cu および uucp に対して異なるハンドシェイクを持つ Dialers ファイルを使用する場合。
- 複数の Systems ファイル、Dialers ファイル、および Devices ファイルを使用する場合。特に Systems ファイルは大きいファイルであり、いくつかのファイルに分割した方が便利です。

Sysfiles ファイルは、次の形式で記述されています。

```
service=w systems=x:x dialers=y:y devices=z:z
```

w パラメータには、uucico または cu のいずれか一方、またはその両方をコロンで区切って指定します。*x* は Systems ファイルとして使用されるファイルです。複数のファイルを指定する場合は、各ファイル名をコロンで区切ります。各ファイルは、記述されている順序で読込まれま

す。*y* は Dialers ファイルとして使用されるファイル、*z* は Devices ファイルとして使用されるファイルです。それぞれ、複数指定することができます。各ファイル名は、フルパスが指示されないかぎり、/etc/uucp ディレクトリからの相対パス名とみなされます。エントリが次の行に渡る場合は、バックスラッシュとエンター・キー（\-**<Enter>**）を使用します。

次に示すのは、通常の Systems ファイルのほかにローカルな Systems ファイルを使用している例です。

```
service=uucico:cu systems=Systems:Local_Systems
```

この行が /etc/uucp/Sysfiles に存在する場合は、uucico と cu は最初に /etc/uucp/Systems を参照します。uucico と cu が呼出そうとしているステーションのエントリが、そのファイルに存在しない場合、またはそのエントリが正しくない場合には、/etc/uucp/Local_Systems を参照します。

別の Systems ファイルが uucico と cu に対して指定されると、ローカル・ステーションはそれぞれのステーションに対応する 2 つのリストを保持することになります。uucico のリストは uuname コマンド、cu のリストは uuname -c コマンドでそれぞれ出力できます。

その他の UUCP ファイル

これまで説明してきたファイルのほかに、ネットワーキングの基本的な動作に影響する 3 つのファイルがあります。通常、ネットワークはデフォルトの値で適切に動作するので、それを変更する必要はありません。デフォルトの値を変更したい場合は、標準の IRIX テキスト・エディタ（ed、vi、または jot）で編集します。

Maxuuxqts　このファイルは、一度に実行できる uuxqt プログラムの最大数を定義します。デフォルト値は 2 です。

Maxuuscheds　このファイルは、一度に実行できる uusched プログラムの最大数を定義します。デフォルト値は 2 です。

unknown　このファイルは、どの Systems ファイルにも存在しないステーションが対話を開始したときに実行されるプログラムです。このプログラムは対話の試みを記録し、接続を拒否します。このファイルが実行されないようにパーミッションを変更すると（chmod 000 unknown）、ステーションはすべての対話要求を受付けるようになります。

UUCP 管理ファイル

UUCP 管理ファイルはスプール・ディレクトリに作成され、デバイスをロックし、一時データを格納し、またはリモートの伝送や実行に関する情報を保持します。

TM（一時データ・ファイル）

一時データ・ファイルは、別のコンピュータからファイルが受信されたときに UUCP プロセスによりスプール・ディレクトリ（たとえば、`/var/spool/uucp/X`）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・コンピュータの名前です。一時データ・ファイルの名前は、次の形式で記述されます。

`TM.`*pid.ddd*

`pid` はプロセス ID であり、*ddd* は 0 で始まる 3 桁の数字です。

ファイル全体が受信されると、`TM.pid.ddd` ファイルは、転送を行った `C.sysnxxxx` ファイル（C.（作業ファイル）を参照）に指定されているパス名に移動します。処理が異常終了した場合は、`TM.pid.ddd` ファイルは X ディレクトリに残ります。このファイルは `uucleanup` で自動的に削除することができます。

LCK（ロック・ファイル）

ロック・ファイルは、使用しているデバイスごとに `/var/spool/locks` ディレクトリに作成されます。ロック・ファイルは、対話が重複したり、同じ呼出し装置を重複して使用したりするのを防止します。ロック・ファイルの名前は、次の形式で記述されます。

`LCK..`*str*

str はデバイスまたはコンピュータの名前です。ロック・ファイルは、コンピュータ・クラッシュなどにより通信リンクが突然はずれた場合にスプール・ディレクトリに残ります。親プロセスが処理不能になると、ロック・ファイルは無視（削除）されます。各ロック・ファイルには、ロックを作成したプロセスのプロセス ID が記述されます。

C.（作業ファイル）

作業ファイルは、リモート・コンピュータの作業（ファイル転送やリモート・コマンド実行）が待ち行列に入れられた場合にスプール・ディレクトリに作成されます。作業ファイルの名前は、次の形式で記述されます。

`C.`*sysnxxxx*

sys はリモート・コンピュータの名前、*n* は作業の優先順位を表す ASCII 文字、*xxxx* は UUCP によって割当てられる 4 桁のジョブ番号です。作業ファイルには、次の情報が記述されています。

- 送信されるファイル、または要求されているファイルのフルパス名
- 送信先、ユーザ、ファイルのフルパス名
- ユーザ・ログイン名
- オプションのリスト
- スプール・ディレクトリにある対応するデータ・ファイルの名前。`uucp -c` または `uuto -p` オプションが指定されている場合は、仮の名前 (**D.0**) が使用されます。
- ソース・ファイルのモード・ビット
- 転送の完了を通知するリモート・ユーザのログイン名

D.（データ・ファイル）

データ・ファイルは、コマンド行でソース・ファイルをスプール・ディレクトリにコピーするように指定した場合に作成されます。データ・ファイルの名前は、次の形式で記述されます。

`D.`*systmxxxxyyy*

systm はリモート・コンピュータ名の最初の 5 文字、*xxxx* は UUCP によって割当てられる 4 桁のジョブ番号です。この 4 桁のジョブ番号の後ろに通し番号 *yyy* が続くこともあります。これは、作業（`C.`）ファイルに対して `D.` ファイルが複数作成された場合に追加されます。

X.（実行ファイル）

実行ファイルは、リモート・コマンドが実行される前にスプール・ディレクトリに作成されます。実行ファイルの名前は、次の形式で記述されます。

`X.`*sysnxxxx*

sys はリモート・コンピュータの名前、*n* は作業の優先順位を表す文字、*xxxx* は UUCP によって割当てられる 4 桁のジョブ番号です。実行ファイルには、次の情報が記述されています。

- 送信を要求した側のログイン名とコンピュータ名
- 実行に必要なファイルの名前
- コマンドの標準入力として使用する文字列
- コマンドからの標準出力を受信するファイルの名前、およびファイルが常駐するコンピュータのホスト名
- コマンド文字列
- コマンド実行の状態を返すオプション行

UUCP の設定

UUCP を設定するには、次の操作を行います。

1. リモート・ステーションおよびローカル・ステーションを決定します。220 ページの「リモート・ステーションとローカル・ステーションの決定」を参照してください。
2. 物理的な接続を行います。221 ページの「物理的な接続」を参照してください。
3. ローカル・ステーション（呼出し側）を設定します。221 ページの「ローカル・ステーションでの UUCP の設定」を参照してください。
4. リモート・ステーション（呼出される側）を設定します。225 ページの「リモート・ステーションでの UUCP の設定」を参照してください。
5. UUCP 接続をテストします。228 ページの「UUCP 接続のテスト」を参照してください。

TCP/IP ネットワーク上でも UUCP を使用できます。230 ページの「TCP/IP ネットワークでの UUCP の設定」を参照してください。

リモート・ステーションとローカル・ステーションの決定

通常、ローカル・ステーションが UUCP 接続の動作を開始します。リモート・ステーションは、UUCP 接続要求に応答するステーションです。ただし、`uugetty`（回線を双方向で使用可能にするプログラム）を使用する場合、ローカル・ステーションとリモート・ステーションを区別できるのはステーション名だけになります。

物理的な接続

UUCP は、TCP/IP を使用したローカル・エリア・ネットワーク、直接リンク、または電話回線などの物理的な接続をサポートしています。次の例では、直接リンクを使用しています。電話回線やローカル・エリア・ネットワーク上で UUCP を実行する手順も直接リンクと同じですが、設定ファイルを部分的に編集する必要があります。

直接リンクは 2 つのデータ端末装置（DTE: Data Terminal Equipment）間を接続します。この 2 つの装置は、データ通信装置（DCE: Data Communication Equipment）と通信しているかのように動作させます。これには、ヌル・モデムを使用します。

表 8-3 に最小のピン配置を示します。

表 8-3 3 線式ヌル・モデムのピン配置

IRIS A	IRIS B
2 送信データ	3 受信データ
3 受信データ	2 送信データ
7 通信用アース	7 通信用アース

ヌル・モデム・ケーブルをローカル・ワークステーションとリモート・ワークステーションのシリアル・ポート 2（*ttyf2*）に接続します。

メモ：ケーブルとモデムについては、『IRIX Admin: Peripheral Devices』の第 1 章「端末とモデム」を参照してください。

ローカル・ステーションでの UUCP の設定

ここではリモート・ステーション名を *us*、ローカル・ステーション名を *japan* とします。ローカル・ステーションを設定するには、次の操作を行います。

1. 標準システム・ファイルを更新します。/etc/passwd に関しては、222 ページの「UUCP のためのローカル・ステーションでの /etc/passwd の更新」を、/etc/group に関しては、223 ページの「UUCP のためのローカル・ステーションでの /etc/group の更新」を、また /etc/inittab に関しては、223 ページの「UUCP のためのローカル・ステーションでの /etc/inittab の更新」をそれぞれ参照してください。
2. UUCP設定ファイルを変更します。/etc/uucp/Systemsに関しては、223ページの「UUCP のためのローカル・ステーションでの /etc/uucp/Systems の変更」を、/etc/uucp/Devices に関しては、224 ページの「UUCP のためのローカル・ステーションでの /etc/uucp/Devices の変更」を、/etc/uucp/Dialers に関しては、224 ページの「UUCP のためのローカル・ステーションでの /etc/uucp/Dialers の変更」 を、また /etc/uucp/Permissions に関しては、225 ページの「UCUP のためのローカル・ステーションでの /etc/uucp/Permissions の変更」 を参照してください。

UUCP のためのローカル・ステーションでの /etc/passwd の更新

適切なセキュリティとアクセスを確立するには、uucp と nuucp に正しいユーザ・エントリを指定します。passwd ファイルの uucp エントリは UUCP 関連ファイルの所有者を指定し、nuucp エントリはリモートからの UUCP アクセスに使用されます。passwd ファイルにこの両方のエントリがあり、次のように記述されていることを確認します。uucp と nuucp のエントリの内容が異なる場合は、次と同じになるように編集します。

```
uucp:*:3:5:UUCP Owner:/usr/lib/uucp:/bin/csh

nuucp::10:10:Remote UUCP
User:/var/spool/uucppublic:/usr/lib/uucp/uucico
```

上の例では、nuucp の passwd エントリが書式上の都合で 2 行に分かれていますが、実際のファイルでは 1 行で記述されます。

新たにインストールしたステーションでは、uucp と nuucp にパスワードが設定されていません。uucp としてログインするユーザはいないので、uucp のパスワード・フィールドには <*> を挿入します。nuucp には、Systems ファイルの nuucp に指定するパスワードと同じものを指定します。202 ページの「UUCP Systems ファイル」を参照してください。たとえば、nuucp にはパスワード "secret" を指定します。

```
New password: secret
Re-enter new password: secret
```

UUCP のためのローカル・ステーションでの /etc/group の更新

uucp と nuucp の両方に対して有効なエントリがこのファイルにあり、次のように記述されていることを確認します。記述されている内容が異なる場合は、次のように編集します。

```
uucp::5:uucp
nuucp::10:nuucp
```

UUCP のためのローカル・ステーションでの /etc/inittab の更新

次に示すのは、ローカル・ステーションに対するエントリです。この例では、ポート 2 で呼出しは行いますが、受信は行いません。/etc/inittab を編集し、「t2」のエントリを次のように記述します。

```
t2:23:off:/usr/lib/uucp/uugetty -Nt 60 ttyf2 co_9600 # port 2
```

uugetty コマンドについては、uugetty(1M) マン・ページを参照してください。/etc/inittab に変更を加えた場合は、telinit q コマンドでこれを再度 init に読込ませます。それには、次のように入力します。

```
/etc/telinit q
```

UUCP のためのローカル・ステーションでの /etc/uucp/Systems の変更

Systems ファイルには、ローカル・ステーションが認識するステーションの情報が記述されています。ファイルの最後に次の行を追加します。

```
us Any systemx 9600 unused ogin:--ogin: nuucp ssword: \ secret
```

メモ：Systems ファイルは読込み専用です。このため、vi で編集する場合は **:wq!** オプションを指定し、変更内容を強制的に書込みます。

最初のフィールドには、リモート・ステーションを呼出すステーション名を指定します。2 番目のフィールドは、指定したステーションが常時呼出しを行えることを示します。3 番目のフィールドは、uucp に使用するデバイス名（**systemx**）を知らせます。このフィールドは、/etc/uucp/Devices ファイルの最初のフィールドにあるエントリの 1 つと一致していなければなりません。4 番目のフィールドには転送速度（9600）を指定します。5 番目のフィールドは、通常は電話番号に使い、直接リンクには使用しません（unused）。残りのフィールドはログイン

操作を処理します。これはローカル・ステーションとリモート・ステーションの間で折衝されるチャット・スクリプトです。このチャット・スクリプトは、uucp 接続を正しく動作させるために重要なものです。

UUCP のためのローカル・ステーションでの /etc/uucp/Devices の変更

Devices ファイルには、2 つのステーション間の物理的な接続に関する情報が記述されています。*systemx* のデバイスのエントリからポンド記号を削除します。

```
# ---A direct connection to a system
systemx ttyf2 - Any direct
```

メモ：別のポート上にステーションに対する直接リンクがある場合は、*systemx* デバイス・エントリをコピーし、ポート番号を変更します。

Devices ファイルの最初のフィールドでは、デバイスを Systems ファイルの 3 番目のフィールドのエントリにリンクします。2 番目のフィールドは、アクセスするポートを uucp に知らせます。3 番目のフィールドは ACU（Automatic Call Unit）を指定します。直接リンクの場合は、プレース・ホルダとして 3 番目のフィールドにダッシュ（-）を指定します。4 番目のフィールドでは転送速度を指定します。「Any」を指定すると、/etc/inittab ファイルが特定デバイスに合わせて速度を決定します。5 番目のフィールドはダイヤラ名です。これは、/etc/uucp/Dialers ファイルにある有効なエントリでなければなりません。

UUCP のためのローカル・ステーションでの /etc/uucp/Dialers の変更

このファイルには uucp デバイス用のチャット・スクリプトが記述されています。直接リンクの場合、チャット・スクリプトは Systems ファイルから検索されます。ただし、直接リンクの場合でも有効なダイヤラ・エントリが必要です。Dialers ファイルに「direct」ダイヤラの有効なエントリがあることを確認します。それには、次のように入力します。

```
grep direct /etc/uucp/Dialers
```

システムは次のように応答します。

```
direct
#   The following entry is for use with direct connections
uudirect ""   "" \r\d in:--in:
```

UCUP のためのローカル・ステーションでの /etc/uucp/Permissions の変更

Permissions ファイルは、リモートのユーザとステーションに関するリモートの uucp アクセスを制御します。オプションについては、207 ページの「UUCP Permissions ファイル」を参照してください。ここでは、Permissions ファイルを次のように記述します。

```
#dent"@(#)uucp:Permissions2.2"
# This entry for public login.
# It provides the default permissions.
# See the Basic Networking Utilities Guide for more information.
LOGNAME=nuucp MACHINE=us READ=/var/spool/uucp/uucppublic \
WRITE=/var/spool/uucppublic REQUEST=yes SENDFILES=yes \
COMMANDS=rmail
```

メモ：上の例ではエントリが複数行に渡って記述されていますが、1 行として解釈されます。

この例では、ユーザ nuucp がリモート・ステーション *us* からログインできるように指定しています。*us* の nuucp ユーザは、/var/spool/uucp/uucppublic ディレクトリにあるどのファイルでも読込むことができ、パブリック・ディレクトリ /var/spool/uucppublic に書込むこともできます。*us* のユーザはファイル転送を要求することができ、*japan* のユーザはファイルを送信することができます。

リモート・ステーションでの UUCP の設定

ここでは、リモート・ステーション名を *japan*、ローカル・ステーション名を *us* とします。リモート・ステーションを設定するには、次の操作を行います。

1. 標準システム・ファイルを更新します。/etc/passwd に関しては、226 ページの「UUCP のためのリモート・ステーションでの /etc/passwd の更新」 を、/etc/group に関しては、226 ページの「UUCP のためのリモート・ステーションでの /etc/group の更新」を、また /etc/inittab に関しては、226 ページの「UUCP のためのリモート・ステーションでの /etc/inittab の更新」 をそれぞれ参照してください。
2. UUCP 設定ファイルを変更します。/etc/uucp/Systems に関しては、227 ページの「UUCP のためのリモート・ステーションでの /etc/uucp/Systems の変更」 を、また /etc/uucp/Permissions に関しては、227 ページの「UUCP のためのリモート・ステーションでの /etc/uucp/Permissions の変更」 を参照してください。

UUCP のためのリモート・ステーションでの /etc/passwd の更新

適切なセキュリティとアクセスを確立するには、uucp と nuucp に正しいユーザ・エントリを指定します。passwd ファイルの uucp エントリは UUCP 関連ファイルの所有者を指定し、nuucp エントリはリモートからの UUCP アクセスに使用されます。passwd ファイルにこの両方のエントリがあり、次のように記述されていることを確認します。uucp と nuucp のエントリの内容が異なる場合は、次と同じになるように編集します。

```
uucp:*:3:5:UUCP Owner:/usr/lib/uucp:/bin/csh
nuucp::10:10:Remote UUCP User:/var/spool/uucppublic:/usr/lib/uucp/uucico
```

新たにインストールしたステーションでは、uucp と nuucp にパスワードが設定されていません。uucp としてログインするユーザはいないので、uucp のパスワード・フィールドには <*> を挿入します。nuucp には、Systems ファイルの nuucp に指定するパスワードと同じものを指定します。202 ページの「UUCP Systems ファイル」を参照してください。たとえば、nuucp にはパスワード "secret" を指定します。

```
passwd nuucp
New password: secret
Re-enter new password: secret
```

UUCP のためのリモート・ステーションでの /etc/group の更新

uucp と nuucp に対して有効なエントリがこのファイルにあり、次のように記述されていることを確認します。記述されている内容が異なる場合は、次のように変更します。

```
uucp::5:uucp
nuucp::10:nuucp
```

UUCP のためのリモート・ステーションでの /etc/inittab の更新

次に示すのは、ローカル・ステーションに対するエントリです。この例では、ポート 2 で呼出しは行いますが、受信は行いません。/etc/inittab を編集し、"t2" のエントリを次のように記述します。

```
t2:23:respawn:/usr/lib/uucp/uugetty -Nt 60 ttyf2 co_9600#pt 2
```

uugetty コマンドについては、uugetty(1M) マン・ページを参照してください。/etc/inittab に変更を加えた場合は telinit *q* コマンドでそれを再度 init に読込ませます。それには、次のように入力します。

```
/etc/telinit q
```

UUCP のためのリモート・ステーションでの /etc/uucp/Systems の変更

Systems ファイルには、リモート・ステーションが認識するステーションの情報が記述されています。Systems ファイルの最後に次の行を追加します。

```
japan Never
```

メモ：Systems ファイルは読込み専用です。このため、変更内容を強制的に書込む必要があります。*jot(1G)* または *vi(1)* マン・ページを参照してください。

最初のフィールドには、ローカル・ステーションを呼出すステーション名を指定します。2 番目のフィールドには、指定したステーションが呼出しを行える時刻を示します。値が **Never** の場合は、このステーションは呼出しは受信できても、呼出しを実行できないことを示します。

UUCP のためのリモート・ステーションでの /etc/uucp/Permissions の変更

Permissions ファイルは、リモートのユーザとステーションに関するリモートの uucp アクセスを制御します。オプションについては、207 ページの「UUCP Permissions ファイル」を参照してください。ここでは、Permissions ファイルを次のように記述します。

```
#ident"@(#)uucp:Permissions2.2"
# This entry for public login.
# It provides the default permissions.
# See the Basic Networking Utilities Guide for more
                                                    information.
LOGNAME=nuucp MACHINE=japan READ=/var/spool/uucp/uucppublic\
WRITE=/var/spool/uucppublic REQUEST=yes SENDFILES=yes \
COMMANDS=rmail
```

メモ：上の例ではエントリが複数行に渡って記述されていますが、1 行として解釈されます。

この例では、ユーザ nuucp がローカル・ステーション *japan* からログインできるように指定しています。*japan* の nuucp ユーザは、/var/spool/uucp/uucppublic ディレクトリにあるどのファイルでも読込むことができ、パブリック・ディレクトリ /var/spool/uucppublic に書込むこともできます。*japan* のユーザはファイル転送を要求することができ、*us* のユーザはファイルを送信することができます。

UUCP 接続のテスト

UUCP 接続をテストする基本ツールは、次の 2 つです。

- cu プログラム。228 ページの「cu による UUCP の接続テスト」を参照してください。
- Uutry プログラム。229 ページの「Uutry による UUCP の接続テスト」を参照してください。

cu による UUCP の接続テスト

cu プログラムは UUCP 接続の基本的な機能をテストします。直接 cu を使用する場合は、uucp プログラムと同じログイン操作を行います。cu プログラムは、端末エミュレーション行うために直接モデムを接続する場合にも使用します。cu に **-d** オプションを指定すると、診断トレースが標準出力であるシェル・ウィンドウに出力されます。UUCP 接続をテストする場合は、常にこのモードを使います。

次のコマンドは、リモート・ステーション、UUCP 設定ファイル、およびオペレーティング・システム・ファイルとの物理的な接続をテストします。また、リモート・ステーション上の uucico デーモンもテストします。

メモ：/dev/ttyf2 デバイスのパーミッションはデフォルトで 622 に設定されています。cu がこのデバイスにアクセスできるようにこれを 666 に変更します。

---

1. cu コマンドはローカル（呼出し側の）ステーションから実行します。次に示すように、cu コマンドをローカル・ステーションの *japan* から実行します。

   ```
   /usr/bin/cu -d us
   ```

   次のような内容が出力されます。

   ```
   conn(us)
   Device Type us wanted
   ```

```
mlock ttyf2 succeeded
filelock: ok
fixline(5, 9600)
processdev: calling setdevcfg(cu, us)
gdial(direct) called
getto ret 5
device status for fd=5
F_GETFL=2,iflag='12045',oflag='0',cflag='6275',lflag='0',line='1'
cc[0]='177',[1]='34',[2]='10',[3]='25',[4]='1',[5]='0',[6]='0',[7]='0',
call _mode(1)
Connected
_receive started
transmit started
```

2. 画面が停止したら、**<Enter>** キーを押します。
3. ログイン・プロンプトが表示されたら、nuucp でログインし、nuucp のパスワードを入力します。この例では **secret** になります。

```
Break your connection with a tilde(~) dot(.) and a carriage return (<CR>).
us login: nuucp
Password: secret
IRIX Release 6.2 IP22 us
Copyright (c) 1987-1996 Silicon Graphics, Inc. All Rights Reserved.
Last login: Thu Aug  8 09:14:29 MDT 1996 on ttyf2
here=japan~[us].
call tilda(.)
```

Uutry による UUCP の接続テスト

Uutry は、コピー・イン／コピー・アウト・プログラム（uucico）をテストするプログラムです。uucico が正しく機能していなければ、実際にデータを転送することはできません。Uutry コマンドをローカル・ステーション *japan* からリモート・ステーション *us* に対して実行します。

```
/usr/lib/uucp/Uutry us
```

次のような内容が出力されます。

```
/usr/lib/uucp/uucico -r1 -sus  -x5 >/tmp/us 2>&1&
tmp=/tmp/us
mchFind called (us)
conn(us)
Device Type us wanted
mlock ttyf2 succeeded
processdev: calling setdevcfg(uucico, us)
```

```
gdial(direct) called
getto ret 5
expect: (ogin:)
```

メモ：ここで画面表示が数分間停止します。

```
sendthem (^M)
expect: (ogin:)
^M^M^J^M^J^Jus login:got it
sendthem (nuucp^M)
expect: (ssword:)
nuucp^M^JPassword:got it
sendthem (secret^M)
Login Successful: System=us
msg-ROK
Rmtname us, Role MASTER, Ifn - 5, Loginuser - root
rmesg - 'P' got Pg
wmesg 'U'g
Proto started g
*** TOP ***  -  role=1, setline - X
wmesg 'H'
rmesg - 'H' got HY
PROCESS: msg - HY
HUP:
wmesg 'H'Y
cntrl - 0
send OO 0,exit code 0
Conversation Complete: Status SUCCEEDED
```

`Status SUCCEEDED` が表示された場合は、`Uutry` による `uucico` のテストが成功しています。**`<Ctrl>-C`** キーを押して `Uutry` を終了します。

TCP/IP ネットワークでの UUCP の設定

通常のTCP/IPネットワーク接続を経由してUUCPツールを使用することがあります。`Devices` ファイルには、そのためのエントリが記述されています。ただし、`/usr/etc/inetd.conf` および `/etc/uuc/Systems` ファイルを変更する必要もあります。その場合、次の手順に従います。

1. リモート・ホストの `/usr/etc/inetd.conf` ファイルで次の行を編集します。

   ```
   #uucp stream tcp nowait root /usr/lib/uucp/uucpd uucpd
   ```

行頭にあるシャープ記号（#）を削除し、この行を有効にします。次のコマンドを実行してこの変更を有効化します。

```
/etc/init.d/network stop
/etc/init.d/network start
```

この変更は、UUCP 転送の要求があった場合に、リモート・システム上で uucpd デーモンを実行することを指示しています。

2. ローカル・システムの /etc/uucp/Systems ファイルに次の行を追加します。

   remotehost **Any TCP Any**

   remotehost には、実際に呼出すリモート・ホストの名前を記述します。

3. ローカル・ホストの **root** で /etc/uucp/genperm コマンドを実行し、UUCP の Permissions ファイルを作成します。

4. 次のコマンドを実行し、設定した内容が正しく動作することを確認します。

   ```
   /usr/lib/uucp/uucheck -v
   ```

   これにより多くの情報が出力されます。作成したエントリに対しては、次のような情報が表示されます。

   ```
   When we call system(s): (remotehost)
    We DO allow them to request files.
    They can send files to
    /var/spool/uucppublic (DEFAULT)
    They can request files from
    /var/spool/uucppublic
    /usr/lib/mail
    /usr/people/ftp
    Myname for the conversation will be MyName.
    PUBDIR for the conversation will be /var/spool/uucppublic.

   Machine(s): (remotehost)
   CAN execute the following commands:
   command (rmail), fullname (/bin/rmail)
   command (rnews), fullname (/usr/bin/rnews)
   command (cunbatch), fullname (/usr/lib/news/cunbatch)
   ```

TCP 接続を経由した UUCP に対しては、cu コマンドは動作しません。代りに /usr/lib/uucp/Uutry コマンドを使用します。Uutry については、Uutry(1M) マン・ページと 229 ページの「Uutry による UUCP の接続テスト」で詳しく説明しています。

UUCP のエラー・メッセージ

ここでは、UUCP 環境で発生する一般的なエラー・メッセージについて説明します。UUCP のエラー・メッセージは、ASSERT エラー・メッセージと STATUS エラー・メッセージの 2 種類に大別できます。ASSERT エラー・メッセージについては、232 ページの「ASSERT エラー・メッセージ」を参照してください。STATUS エラー・メッセージについては、234 ページの「STATUS エラー・メッセージ」を参照してください。

ASSERT エラー・メッセージ

プロセスが異常終了すると、ASSERT エラー・メッセージが `/var/spool/uucp/.Admin/errors` に記録されます。これらのエラー・メッセージには、ファイル名、*sccsid*、行番号、表 8-4 に示すテキストが含まれます。これらのエラーのほとんどは、ファイル・システムの問題に起因しています。

表 8-4 Assert エラー・メッセージ

エラー・メッセージ	説明
CAN'T OPEN	**open()** または **fopen()** に失敗しました。
CAN'T WRITE	**write()**、**fwrite()**、**fprint()**、またはその他の呼出しに失敗しました。
CAN'T READ	**read()**、**fgets()**、またはその他の呼出しに失敗しました。
CAN'T CREATE	**create()** の呼出しに失敗しました。
CAN'T ALLOCATE	動的な割当てに失敗しました。
CAN'T LOCK	LCK（lock）ファイルの作成に失敗しました。これは致命的なエラーになる場合もあります。
CAN'T STAT	**stat()** の呼出しに失敗しました。
CAN'T CHMOD	**chmod()** の呼出しに失敗しました。
CAN'T LINK	**link()** の呼出しに失敗しました。
CAN'T CHDIR	**chdir()** の呼出しに失敗しました。
CAN'T UNLINK	**unlink()** の呼出しに失敗しました。
WRONG ROLE	内部に論理エラーが発生しました。

表 8-4 Assert エラー・メッセージ（続き）

エラー・メッセージ	説明
CAN'T MOVE TO CORRUPT DIR	正しくない `C.` または `X.` ファイルをディレクトリ `/var/spool/uucp/.Corrupt` に移動しようとして失敗しました。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ありません。
CAN'T CLOSE	**close()** または **fclose()** の呼出しに失敗しました。
FILE EXISTS	`C.` または `D.` ファイルを作成しようとしましたが、すでにファイルが存在しています。このエラーは、シーケンス・ファイル・アクセスに問題があると発生します。通常、これはソフトウェアのエラーです。
NO UUCP SERVER	TCP/IP を呼出しましたが、UUCP 用のサーバが存在しません。
BAD UID	`/etc/passwd`ファイルに*uid*がありません。ファイルシステムに問題があるか、または `/etc/passwd`ファイルに矛盾があります。
BAD LOGIN_UID	`/etc/passwd`ファイルに*uid*がありません。ファイルシステムに問題があるか、または `/etc/passwd`ファイルに矛盾があります。
ULIMIT TOO SMALL	現在のユーザのプロセスに対する **ulimit** が小さすぎます。ファイルの転送に失敗する可能性があるため、転送は行われません。
BAD LINE	`Devices`ファイルに記述されている行が正しくありません。指定すべき引数が記述されていません。
FSTAT FAILED IN EWRDATA	イーサネット媒体に何らかの問題があります。
SYSLST OVERFLOW	`gename.c` の内部テーブルがオーバーフローしました。要求が大きすぎるか、または不適正です。
TOO MANY SAVED C FILES	`gename.c` の内部テーブルがオーバーフローしました。要求が大きすぎるか、または不適正です。

表 8-4 Assert エラー・メッセージ（続き）

エラー・メッセージ	説明
RETURN FROM FIXLINE IOCTL	失敗すべきでない `ioctl` に失敗しました。システム・ドライバに問題がある可能性があります。
PERMISSIONS file: BAD OPTION	`Permissions` ファイルに記述されている行またはオプションが間違っています。直ちに修正する必要があります。
BAD SPEED	`Devices`または`Systems`ファイルのClassフィールドに記述されているデバイスの速度が正しくありません。
PKCGET READ	リモート・ステーションがハングアップしています。このエラーに対しては、何も行う必要はありません。
PKXSTART	リモート・ステーションが回復不能な形で異常終了しました。このメッセージは無視してもかまいません。
SYSTAT OPENFAIL	`/var/spool/uucp/.Status`のモードに問題があるか、またはディレクトリにモードが正しくないファイルがあります。
TOO MANY LOCKS	内部エラーが発生しました。
XMV ERROR	ファイルまたはディレクトリに問題があります。このプロセスの前に送信先のモードが確認されているため、スプール・ディレクトリに原因があります。
CAN'T FORK	フォークと実行に失敗しました。現在処理中のジョブが失われないように後で再度実行（`uuxqt`）されます。このエラーに対しては、何も行う必要はありません。

STATUS エラー・メッセージ

STATUS エラー・メッセージはディレクトリ `/var/spool/uucp/.Status` に記録されます。このディレクトリには、ローカル・ステーションが通信するリモート・ステーションごとのファ

イルがあります。これらのファイルには、通信を試みた場合にそれが成功したかどうかを示す情報が記述されています。表 8-5 に、記録される一般的なエラー・メッセージを示します。

表 8-5 STATUS エラー・メッセージ

エラー・メッセージ	説明
OK	システムは正常に動作しています。
NO DEVICES AVAILABLE	現在、呼出しを行うシステムに有効なデバイスがありません。指定したステーション用のデバイスが`Devices`ファイルに指定されているかどうか確認します。`Systems` ファイルを調べ、ステーションの呼出しに使用するデバイスを確認します。
WRONG TIME TO CALL	`Systems`ファイルに指定されていない時刻にステーションの呼出しを行いました。
TALKING	現在、通信が行われています。
LOGIN FAILED	指定したステーションへのログインに失敗しました。原因としては、ログイン名、パスワード、または番号が正しくないか、ステーションが特に低速であるか、DTP スクリプトの実行に失敗したことが考えられます。
CONVERSATION FAILED	システム起動後の対話に失敗しました。原因としては、一方のステーションがダウンしているか、プログラムが中断されたか、または回線（リンク）がはずれたことが考えられます。
DIAL FAILED	リモート・ステーションが応答しません。原因としては、ダイヤラが悪いか、または電話番号が間違っていることが考えられます。
BAD LOGIN/MACHINE COMBINATION	ステーションが呼出しに使ったログイン名またはステーション名が `Permissions` ファイルに指定されていません。このような状況では、システムのセキュリティを破壊する可能性があります。
DEVICE LOCKED	要求したデバイスが現在、ほかのプロセスによって使用され、ロックされています。

表 8-5 STATUS エラー・メッセージ（続き）

エラー・メッセージ	説明
ASSERT ERROR	ASSERT エラーが起こりました。`/var/spool/uucp/.Admin/errors` ファイルでエラー・メッセージを調べてから、217 ページの「その他の UUCP ファイル」を参照。
SYSTEM NOT IN Systems	`Systems` ファイルにステーションが指定されていません。
CAN'T ACCESS DEVICE	通常、このメッセージはデバイス・ファイル `/dev/tty*` のパーミッションが正しく設定されていないことを示します。これらのパーミッションを設定するプログラムが異常終了すると、パーミッションは正しい状態に再設定されません。`Systems` と `Devices` ファイルのエントリが正しいかどうかについても確認します。
DEVICE FAILED	デバイスを開くのに失敗しました。
WRONG MACHINE NAME	呼出されたステーションが、呼出しの際に指定したものとは異なる名前を報告しています。
CALLBACK REQUIRED	接続を開始するには、呼出されたステーションがローカル・ステーションを呼戻す必要があります。
REMOTE HAS A LCK FILE FOR ME	リモート・サイトがローカル・システムの LCK ファイルを所有しています。リモート・ステーションがローカル・ステーションを呼出し中である可能性があります。リモート・ステーションの UUCP が古いバージョンの場合、ローカル・ステーションへの通信は、このエラーよりも先に失敗し、LCK ファイルを残した可能性があります。ローカル・ステーションが UUCP の新しいバージョンを使用しており、ローカル・ステーションと通信していない場合は、LCK ファイルを所有しているプロセスが停止します。
REMOTE DOES NOT KNOW ME	リモート・ステーションの `Systems` ファイルにローカル・ステーションのノード名が指定されていません。

表 8-5 STATUS エラー・メッセージ（続き）

エラー・メッセージ	説明
REMOTE REJECT AFTER LOGIN	ローカル・ステーションがログインするために使用した ID が、リモート・ステーションで適切なパーミッションを所有していません。
REMOTE REJECT, UNKNOWN MESSAGE	リモート・ステーションが標準でないためメッセージを出力してローカル･ステーションとの通信を拒否しました。リモート・ステーションが UUCP の標準バージョンを実行していない可能性があります。
STARTUP FAILED	ログインできましたが､最初のハンドシェイクに失敗しました。
CALLER SCRIPT FAILED	このエラー・メッセージが示している問題は、DIAL FAILED が示すエラーと同じです。ただし、このエラーがたびたび起こる場合は、`Dialers` ファイルの caller スクリプトを確認します。これには、`Uutry` を使用します。

IRIX sendmail

この章では、イントラネットを介したメールの経路制御を行う IRIX sendmail について説明します。この章はステーションやネットワーク上でメール・システムを設定し、管理するシステム管理者を対象としており、sendmail の実装について説明します。この章では、以下について説明します。

IRIX sendmail に関しては、付録 B「IRIX sendmail リファレンス」にも説明があります。

メール・システム

メール・システムは、ネットワーク上のユーザ間でメッセージを送受信するのに使用するプログラムです。メールは UUCP または TCP/IP を介して送信することができます。IRIX オペレーティング・システムでは、Netscape Mail、System V の `/bin/mail`、4.3BSD の `/usr/sbin/Mail`、および sendmail を使用できます。

メールの配信には、次の 4 つのプロセスがあります。

ユーザ・インタフェース
: ユーザ・インタフェースでは、新しいメッセージの作成、受信したメッセージの読取り、削除、保存を実行できます。IRIX のユーザ・インタフェースは、Netscape Mail、System V の `/bin/mail`、4.3BSD の `/usr/sbin/Mail` です。これらのインタフェースについては、それぞれのマン・ページを参照してください。Netscape にはオンライン・ヘルプも用意されています。

メールのルーティング
: メール・ルータはメッセージを受取ると、ネットワークを介してそれを適切なステーションにルーティングします。sendmail プログラムはメッセージをルーティングするだけでなく、それを受信するステーションの形式に合わせてメールを変換します。

メール転送
: メール転送プログラムは、あるステーションから別のステーションへメールを送信します。sendmail は TCP/IP 上で SMTP（Simple Mail Transfer Protocol）を実装します。TCP/IP メールでは、sendmail はルーティングと転送の統合プログラムとして機能します。メール転送には常に受信相手が存在します。大半のプログラムでは、送信と受信の両方を実行できます。UUCP はシリアル回線を介し、独自のプロトコルでメールを転送するプログラムです。

メール配信
: メール配信プログラムは、ユーザまたはほかのプログラムが後で読込むことができるように、メールをデータ・ファイルに保存します。`/bin/mail -d` プログラムはローカル・メールを配信します。

Netscape Mail、`/bin/mail`、または `/usr/sbin/Mail` でメッセージを作成すると、そのメッセージは sendmail に送られ、sendmail がメールの配信先を判断します。sendmail は、ローカル・ステーション上のユーザ宛メールの場合は `/usr/bin/mail.local` を呼出し、リモート・ステーション上のユーザ宛メールの場合は、メッセージを適切なメール転送プログラムに渡します。

sendmail はほかのステーションからメッセージを受信すると、受信者のアドレスを分析します。ローカル・ステーション宛のメールの場合は `/usr/bin/mail.local` を呼出して配信を完了し、そうでない場合はメッセージをメール転送プログラムに渡します。TCP/IP SMTP を使用する場合、sendmail 自身がメールの転送も担当します。

TCP/IP を使用しているネットワーク上でメール・メッセージを送信すると、複数のレイヤ（層）でネットワーク・ソフトウェアが実行されます。図 9-1 に TCP/IP メール・ネットワーク・ソフトウェアのレイヤを示します。

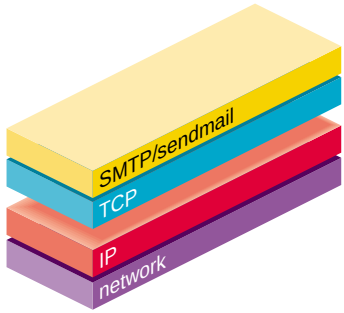

図 9-1　TCP/IP メール・ソフトウェアのレイヤ

sendmail の概要

TCP（Transmission Control Protocol）のレイヤは、sendmail がほかの TCP/IP ステーションにメールを転送する際に使用する SMTP をサポートします。sendmail は、ローカルな配信プログラムを呼出し、メールをルーティングし、TCP/IP メール転送を行います。また、ほかのメール転送プログラムを呼出すこともあります。たとえば、sendmail では UUCP ステーションに送信するメッセージを処理するのに UUCP 伝送プログラムを使います。

sendmail の特徴としては、エリアス、フォワード、ネットワーク・ゲートウェイへの自動経路制御、柔軟な設定が可能であることが挙げられます。

単純なネットワークではノードごとに 1 つのアドレスを持ち、すべてのリソースはホストとリソースのペアで識別できます。たとえば、メール・システムは、ホスト名とユーザ名のペアでユーザを参照します。ステーションの名前とアドレスは中央で集中管理する必要がありますが、ユーザ名はステーションごとにローカルで割当てることができます。

イントラネットでは、異なる特徴や管理方法を持つ複数のネットワークが相互に通信する必要があります。特に異なるのは、リソースを識別する構文と意味です。単純なネットワーク環境であれば、ほかのネットワーク上にあるステーションにローカルなネットワーク名を与えるなどの方法で設定することができます。しかし、実際にはネットワークは複雑な環境になっており、このように簡単に設定することはできません。たとえば、あるネットワークでは、周辺にあるステーションだけをシステム・テーブルに登録すればよいので、point-to-point ルーティングを行っています。このルーティングにより、データベースの更新が簡単になります。ところがほかのネットワークでは、end-to-end アドレス指定を行っています。また、左優先の構文を使用しているネットワークもあれば、右優先の構文を使用しているネットワークもあり、両方のアドレスが混在し、統一されていません。

イントラネットの規格は、こうした問題を解消するためのものです。当初、これらの規格はアドレスのペアをネットワーク、ステーション、リソースの 3 つの要素から構成されるトリプルに拡張しようとしました。ネットワーク番号は統一する必要がありますが、ステーションは各ネットワークでローカルに割当てることができます。ユーザ側から見ると、ドメインを即座に指定できるようになりました。つまり、RFC 1034 の規定に従って、ローカル・リソース、および上位ドメインを含む階層構造になったドメインでアドレスを指定できます。この方式により、物理的な場所（アドレス）にかかわらず、論理的にアドレスを指定できるようになりました。たとえば、jane@iris1.company.com という形式のアドレスは、アドレス空間における論理的な組織を表しています。

sendmail は、完全に独立した相互関係のないネットワークと、一意なネットワーク番号で関連付けられたネットワークとのかけ橋となるものです。このアプリケーションでは、旧式のアドレス構文も受付けます。この場合、sendmail はネットワーク管理者が指定した発見的手法とドメイン・ベースのアドレス指定によってあいまいさを解決します。これは、異種のネットワーク間でやり取りするメッセージの形式も変換します。つまり、sendmail は、一貫したイントラネットのアドレス指定への移行を支援するよう設計されています。

システムの構成

sendmail は次の目的を実現できるように設計されています。

1. 信頼性の高いメッセージ配信を行えるようにします。すべてのメッセージが正しく配信されるか、または配信されない場合は、ユーザにそれを知らせて正しく処理できるようにします。つまり、メッセージが完全になくなってしまうことがないように設計されています。
2. 既存のソフトウェアで実際のメッセージ配信を行うことも可能です。
3. かなり複雑な環境にも対応できるよう、簡単に拡張できます。
4. 設定はプログラム本体に組込まないようにします。
5. ステーションのエリアス・ファイルを変更せずに、いろいろなグループが独自のメール・リストを管理したり、各ユーザが個人での配信先を指定できるようにします。
6. 各ユーザが配信メールを処理するメーラを指定できるようにします。これによって、ユーザがシステムを変更せずに、異なる形式の特殊なメーラで独自の環境を構築できます。また、「休暇中」というメッセージを返送するなど、特殊な機能が使えるようになります。
7. ネットワークのトラフィックを最少限に抑えるために、可能な場合はユーザの手を煩わさずに複数のアドレスを単一のステーションにまとめます。

sendmail の構成

図 9-2 に sendmail の目的に基づくシステム構成を示します。

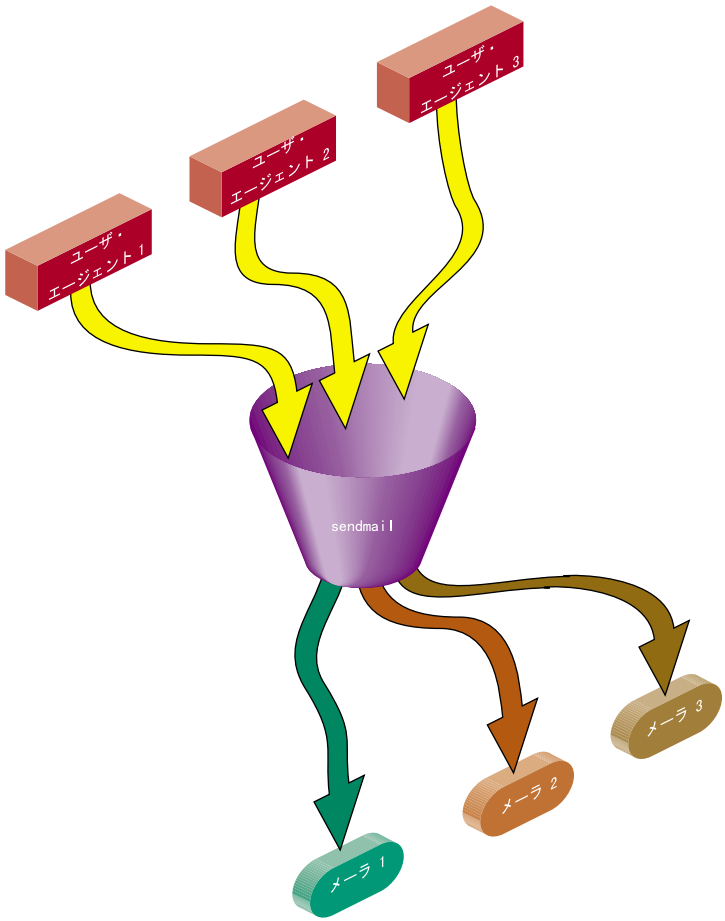

図 9-2 sendmail のシステム構成

sendmail はユーザとのインタフェースを取ったり、メールを実際に配信することはありません。sendmail が実際に行うのは、Berkeley 版の `Mail` などのユーザ・エージェント・プログラムで作成したメッセージを収集し、配信先のネットワークの要求に従ってメッセージを編集し、適切なメーラを呼出してメールを配信するかまたはネットワーク伝送の待ち行列に入れることです。ただし、ファイルに送るメールは例外で、sendmail が直接メールを配信します。

この方法を使うと、最小限のコストで新しいメーラを追加することができます。

送信者がネットワーク・サーバであったり、メーラがネットワーク・クライアントであったりするため、sendmail をイントラネットのメールのゲートウェイとして使うことができます。

sendmail の構成要素

sendmail プログラムについて理解するには、各種の構成要素を理解する必要があります。これらの構成要素には、デーモン、スクリプト、ファイル、およびコマンドがあります。ここでは、これらの sendmail の構成要素について説明します。

- 「sendmail デーモン」（245 ページ）
- 「sendmail のスクリプト」（246 ページ）
- 「sendmail 関連のファイルとディレクトリ」（247 ページ）

sendmail デーモン

sendmail が受信メールを処理するには、デーモンが実行されている必要があります。sendmail デーモンは特有のオプションを持つ sendmail プログラムです。`sendmail` コマンド行オプションについては、付録 B「IRIX sendmail リファレンス」を参照してください。このデーモンは、ステーションの起動時に `/etc/init.d/mail` スクリプトによって自動的に起動されます。次は、sendmail デーモンのデフォルトのコマンドです。

```
/usr/lib/sendmail -bd -q15m
```

`-bd` フラグは `sendmail` をデーモン・モードで実行します。**`-q15m`** フラグは sendmail に待ち行列を行うサブデーモンを 15 分間隔でフォークさせます。**`-bd`** フラグと **`-q`** フラグは 1 つの呼出しで組合わせて指定できます。

sendmail のスクリプト

システムには sendmail の一般的な機能を実行する 2 つのスクリプトが用意されています。次のスクリプトはテスト済みであり、タスクを正しく実行することが分かっているので、可能なかぎり使用します。

- 「sendmail の /etc/init.d/mail スクリプト」（246 ページ）
- 「sendmail の /usr/etc/configmail スクリプト」（247 ページ）

sendmail の /etc/init.d/mail スクリプト

ごくまれなケースですが、ユーザが手作業で sendmail デーモンを停止または起動することがあります。たとえば、設定ファイルを変更する場合は、実行中のすべての sendmail プロセスを停止し、更新した設定ファイルを有効にするために sendmail デーモンを再起動します。このような sendmail の起動と停止が簡単に行えるように IRIX には `/etc/init.d/mail` と呼ばれるシェル・スクリプトが用意されています。

このスクリプトは引数として **`start`** または **`stop`** のいずれかを指定でき、それぞれ sendmail デーモンを起動し、停止します。このスクリプトを実行できるのは、特権ユーザ（root）だけです。たとえば、sendmail を停止するには、次のコマンドを実行します。

```
/etc/init.d/mail stop
```

`/etc/init.d/mail` スクリプトに引数 **`start`** を指定して呼出すと、*sendmail* に関連する各種ファイルとディレクトリ（247 ページの「sendmail 関連のファイルとディレクトリ」を参照）があるかどうか、またそのパーミッションを確認します。`/var/spool/mqueue` ディレクトリなどの必須の構成要素がない場合は、スクリプトがそのディレクトリを作成します。`/etc/aliases` のような複雑な構成要素がない場合は、スクリプトがメッセージを出力して終了します。

`sendmail_cf` “chkconfig” フラグがオンになっている場合、新しい `/etc/sendmail.cf` ファイルは `/etc/sendmail.mc` ファイルから生成されます。

`/etc/init.d/mail` スクリプトに引数 **`stop`** を指定して呼出すと、SIGTERM シグナルで実行中のすべての `sendmail` プロセスを強制終了します。

メモ：ステーションの起動時には、`/etc/init.d/mail` スクリプトが引数 **`start`** で自動的に呼出されます。ステーションが冗長モードで起動する場合、つまり `/etc/chkconfig` の verbose がオンになっていると、次のメッセージが表示されて sendmail が起動したことを確認できます。

```
Mailer daemons: sendmail
```

詳細については、`/etc/init.d/mail` スクリプトを参照してください。

sendmail の /usr/etc/configmail スクリプト

`/usr/etc/configmail` スクリプトは、コマンド行での入力と `sendmail.mc` ファイルとのインタフェースを提供します。`configmail` パラメータは `sendmail.mc` ファイルに保存されます。この `configmail` スクリプトは、`configmail` パラメータの変更を `sendmail.mc` ファイルに保存します。

従来から、`configmail` スクリプトは `sendmail.cf` 設定ファイルを管理するための最も一般的な方法です。次のコマンドを使用すると、`configmail` を使用して `sendmail.mc` ファイルに保存されているパラメータから `/etc/sendmail.cf` ファイルを生成できます。

```
configmail mc2cf
```

`/etc/sendmail.mc` ファイルは `configmail` よりも柔軟性に富んでいるため、`sendmail.mc` を直接修正することをお勧めします。`/etc/sendmail.cf` ファイルを直接編集しないでください。

sendmail 関連のファイルとディレクトリ

sendmail の設定ファイルとディレクトリは次のとおりです。

- 「/etc/sendmail.cf 設定ファイル」（248 ページ）
- 「/etc/sendmail.mc 設定ファイル」（248 ページ）
- 「/etc/sendmail.hf ヘルプ・ファイル」（248 ページ）
- 「/etc/sendmail.st 統計ファイル」（249 ページ）
- 「/etc/aliases エリアス・ファイル」（249 ページ）

- 「/var/spool/mqueue メール待ち行列ディレクトリ」（249 ページ）
- 「/usr/lib/sendmail.cf_m4」（250 ページ）
- 「/var/mail 受信メール・ディレクトリ」（250 ページ）

sendmail のコマンド行にオプションを指定し、sendmail の機能をニーズに合わせて設定することができます。詳細については、付録 B「IRIX sendmail リファレンス」を参照してください。

/etc/sendmail.cf 設定ファイル

sendmail プログラムの中核にあるのが sendmail の設定ファイル `/etc/sendmail.cf` です。この `sendmail.cf` ファイルは、ほとんどの設定情報が記述された ASCII ファイルであり、実行時に読込まれます。このファイルはオプション、ヘッダ宣言、メーラ宣言、認可ユーザ宣言、メッセージの優先順位、アドレス書換えルール、マクロ定義、およびクラス定義をコード化します。

sendmail を正しくセットアップするには、メール管理者は変更する `sendmail.cf` マクロおよび変数を認識する必要があります。

/etc/sendmail.mc 設定ファイル

`sendmail.mc` ファイルには、`sendmail.cf` ファイルを生成するのに必要な一連のコマンドが含まれています。`sendmail.mc` ファイルは `sendmail.cf` ファイルの多様な機能を使用可能にする目的でも使用されます。`sendmail.mc` ファイルは m4 マクロ・プロセッサで書かれています。このため、引用符がつく文字列にはすべて開きの引用符と閉じの引用符が必要です（‘ 例 ’）。`sendmail.mc` の設定についての詳細は、254 ページの「sendmail の設定」を参照してください。

`sendmail.mc` ファイルから `sendmail.cf` ファイルを生成するには、次のコマンドを使用します。

```
configmail mc2cf
```

/etc/sendmail.hf ヘルプ・ファイル

`sendmail.hf` ファイルは SMTP のヘルプ・ファイルです。このファイルには、各種の SMTP コマンドの簡潔な情報が記述されています。

/etc/sendmail.st統計ファイル

sendmail.st ファイルは、sendmail に関連する統計情報を収集します。このファイルはデフォルトでは存在しませんが、touch コマンドで作成することができます。このファイルが存在する場合は、sendmail の統計情報が自動的に記録されます。

/etc/aliasesエリアス・ファイル

aliases ファイルには、sendmail プログラムが使用する aliases データベースがテキスト形式で記述されています。aliases データベースには、ローカルなメール受信者のエリアスが記述されています。たとえば、次のエリアスは、ローカル・ステーション上の jd 宛のメールを johndoe@company.com に配信します。

```
jd:johndoe@company.com
```

メモ：aliases データベース・ファイルを変更した後は、newaliases プログラムを実行する必要があります。aliases データベースの構築については、251 ページの「aliases データベースの構築」を参照してください。

/var/spool/mqueueメール待ち行列ディレクトリ

/var/spool/mqueue は、メールの待ち行列と一時ファイルが置かれるディレクトリです。メッセージは /var/spool/mqueue ディレクトリにある各種の待ち行列ファイルに格納されます。待ち行列ファイルは、次の形式になります。

- qf*— メッセージの制御（待ち行列）ファイル
- df*— データ・ファイル
- tf*— 一時ファイル
- nf*— 一意な ID を生成するファイル
- xf*— 現在のセッションのトランスクリプト・ファイル

通常、sendmail のサブデーモンが定期的にこの待ち行列にあるメッセージを処理し、各メッセージの配信を試みます。/etc/init.d/mail スクリプトが sendmail デーモンを起動し、この

デーモンが 15 分間隔でサブデーモンをフォークしてメールの待ち行列を処理します。sendmail は待ち行列を処理するたびにそれを読込み、ソートし、すべてのジョブを順次実行します。

メモ：この方法で生成されたメールの待ち行列は、前のバージョンの sendmail では読取ることができません。

/usr/lib/sendmail.cf_m4

`/usr/lib/sendmail.cf_m4` ディレクトリには、`sendmail.cf` ファイルの生成に使用するデータが含まれています。このディレクトリの内容は `sendmail` のソース・ディストリビューションの `cf` サブディレクトリに類似しています。`sendmail.mc` ファイルは、`/usr/lib/sendmail.cf_m4` ディレクトリ内のファイルによって実行される設定および機能を選択します。これらのファイルから新しい `sendmail.cf` ファイルを生成するには、次のコマンドを使用します。

```
configmail mc2cf
```

/var/mail 受信メール・ディレクトリ

`/var/mail` はすべての受信メールを格納するディレクトリです。ローカル・ステーションの各ユーザは、自分のメールを `/var/mail` ディレクトリにあるファイルに受信します。たとえば、ユーザ *guest* は自分のメールを `/var/mail/guest` ファイルに受信します。

sendmail の aliases データベース

ここでは、sendmail の aliases データベースの使い方と、予測される問題点やエラーについて説明します。以下のセクションを参照して下さい。

- 「aliases データベースの構築」（251 ページ）
- 「sendmail の aliases データベースのテスト」（253 ページ）
- 「sendmail の aliases データベースにかかわる問題」（253 ページ）
- 「sendmail リストの所有者」（253 ページ）

aliases データベースとは、sendmail プログラムが使用するメール・エリアスが記述された NEWDB データベースです。このデータベースのテキスト形式は `/etc/aliases` ファイルで管理されます。エリアスは次の形式で記述します。

```
name: name1 [, name2, ...]
```

たとえば、次のコマンドは jd 宛のメールを johndoe@company.com に送信します。

```
jd:johndoe@company.com
```

メモ：エリアスを指定できるのは、アドレスのローカル部分だけです。たとえば、次のようにコマンドで指定した場合、メールは送信されません。

```
jd@big.university.edu:jd@company.com
```

sendmail が aliases データベースを参照するのは、メッセージをローカルに配信すると判断し、ローカル部分しか持たないアドレスにメール・アドレスを書換えた後です。

エリアスの継続行は、スペースまたはタブで開始します。空白行やシャープ記号（#）で開始されている行は、コメントとして処理されます。

aliases データベースの構築

sendmail は起動時に自動的に `newdb` ライブラリを使って `/etc/aliases` ファイルの内容を `/etc/aliases.db` ファイルに変換します。このファイルを使用してエリアスを実行すると、パフォーマンスが向上します。

NEWDB 形式のデータベースを使用して aliases を管理する場合、sendmail は `/etc/alias.db` ファイルを使用します。NEWDB 形式のデータベースを構築する場合、sendmail は各エントリの左側を調べ、それがローカル・アドレスであることを確認します。左側がローカル・アドレスではないエントリが `/etc/aliases` ファイルにあると、sendmail は警告を出力します。このようなエントリは、NEWDB 形式のデータベースには加えられません。

前の IRIX sendmail のバージョンでは、`/etc/alias` ファイルでキーワード **+:+** を検索し、NIS への照会を有効にしていました。現行の IRIX では、この操作は明示的に行う必要があります。デフォルトで IRIX sendmail は、`/etc/aliases` データベースと NSD マップである

mail.aliases を照会に使用します。NIS を使用するように nsd を構成している場合は、特別な作業は必要ありません。そのような構成になっていない場合は、sendmail.mc ファイルに次のラインを入れる必要があります。

```
define (ALIAS_FILE',/etc/aliases,nsd:mail.aliases') dnl
```

設定ファイルまたはコマンド行で **D** オプションを指定すると、データベースが古い場合、sendmail が自動的に aliases データベースを再構築します。

sendmail は次のいずれかの場合に aliases データベースを再構築します。

- NEWDB 形式のデータベースのモードが 666 になっている場合。
- sendmail が setuid を root で実行している場合。

メモ：負荷の重いステーションが大きな aliases ファイルを持っている場合、自動再構築を行うと危険です。データベースの再構築に 5 分以上かかる場合は、複数のプロセスが同時に再構築を行う可能性があります。

---

newaliases プログラムは、NEWDB 形式の aliases データベースを再構築します。テキスト形式の aliases ファイルを変更する場合は、常にこのプログラムを実行する必要があります。aliases ファイルを変更した後に newaliases を実行しないと、変更が NEWDB 形式の aliases データベースに組込まれないため、sendmail プログラムで認識されません。

sendmail を再起動せずに NEWDB 形式のデータベースを再構築するには、次のコマンドを実行します。

```
newaliases
```

newaliases を実行する代わりに、次のように **-bi** フラグを指定して sendmail を実行しても同様の結果が得られます。

```
/usr/lib/sendmail -bi
```

sendmail の aliases データベースのテスト

aliases データベースをテストするには、sendmail に **-bv** フラグを指定します。詳細については、付録 B「IRIX sendmail リファレンス」を参照してください。

sendmail の aliases データベースにかかわる問題

NEWDB 形式のデータベースの再構築が完了する前に sendmail がこのデータベースにアクセスすると、aliases データベースに問題が発生する可能性があります。これは、次の 2 つが原因と考えられます。

- あるプロセスがデータベースを再構築中に、ほかのプロセスがデータベースにアクセスしています。
- データベースの再構築を行っているプロセスが強制終了したか、再構築が完了する前にステーションがクラッシュしました。

sendmail には、この問題を解決する 2 つの方法があります。1 つは、再構築が途中で中断されないように、データベースの再構築中は割り込みを無視することです。もう 1 つの方法は、再構築終了時に次の形式のエリアス（通常は無効なエリアス）を追加することです。

```
@: @
```

sendmail はデータベースにアクセスする前に、このエントリがあるかどうか確認します。この動作を実行させるには、設定ファイルに **-a** オプションを指定します。

@:@ エントリがない場合、sendmail はそれが現れるまで待ちます。指定された待ち時間が経過すると、sendmail は強制的に再構築を開始します。この動作を実行するには、設定ファイルに **D** オプションを指定します。**D** オプションが指定されていないとエリアスに関する警告メッセージが出力され、sendmail は処理を続けます。

sendmail リストの所有者

メールを任意のアドレス（たとえば *x*）に送信したときにエラーが発生した場合、sendmail は次の形式のエリアスを探してエラーを受取ります。

```
owner-x
```

これは、メーリング・リストにメールを送るユーザがリスト自体の管理をできない場合に有用です。つまりこの場合、リストの管理者がリストの所有者になります。たとえば、次に示すエリアスでは、unix-hackers にメールを送った際に発生するエラーを jd@1company.com が受取り、sendmail はリストにある偽のユーザ nosuchuser を見つけ出すことができます。

```
unix-hackers: jd@company1.com, ed@big.university.edu,nosuchuser,
jane@company2.com
owner-unix-hackers: jd@company1.com
```

sendmail の設定

このセクションでは、sendmail の設定について説明します。ここでの説明は、`sendmail.cf` ファイルおよびそのすべての機能をプログラミングするための完全なリファレンスではありません。sendmail の設定の詳細については、以下を参照してください。

- Bryan Costales、Eric Allman 共著『sendmail, 2nd Edition』。sendmail の権威とされているこの本は O'Reilly & AssociatesInc から購入できます。（ISBN: 1-56592-222-0）（URL: http://www.oreilly.com/catalog/sendmail2）
- `sendmail.org` の Web サイト（http://www.sendmail.org）には、sendmail に関する情報が多く提供されています。
- `/usr/lib/sendmail.cf_m4/README` ファイルには、`sendmail.mc` ファイルおよび m4 マクロ・プロセッサの機能と構成に関する詳細情報が含まれています。このファイルは、パラメータを変更できる定義ステートメントも含んでいます。

sendmail.cf ファイル

`sendmail.cf` ファイルは sendmail の各コピーからリアルタイムに読込まれます。IRIX sendmail の前バージョンがサポートしていたこのファイルの凍結バージョンはありません。このセクションでは sendmail.cf ファイルの詳細は提供されていませんが、代わりに `sendmail.mc` ファイルを使って `sendmail.cf` ファイルの設定および生成を行う方法について説明があります。

sendmail.mc ファイル

sendmail.mc ファイルはいくつかの定義、設定、コマンドによって構成されており、数々の機能を使用可能にします。このファイルを使用して、/usr/lib/sendmail.cf_m4 ディレクトリ内にあるプロトタイプ・ファイルから sendmail.cf ファイルを作成します。

sendmail.mc ファイルにある定義から sendmail.cf を作成するには、次のコマンドを使用します。

```
configmail mc2cf
```

次のセクションにある定義と機能を使用することにより、ここにリストされているような複雑な機能をいくつも実行できます。なお、実行可能な機能はここにリストされているものに限りません。

- マスカレードおよびリレー
- アンチスパム設定の管理
- 新しいメーラーまたはルールの追加
- 非 smtp ベース設定の使用
- メーラー・テーブルの使用
- userdb を使用したフル・ネームの割当て

m4 マクロ・プロセッサ

sendmail アプリケーションは m4 マクロ・プロセッサ（通称 m4）を使用して設定ファイルを「コンパイル」します。m4 はストリーム・ベースであるため、行を認識しません。このため、出力に不必要な空白行ができてしまうことを避けるために、**dnl**（delete through newline の略）の文字が表示されることがあります。**dnl** 命令で始まる文字と次の改行文字がすべて削除されます。

もうひとつの重要な命令は **define**（A、B）です。この命令は値 B を得るためにマクロ A を定義します。マクロは読込まれるときに展開します。展開を避けるには、次の例のように両方の値を引用符で囲む必要があります。

```
define(‘SMART_HOST’, ‘smart.foo.com’)
```

開き引用符はティルダ（~）のキーをシフト変換させないで打つと表示されます。エディタによっては、このキーを 2 度押さないと引用符が表示されない場合があります。閉じ引用符は通常の一重引用符で、二重引用符（"）と同じキーです。

警告：m4 マクロは、行内にコメントとして引用されている場合も拡張します。次のコメント例では、FEATURE(foo) が展開されます。

```
# See FEATURE(foo) above
```

このような展開を避けるには、次の例のように FEATURE(foo) を引用符で囲みます。

```
# See ‘FEATURE(foo)’, above
```

次のコメント例では、**define** は m4 キーワードなので、引用符で囲む必要があります。

```
# And then ‘define’ the $X macro to be the return address
```

変数の定義

設定オプションの大半は変更する必要がありません。しかし、修正が必要な場合は、m4 変数を定義できます。これらの変数の一覧は、次に示すURLのsendmail標準で見つけることができます。

```
http://www.sendmail.org/m4/tweakingoptions.html
http://www.sendmail.org/m4/ostype.html
```

修正可能な変数の一覧には次のような情報が表示されます。

- 変数名
- 影響を受けるオプションまたはマクロ（オプションは Ox、マクロは Dx）
- 定義のデフォルト値
- 簡単な説明

いくつかのオプションは、将来のバージョンではその重要度が低くなることが予想されます。つまり、前バージョンとの互換性を保つ目的のためだけにそれらのオプションは含まれる場合があります。こうしたオプションにはアスタリスク（*）が付いています。

これらのオプションは m4 変数であるため、引用符で囲む必要がある場合があります。特にカンマが付いている引数は、カンマが存在することで起こる混乱を防ぐために、通常 “ このように、二重引用符で囲む ” 必要があります。これは、エリアス・ファイルの定義および読込みタイムアウトについても同様です。

オペレーティング・システム環境の定義（OSTYPE マクロ）

オペレーティング・システム環境を定義しないと、設定ファイルの生成は失敗します。定義できる環境はいくつかあります。現在定義できる環境の一覧は、`sendmail.cf_m4` ディレクトリ内の `ostype` ディレクトリにあるファイルをご覧ください。このマクロは、エリアス・ファイルおよびキュー・ディレクトリの場所といった要素を変更します。これらのファイルの中には同一のものがあります。

MAILER 定義の前に OSTYPE マクロを置く必要があります。一般的に、OSTYPE マクロはバージョン情報のすぐ後に置き、MAILER 定義は常に一番最後になります。

ユーザが OSTYPE を修正する必要はありません。OSTYPE はデフォルトで **`irix6`** に設定されています。

ドメインの定義（DOMAIN マクロ）

ドメイン依存の定義を 1 つのファイルに集め、DOMAIN マクロがそのファイルを参照できるようにしておくと便利な場合があります。必ずしも 1 つのファイルに集めなければならないわけではありませんが、こうしておくと 1 つのファイルにサイトの設定を定義しておき、このファイルを個別のシステムに配布することができます。たとえば、Berkeley ドメイン・ファイルにはいくつかの内部ホストの定義が含まれています。

USENET_RELAY
: USENET 宛ての電子メールを受け入れるホスト。定義されていない場合、USENET 宛ての電子メールは USENET メーラーによって通常のメールとして扱われます。USENET 宛ての電子メールの形式は *newsgroup*.USENET です。

UUCP_RELAY
: UUCP 宛ての電子メールを受け入れるホスト。定義されていない場合、すべての UUCP サイトは直接接続されていなければなりません。

BITNET_RELAY
: BITNET 宛ての電子メールを受け入れるホスト。定義されていない場合、.BITNET の擬似ドメインは動作しません。

DECNET_RELAY
: DECNET 宛ての電子メールを受け入れるホスト。定義されていない場合、.DECNETの擬似ドメインおよびノード::ユーザ形式のアドレスは動作しません。

FAX_RELAY
: .FAX 擬似ドメイン宛ての電子メールを受け入れるホスト。「fax」メーラーはこの値をオーバーライドします。

LOCAL_RELAY DEPRECATED
: @ の後にドメイン拡張子がないような、不適切な名前を処理するサイト。設定されていない場合、これらの名前はこのマシン上にあると想定されます。これにより、中央サイトに全社レベルまたは部署レベルのエリアス・データベースを保存することができます。これは小規模サイトでのみ動作し、動作するユーザ・エージェントも限られます。

LUSER_RELAY
: ローカル・アカウントやローカル・エリアスでないローカル名など、適切ではないメールを処理するサイト。

メーラーの定義（MAILER マクロ）

このバージョンの sendmail では、多くのメーラーがサポートされています。一般的な決まりでは、MAILER 定義は `.mc` ファイルの最後に置くことになっています。いくつかの機能と定義はメーラーの定義を修正するため、また、smtp メーラーは uucp メーラーを修正するため、MAILER(smtp) は常に MAILER(uucp) の前に置きます。サポートされているメーラーを以下に示します。

local
: local および prog メーラー。これらはほぼ必須ですが、メールをすべて別のサイトにリレーする場合は例外です。このメーラーは自動的にインストールされます。

smtp
: 簡易メール転送プロトコル（SMTP）メーラー。このメーラーはインターネット上でのピア・ツー・ピア・メール転送の標準メーラーです。このメーラーは、すべてのサイトがネーム・サーバを起動していることを想定しています。このファイルは実際は 4 つのメーラーを定義します。1 つ目は従来の SMTP から別のサーバへの転送に使う「smtp」です。2 つ目は拡張 SMTP から別のサーバへ

の転送に使う「esmtp」です。3 つ目は 8 ビットのデータを MIME に変換せずに SMTP から別のサーバに転送する「smtp8」です。この場合、転送先の確認がなくても、転送元は転送先が 8 ビットのデータを処理する能力を持っていることを示唆しています。4 つ目は弊社の RELAY_HOST, LUSER_RELAY または MAILER_HUB への転送に使う「relay」です。

uucp — UNIX から UNIX へのコピー・プログラム・メーラー。このメーラーは実際には 2 つのメーラー（「uucp-old」別名「uucp」、「uucp-new」別名「suucp」）を指します。後者は、転送先の UUCP メーラーが 1 回の転送で複数の受信者を処理できる場合に使用します。設定に smtp メーラーも含まれている場合、別の 2 つのメーラー（「uucp-dom」と「uucp-uudom」）も定義されます［警告 : MAILER(uucp) を指定する前に MAILER(smtp) を指定する必要があります］。uucp メーラーが含まれている場合、sendmail は $=U クラスにある名前をすべて検索し、それらの名前を uucp-old メーラーに送ります。$=Y クラスにある名前はすべて uucp-new に送られます。また、$=Z クラスにある名前はすべて uucp-uudon に送られます。この機能は受信側で実行している rmail のバージョンに依存するため、送信側でコントロールできない場合があることに注意してください。詳細は、`/usr/lib/sendmail.cf_m4` ファイルを参照してください。

usenet — Usenet（ネットワーク・ニュース）の配信。このメーラーが指定されていると、ruleset 0 のルール・セットにルールが追加され、「group.usenet」の名前がつくユーザに送られたローカルの電子メールがすべて「inews」プログラムに転送されます。このメーラーはすべてのグループに対して動作するため、セキュリティ問題を考慮する必要があります。

fax — ファックス転送。これは現在実験段階にあり、Sam Leffler の HylaFAX ソフトウェアを使って行われています。詳細は、http://www.vix.com/hylafax/ を参照してください。

pop — ポスト・オフィス・プロトコル

procmail — procmail へのインタフェース（sendmail には含まれていません）。このメーラーは、メーラー・テーブル内で使用するように設計されています。「あるドメインのメールを 1 人のユーザにすべて転送するにはどうしたらよいか ?」というよくある問題も、メーラー・テーブルで解決できます。このメーラーが定義されている場合、メーラー・テーブルを次のようにセットアップできます。

```
host.comprocmail:/etc/procmailrcs/host.com
```

/etc/procmailrcs/host.com ファイルには、次のエントリが含まれます。

```
:0# forward mail for host.com

! -oi -f $1 person@other.host
```

この設定によって、(xxxxx)@host.com に対するメールは person@other.host に送られます。procmail スクリプト内では、$1 は送信元の名前で、$2 は送信先の名前です。これを FEATURE(`local_procmail') と共に使用する場合は、FEATURE マクロを最初に記述してください。

mail11 — DECnet mail11 メーラー。mail11 プログラム（DECnet および gatekeeper.dec.com:/pub/DEC/gwtools から入手可能）がある場合にのみ使用できます。このメーラーは Phase IV DECnet に対応しており、サイトに Phase V がある場合、問題が生じる可能性があります。

phquery — phquery プログラム。これは、直感的な略称ではありませんが「ph」メーラーとして内部的に参照されています。CCSO ネーム・サーバの検索に使用できます。このメーラーが使用する phquery プログラムは、ph クライアントと共に配布されます。

cyrus — cyrus および cyrusbb メーラー。cyrus メーラーはローカル cyrus ユーザへの配信を行います。このメーラーは「user+detail@local.host」構文を利用できます。メールボックスの ACL の許可があれば、メールはユーザの「detail」メールボックスに配信されます。cyrusbb メーラーは、メールボックスの ACL の許可があれば、システム全体の cyrus メールボックスへの配信を行います。

ローカル・メーラーは「user+detail」のアドレスに対応しています。この場合、「+detail」はメールボックスのマッチングには使用されず、特定のローカル・メール・プログラムに使用されます（詳細は FEATURE(`local_procmail') を参照してください）。たとえば、「eric」、「eric+sendmail」、「eric+sww」はすべて同じユーザを示していますが、メールをソートする目的で追加の引数 <null>、「sendmail」、「sww」が提供される場合があります。

機能の要求（FEATURE マクロ）

FEATURE マクロを使用すると、特別な機能を要求できます。たとえば、/etc/sendmail.cw ファイルを読取って、クラス $=w の値を取得するように sendmail に指示するには、sendmail.mc ファイルに次の FEATURE マクロを入れます。

```
FEATURE(`use_cw_file')
```

FEATURE マクロには、次の例にあるように、1 つのパラメータをオプションで含めることができます。

```
FEATURE(`mailertable', `dbm /usr/lib/mailertable')
```

define コマンドを使用すると、テーブル機能にデフォルトのデータベース・マップ・タイプを設定できます。次のコマンドは NEWDB データベースを使用するためのデフォルトを設定します。

```
define(`DATABASE_MAP_TYPE', `dbm')
```

デフォルトは、Berkeley db ハッシュ・データベース形式です。FEATURE マクロで引数を指定する場合も、データベース・マップ・タイプを宣言する必要がある点に注意してください。DATABASE_MAP_TYPE は、FEATURE マクロに引数がない場合にのみ使用されます。

次の機能が使用できます。

use_cw_file　`/etc/sendmail.cw` ファイルを読取り、このホストの代替名を取得します。これは、別のホストの動的集合に使う MX レコードを提供するホスト上にいるときに使用できます。集合が静的な場合は、「Cw< 名前 1>< 名前 2>...」（ここでの名前は完全なドメイン名です）という行を追加します。confCW_FILE 変数を再び定義すると、実際のファイル名をオーバーライドできます。

use_ct_file　`/etc/sendmail.ct` ファイルを読取り、信頼できるユーザの名前を取得します。つまり、警告メッセージを生成せずに、**`-f`** を使ってエンベロープの「送信元」アドレスを設定できるようになります。confCT_FILE 変数を再び定義すると、実際のファイル名をオーバーライドできます。

redirect　「address.REDIRECT」に送られたメールをすべて拒否し、次のメッセージを表示します。

```
551 User not local; please try <address>
```

この機能が設定されると、「.REDIRECT」が付いている新しいアドレスに移行したユーザをエリアシングすることができます。

nouucp　UUCP アドレスの特別扱いを防ぎます。

nocanonify　Canonical 化のためにアドレスを $[...$] に渡すのを防ぎます。この機能はメール・ゲートウェイとしてのみ機能しているサイト、または完全な Canonical 化を行うユーザ・エージェントがあるサイトのみが使用できます。次に示す定義コマンドを使用して、同じような機能を実行する通常のレゾルバ・オプションをオフにすることもできます。

```
define(`confBIND_OPTS',`-DNSRCH -DEFNAMES')
```

stickyhost　「user@local.host」に送られた電子メールが「sticky」とマークされていることを示します。つまり、ローカル・アドレスが UDB と一致せず、ruleset 5 を通過しないことを意味します。この機能は、「user」が必ずしも「user@local.host」と同一ではない場合に、たとえば異なるドメイン全体のネームスペースを作成するためなどに使用します。バージョン 8.7 より前のリリースでは stickyhost がデフォルトで、この機能をオフにするには notsticky を使用します。

mailertable　特定のドメインのルーティングをオーバーライドするために使用する「メーラー・テーブル」が含まれていることを示します。FEATURE の引数がキー定義となる場合もあります。何も指定されていない場合、次の定義が使用されます。

```
hash -o /etc/mailertable
```

このデータベースのキーは、完全なドメイン名（「vangogh.CS.Berkeley.EDU」など）あるいはドットが先頭の不完全なドメイン（「.CS.Berkeley.EDU」など）のいずれも可能です。値は「mailer:domain」の形式にする必要があります。この場合、「mailer」は内部メーラーの名前で、「domain」はメッセージの送信先になります。これらのマップはメッセージ・ヘッダには反映されません。特別なケースとして、次の場合があります。

local:user — ローカル・メーラーを使用して、指定されているユーザに転送します。

local: — ローカル・メーラーを使用して、電子メール・アドレスにある送信元ユーザに転送します。

error:code message — コードとメッセージがついたエラー・メッセージを表示します。

domaintable　ドメイン名のマッピングを行う「ドメイン・テーブル」が含まれていることを示します。この機能は自分のドメインに対してのみ使用してください。この機能は、会社が「古い社名 .com」から「新しい社名 .com」に社名変更したような場合に便利なことがあります。FEATURE マクロの引数がキー定義となることがあります。何も指定されていない場合、次の定義が使用されます。

```
hash -o /etc/domaintable
```

このテーブルでのキーはドメイン名です。値は新しい（完全な名前の）ドメインです。domaintable にあるものはすべてヘッダに反映されます。つまり、これは ruleset 3 で行われます。

bitdomain　テーブルにある bitnet ホストを検索し、これらのホストをインターネット・アドレスに変換します。このテーブルは、John Gardiner Myers の bitdomain プログラムを使用して作成できます。FEATURE マクロの引数がキー定義となることがあります。何も指定されていない場合、次の定義が使用されます。

```
hash -o /etc/bitdomain.db
```

キーは bitnet ホスト名です。値は対応するインターネット・ホスト名です。

uucpdomain　bitdomain と同じような機能を UUCP ホストに提供します。デフォルトのマップ定義は以下のとおりです。

```
hash -o /etc/uudomain.db
```

現在のところ、このデータベースを自動的に作成するためのツールはありません。

always_add_domain

ローカルに送信されるメールにもローカル・ホスト・ドメインが含まれていることを示します。通常、不完全な名前にはローカル・ホスト名は追加されません。ただし、共有のメッセージ保管場所を使用していても同じユーザ名スペースをあらゆる場所で使用していない場合、ローカル名にローカル・ホスト名が必要なことがあります。

allmasquerade

MASQUERADE_AS マクロによってマスカレードが有効になっている場合、この機能によって送信先のアドレスもマスカレードされます。通常、送信先のアドレスにはローカル・ホスト名が与えられます。これは多くのユーザにとって役立つ機能ですが、一方でローカル・エリアスを壊してしまう場合がありま

す。たとえば、「localalias」に送信する場合、送信元の sendmail はこのエリアスを確認してすべてのメンバーに送信しますが、そのメッセージは「To: localalias@masqueradehost」という形で送信されます。このエリアスが実在することはほとんどないため、返信は失敗します。このエリアスは、マスカレード・ホスト上の全ネームスペースがすべてのローカル・エンティティを確実にスーパーセットできる場合にのみ使用してください。

limited_masquerade

通常、$=w にリストされているホストはいずれもマスカレードされています。この機能がオンになっている場合、$=M にリストされているホストだけがマスカレードされます。この機能は、同じマシン上にホストされている共通要素を持たないネームスペースにドメインがいくつかある場合に便利です。

masquerade_entire_domain

MASQUERADE_AS によってマスカレードが有効になっており、MASQUERADE_DOMAIN が設定されている場合、この機能によってアドレスは書き換えられ、マスカレード・ドメインは、実際には見えませんがドメイン全体となります。マスカレード・ドメイン内のホストはすべて、MASQUERADE_ASによってマスカレード名に書き換えられます。次の例では、*foo.orgと*bar.comはmasq.comに変換されます。「masquerade_entire_domain」の機能が設定されていない場合、マスカレードされるのは foo.org と bar.com のみです。

```
MASQUERADE_AS(masq.com)
MASQUERADE_DOMAIN(foo.org)
MASQUERADE_DOMAIN(bar.com)
```

メモ：この機能は、自分の管轄と現在の階層の中にあるマスカレード・ドメインにのみ使用してください。

masquerade_envelope

通常、ヘッダ・アドレスのみがマスカレードされます。この機能を使用すると、メッセージのエンベロープをマスカレードできます。

genericstable　ローカルから送信される特定のアドレス（不完全なアドレス）または $=G にリストされているドメインがマップ内で検索され、別の（一般的な）形式に変換されます。このとき、ドメイン名もユーザ名も変更される場合があります。こ

れは userdb の機能に似ています。マスカレードに関しては、allmasquerade 機能と masquerade_envelope 機能の両方、あるいはどちらか一方が選択されていない限り、同じ種類のアドレスが検索されます（ヘッダ送信元アドレスのみ）。完全なアドレスには、GENERICS_DOMAIN マクロまたは GENERICS_DOMAIN_FILE マクロ（MASQUERADE_DOMAIN および MASQUERADE_DOMAIN_FILE に類似）から得られる名前のリストにあるドメイン部分が必要です。

FEATURE(`genericstable') の引数がマップ定義になることもあります。デフォルトのマップ定義は次のとおりです。

```
hash -o /etc/genericstable
```

このテーブルのキーは、完全なアドレスか不完全なユーザ名のいずれかです（完全なアドレスが先に試されます）。値は新しいユーザ・アドレスで、これにドメインが含まれていない場合、$j またはマスカレード名を使用するという通常の方法で完全な名前にされます。検索されるアドレスが完全な名前である必要がある点に注意してください。ローカル・メールでアドレスを完全にするには、FEATURE(`always_add_domain') を使用します。

virtusertable　エリアスのドメイン依存形式を示し、1 つのマシンが複数のバーチャル・ドメインをホストできるようにします。`virtusertable` は、たとえば次のように使用します。

```
info@foo.com      foo-info
info@bar.com      bar-info
@baz.org          jane@elsewhere.net
```

info@foo.com に宛てられたメールは foo-info のアドレスに送信されます。info@bar.com に宛てられたメールは bar-info に送信されます。baz.org にリストされているユーザに宛てられたメールは、どのユーザに宛てられたものであれ、jane@elsewhere.net に送信されます。送信元アドレスからのユーザ名は %1 として渡され、以下のようになります。

```
@foo.org          1@elsewhere.com
```

この指定により、someone@foo.org は someone@elsewhere.org に送信されます。

左側のホスト名（foo.com、bar.com、および baz.org）はすべて $=w の中になければなりません。デフォルトのマップ定義は以下のとおりです。

```
hash -o /etc/virtusertable
```

次の例で示すように、新しい定義を FEATURE マクロの 2 番目の引数として指定できます。

```
FEATURE(`virtusertable', `dbm -o /etc/mail/virtusers')
```

nodns　サイトが DNS を実行していないことを示します（たとえば、サイトが UUCP のみに接続されている場合)。

nullclient　ローカルの SMTP ベース・ネットワークを経由してすべてのメールを中央ハブに転送するサポートのみが含まれている設定ファイルを作成します。ハブの名前が引数になります。

nullclient と共に使用できる機能は「nocanonify」のみです。nocanonify では、アドレスは SMTP 接続を経由して不完全なまま送信されます。通常、これらのアドレスはマスカレード名によって完全な形にされますが、マスカレード名はハブ・マシンの名前をデフォルトとして指定します。メーラーは一切定義されませんし、エリアスや転送も行われません。また、ヌル・クライアント設定ではアンチスパムやアンチリレーも行われない点に注意してください。

local_lmtp　LMTP 対応のローカル・メーラーを使用していることを示します。この機能の引数は LMTP 対応のメーラーへのパス名です。デフォルトで mail.local が使用されます。このデフォルトは、バージョン 8.9 に添付されている LMTP 対応の mail.local を前提としています。mail.local へのパスは confEBINDIR m4 変数によって設定され、LOCAL_MAILER_PATH`/usr/libexec/mail.local` をデフォルトに指定します。

local_procmail

ローカル・メーラーとして procmail を使用していることを示します。このメーラーは「user+indicator@local.host」の構文を利用できます。通常、+indicator は使用されませんが、これはデフォルトでは procmail への **-a** 引数として渡されます。この機能の引数は procmail のパス名で、デフォルトは PROCMAIL_MAILER_PATH です。ローカル・メーラーに PROCMAIL_MAILER_FLAGS または PROCMAIL_MAILER_ARGS が使用されていない点、その代わりに LOCAL_MAILER_FLAGS および LOCAL_MAILER_ARGS に変更されている点に注意してください。

bestmx_is_local

最も可能性のある MX レコードを含むローカル・アドレスがリストされているホストであれば、メールがあたかもローカルに宛てられたかのように受入れられることを示します。受入れられることによって追加の DNS トラフィックが生じますが、低〜中程度のトラフィックのホストでは特に問題にはなりません。また、引数に複数のドメインを指定することもできます。この場合、この機能は決められたドメインにのみ適用され、不必要な DNS トラフィックを減らすことができます。

注意：この機能は基本的にはワイルドカード MX レコードと互換性がありません。ドメインに一致するワイルドカード MX レコードがある場合、この機能は使用できません。

---

smrsh　メール送信プログラムとして、`/bin/sh` の代わりに添付されている SendMailRestricted Shell (smrsh) を使用します。これにより、ローカルのシステム管理者は電子メールを経由して実行される機能の制御能力を向上できます。引数がある場合、それは smrsh へのパス名になります。引数がない場合、confEBINDIR で定義されたパスが smrsh バイナリ・ファイルへのパスとして使用されます。通常、デフォルトは `/usr/libexec/smrsh` です。

promiscuous_relay

デフォルトでは、sendmail 設定ファイルはメールのリレーを許可しません。つまり、デフォルトでは自分のドメイン以外からのメールを受入れ、そのメールを自分のドメイン以外のホストに送信することはありません。このオプションを使うと、どのサイトからも、どのサイトへもメールのリレーができるように設定できます。一般的に、リレーはアクセス・データベースおよび「R」クラス（$=R）を使って注意深く管理してください。ドメインは RELAY_DOMAIN マクロまたは RELAY_DOMAIN_FILE マクロ（MASQUERADE_DOMAIN および MASQUERADE_DOMAIN_FILE に類似）によってクラス「R」に追加できます。

relay_entire_domain

デフォルトでは、アクセス・データベースに RELAY としてリストされているホストのみがリレーできる設定になっています。このオプションを使うと、「m」クラス（$=m）によって定義されているように、ドメイン内のどのホストもリレーが許可されます。

relay_hosts_only

デフォルトでは、アクセス・データベースとクラス「R」($=R) にリストされている名前はドメイン名であり、ホスト名ではありません。たとえば、「foo.com」を指定すると、foo.com、abc.foo.com、a.very.deep.domain.foo.com が送受信するメールはすべてリレーを許可されます。この機能によって、個別のホスト名だけを検索する方法が変わります。

relay_based_on_MX

受信者のホスト部分の MX レコードに基づいて送られてくるメールのリレーを行います。つまり、ホスト foo.com の MX レコードが示しているサイトは、foo.com に宛てられたメールを受入れ、リレーします。

メモ：ルート・アドレス構文（あるいは %-hack 構文）を使用すると、FEATURE(`relay_based_on_MX') コマンドがこれらのメッセージのルーティングを許可しない場合があります。このことが問題になる場合、アクセス・テーブルにエントリを追加するか FEATURE(`loose_relay_check') を使用してください。

---

relay_local_from

メール送信元のドメイン部分がローカル・ホストである場合にリレーを許可します。この機能を使うとスパム・メールが入り込む隙を与えるため、どうしても必要な場合にのみ使用してください。メールは直接であれルーティングされたアドレスからであれ、あなたのドメインからだと偽ってあなたのメール・サーバーに送られてくることがあります。さらに、このメールをインターネット上にある不特定のホストにリレーしてしまうことがあります。

accept_unqualified_senders

不完全な送信元アドレスを許可します。通常、ネットワーク接続において送信元アドレスにドメイン名がない場合、SMTP セッションにある MAIL FROM: コマンドは拒否されます。不完全なアドレスを使用してローカル・メールを送信する設定になっている場合（たとえば、. MAIL FROM: <joe>）、この機能を使用する必要があります。

accept_unresolvable_domains

解決できないドメインもすべて受入れます。通常、MAIL FROM: への引数のホスト部分が DNS などのホスト名サービスで見つからない場合、SMTP セッ

ションにある MAIL FROM: コマンドは拒否されます。ファイアウォールの内側にいてインターネットのホスト名スペースの表示に限界がある場合に問題となることがありますが、そのような場合にこの機能を使用できます。

access_db
: アクセス・データベース機能をオンにします。access_db 機能によって、管理者は特定のドメインからのメールを許可したり拒否したりできるようになります。デフォルトでは、アクセス・データベースの仕様は以下のとおりです。

```
hash -o /etc/mail/access
```

アクセス・データベースの形式については、`/usr/lib/sendmail.cf_m4/README` ファイルに説明があります。

blacklist_recipients
: 特定の送信先ユーザ名、ホスト名、アドレスに対して送信されてくるメールをブロックすることができます。たとえば、nobody というユーザに送られてくるメールをブロックしたり、ホスト foo.mydomain.com や guest@bar.mydomain.com へのメールをブロックしたりできます。これらの仕様はアクセス・データベースに入力します。アクセス・データベースの形式については、`/usr/lib/sendmail.cf_m4/README` ファイルに説明があります。

rbl
: Realtime Blackhole List にあるホストを拒否します。引数がある場合、それは接続するネーム・サーバを表します。引数がない場合、rbl.maps.vix.com のメイン RBL サーバを使用します。詳細については、http://maps.vix.com/rbl/ を参照してください。

loose_relay_check
: 通常、送信先アドレスに % が使用されていて（たとえば、user%site@othersite）、別のサイトがクラス「R」にある場合、check_rcpt ruleset は @othersite を取除き、user@site を再度チェックしてリレーします。この機能はこの通常の動作を変更します。ほとんどのセットアップではこの機能は必要ありません。

percpu_QueueLA、percpu_RefuseLA、percpu
: これらの機能は、QueueLA および RefuseLA の設定が、基数にシステム内にある CPU の数を掛けたものになるようにします。引数は掛ける値です。以下はその例です。

```
FEATURE(`percpu_QueueLA', 10)dnl
```

smart_host_domain
: SMART_HOST パラメータが定義されている場合に、一定のドメインへのダイレクト送信を許可します。この機能はファイアウォールの内側で有効です。引数内のドメインはすべてローカルに接続されていると見なされます。引数がない場合、ローカル・ドメインと見なされます。

mailnews
: メールが usenet ニュースグループに転送されるようにします。mailnews FEATURE で指定されているグループに宛てられたメールは、group.USENET に変換され、USENET_MAILER 経由でローカル・アドレスに送信されるか、USENET_RELAY 経由で転送されます。

sendmail の管理

sendmail 環境の管理方法は、次の節で説明します。

- 「sendmail デーモンの起動」（270 ページ）
- 「sendmail メール待ち行列の表示」（271 ページ）
- 「sendmail メール待ち行列の強制処理」（271 ページ）
- 「.forward ファイルによるメールの転送」（272 ページ）

sendmail 環境の管理についての詳細は、付録 B「IRIX sendmail リファレンス」を参照してください。

sendmail デーモンの起動

`sendmail.cf` ファイルを編集して `aliases` データベースを変更した後、sendmail を起動できます。

IRIX は、ステーション起動時にシェル・スクリプト `/etc/init.d/mail` を使って、sendmail を自動的に起動します。しかし sendmail を設定してテストするときにステーションを再起動しなくても、`/etc/init.d/mail` スクリプトを手作業で実行できます。この場合、通常 `mail` スクリプトを使用して sendmail の停止と起動を行います。このスクリプトは、sendmail に関連するファイルとプログラムを正しい順序で処理しチェックします。

sendmail デーモンを起動します。

```
/etc/init.d/mail start
```

sendmail を停止する必要がある場合は、次のコマンドを実行します。

```
/etc/init.d/mail stop
```

sendmail メール待ち行列の表示

`mailq` コマンドを実行するか、または `sendmail` コマンドで **`-bp`** オプションを指定すると、待ち行列のリストが表示されます。このリストには、待ち行列の ID、各メッセージのサイズ、メッセージが待ち行列に登録された日付、および送信者と受信者が示されます。

sendmail メール待ち行列の強制処理

`-q` フラグに値を指定しないと、sendmail が待ち行列を強制処理します。待ち行列を手作業で処理する場合は、次に示すように **`-v`** (verbose) フラグを組合わせると便利です。

```
/usr/lib/sendmail -q -v
```

冗長モードでは、sendmail がほかのステーションとの SMTP 通信のほかに、メールの配信エラーや最終的なメッセージの置き場所を示すメッセージを表示します。

ロッキング・アルゴリズムにより、1 つのジョブが待ち行列全体を凍結することはありません。ただし、非協力的な受信ステーションや、返答のないプログラム受信ステーションに対する送受信では、ステーションの資源を大量に消費してしまう可能性があります。sendmail が使用する SMTP プロトコルに則してこのような問題を解決する適当な方法は、残念ながらありません。

主要ステーションが数日ダウンすると、膨大な数の待ち行列が生成されてしまうことがあります。その結果、sendmail が待ち行列のソートに多大な時間を費やすことになってしまいます。このような場合には、管理者が待ち行列を一時的な場所に移して新しい待ち行列を作成します。古い待ち行列は、ダウンしていたステーションが復旧した後に実行します。

まず、次のコマンドで待ち行列のディレクトリ全体を移動します。メールの待ち行列は `root` が所有し、`mail` グループに属します。

1. `/var/spool` ディレクトリに移動します。

   `cd /var/spool`

2. 古いメールの待ち行列に別の名前を指定します。

   `mv mqueue omqueue`

3. 新しいメールの待ち行列のためのディレクトリを作成します。

   `mkdir mqueue`

4. root のみがこのディレクトリを所有するようにパーミッションを変更します。

   `chmod 755 mqueue`

次に、既存の sendmail デーモンは古い待ち行列のディレクトリを処理しているため強制終了し、新しいデーモンを作成します。

1. 現在実行されている sendmail デーモンを停止します。

   `/etc/init.d/mail stop`

2. 新しい sendmail デーモンを起動します。

   `/etc/init.d/mail start`

3. 古いメールの待ち行列を実行するには、次のコマンドを実行します。

   `/usr/lib/sendmail -oQ /var/spool/omqueue -q`

   `-oQ` フラグは代替の待ち行列ディレクトリを指定し、**`-q`** フラグは sendmail に待ち行列にあるすべてのジョブを一度実行させてから戻します。状況を確認するには **`-v`** フラグを指定します。古いメール待ち行列は何回も実行しないと、すべてのメッセージを配信できない場合があります。

4. 待ち行列が空になったら、このディレクトリを次のコマンドで削除します。

   `rmdir /var/spool/omqueue`

.forward ファイルによるメールの転送

aliases データベースの代わりに `.forward` という名前でファイルをユーザのホーム・ディレクトリに置くことができます。`.forward` ファイルがホーム・ディレクトリに存在していると、sendmail がそのユーザ宛のメールをファイルで指定されている受信者リストに配信します。受信者のアドレスはそれぞれコンマまたは改行によって区切られています。たとえば、ユーザ jane

のホーム・ディレクトリにある .forward ファイルに次の内容が記述されている場合、jane 宛に到着したすべてのメールは指定されたアカウントに配信されます。

```
zippy@state.edu
bongo@widgets.com
```

また、.forward ファイルでメールをファイルまたはプログラムに配信することもできます。次の内容が記述された .forward ファイルは、すべての受信メッセージを jd@company.com に配信し、メッセージの内容を /var/tmp/mail.log ファイルに追加し、同じメッセージを */usr/bin/mymailer* プログラムの *stdin* に送ります。

```
jd@company.com
/var/tmp/mail.log
| /usr/bin/mymailer
```

一般的にファイル型の受信者の場合、だれもが書込みをできるようにしておきます。ただし、sendmail を root として実行していて、ファイルに setuid または setgid ビット・セットがある場合、メッセージはファイルに書込まれます。

ユーザは、メールをほかの相手に送るだけでなく、自分自身にも転送できます。これは、各受信メッセージをほかの配信先に送り、同時に自分のメールボックスでもメールを受信したい場合に特に有用です。たとえば、ユーザ john のホーム・ディレクトリに次の内容が記述された .forward ファイルがあるとします。

```
\john, |/usr/sbin/vacation
```

sendmail プログラムは次のように動作します。

- 受信メッセージをそれぞれ john の正規のメールボックスに送ります。名前の前のバックスラッシュ（\）は、これ以上エリアスが展開されないことを示します。
- 各メッセージのコピーを /usr/sbin/vacation プログラムの stdin に送ります。垂直バー [|] は UNIX の標準的なパイプ記号です。

sendmail の MX レコード

MX レコードは BIND データベースにあるリソース・レコードです。各レコードには、配信先のステーション名、設定レベル、および配信先のステーション宛のメールを処理するエクスチェンジャ・ステーションの名前が記述されています。エクスチェンジャ・ステーションは配信先のステーション自体でもかまいません。

BIND データベースには、配信先の各ステーションに関する MX レコードを記述できます。設定レベルが最も低いレコードから処理されます。

MX レコードはメールを代替ステーションに送れるようにします。MX レコードを使用すると、sendmail 設定ファイルから静的経路を排除できます。

IPC 型（メーラ定義行の P=[IPC]）のメーラにより接続されている配信先の各ステーションに対して、sendmail は DNS データベースを検索して配信先のステーションに関連する MX レコードを調べます。MX レコードの問い合わせが正しく行われると、RFC 974 で規定され、RFC 1123 で要求されているように、返された MX レコードから検出される適切なエクスチェンジャー・ステーションを経由してメールはルーティングされます。

その結果、現行のバージョンの sendmail は旧バージョンとは異なる方法で、MX レコードを利用できるステーションに宛てられたメールをルーティングします。メーラ定義については、付録 B「IRIX sendmail リファレンス」を参照してください。

BIND 標準リソース・レコードの形式

Berkeley インターネット・ネーム・ドメイン（BIND: Berkeley Internet Name Domain）は、ネーム・サーバのデータ・ファイルに特定のレコード形式を使用しています。ここでは、BIND 標準リソース・レコードの形式について、レコードのタイプ別に詳しく説明します。

- 「BIND リソース・レコードの MX（Mail Exchanger：メール・エクスチェンジャ）の指定」（284 ページ）
- 「BIND リソース・レコードの RP（Responsible Person：責任者）の指定」（284 ページ）
- 「BIND リソース・レコードの TXT（Text：テキスト）の指定」（285 ページ）

BIND 標準リソース・レコードの形式

ネーム・サーバのデータ・ファイル内のレコードをリソース・レコード（*RR: resource records*）と呼びます。標準リソース・レコードの形式は、RFC 1035 で定義されています。これは、次のような形式になります。

```
{name} {ttl} addr-class Record Type Record-specific data
```

- 第 1 フィールドはドメイン・レコード名です。このフィールドは、必ず第 1 カラムから開始します。レコードによっては、このフィールドが空白になる場合もあります。この場合は、直前のレコード名が使われます。
- 第 2 フィールドはオプションの有効期間（TTL: time-to-live）フィールドです。これは、データベースにデータを格納しておく期間を指定します。このフィールドが空白の場合は、後述する権限の開始（SOA: Start of Authority）レコードに指定した TTL がデフォルトとして用いられます。
- 第 3 フィールドはアドレス・クラスです。現バージョンでは、*IN* クラス（インターネットのホストとアドレス）しか認識されません。
- 第 4 フィールドはリソース・レコードのタイプです。
- 第 5 フィールド以降は、リソース・レコードのタイプによって異なります。

名前フィールドとデータ・フィールドで使われている大文字／小文字は、ネーム・サーバにロードされるときにそのまま保持されます。ただし、ネーム・サーバのデータベースで比較や検索を行う場合は、大文字／小文字は区別されません。

BIND リソース・レコードの TTL

レコードの TTL を指定する場合は、適切な値を設定することが大切です。TTL は、リゾルバがサーバから取得したデータを、サーバに再要求せずに使用してよい時間を秒単位で表したものです。値を低く設定すると、繰返し発生する再要求によってサーバの負荷が大きくなります。値を高く設定すると、変更した情報が適当な時間内に配布されません。

ほとんどのホスト情報は長期間変更されません。TTL の設定方法として、普段は高い値を設定しておき、レコードが変更される場合にだけ低い値を設定することです。通常 TTL は 1 日（86400）から 1 週間（604800）の間で設定できます。あるレコードが近い将来に変更される場合は、そのレコードの TTL を低い値、つまり、およそ 1 時間（3600）から 1 日の間で設定します。変更が行われたら元の値に戻します。名前、クラス、およびタイプが同じレコードは、すべて同じ TTL 値を持つ必要があります。

BIND リソース・レコードの特殊文字

次の文字は、レコード内で特別な意味を持っています。

(空白)	名前フィールドが空白またはタブ文字の場合は、直前のレコードと同じ名前が用いられます。
@	名前フィールドが単一のアット・マーク（@）の場合は、その時点のオリジンを示します。
.	名前フィールドが単一のピリオドの場合は、ルート・ドメイン名を示します。
\x	バックスラッシュは、文字 x が持つ特別な意味を取消します。ここでは、x は数字（0 ～ 9）以外の任意の文字を表します。たとえば、ラベル内にドット文字を入れるには、\. とします。
\DDD	D はすべて数字になります。完全な文字列は、DDD で指定された 10 進数に対応する 8 進数になります。8 進数はテキスト文字であるとみなされ、特別な意味を持つ文字とは解釈されません。
( )	かっこは、複数行にまたがる、グループ化されたデータを囲みます。改行はかっこ内では無視されます。この表記は、SOA レコードと WKS レコードに有効です。

; コメントの前に置かれます。セミコロン以降はコメントとして解釈されます。

* アスタリスクはワイルドカード文字です。

通常、名前がピリオド（.）で終了していない場合には、リソース・レコードに現在のオリジンが追加されます。この方法は、ワークステーション名などのデータに現在のドメイン名を追加するのに便利ですが、追加しない場合には問題が発生することがあります。名前が現在のオリジンで示されるドメイン内のものではない場合、ピリオドで名前を終了します。ただし、インターネット・アドレスにピリオドを付けてはなりません。

BIND リソース・レコードの $INCLUDE の指定

インクルード行は、第 1 カラムから $INCLUDE という文字列で開始し、ファイル名がそれに続きます。これは、データのタイプに合わせて複数のファイルを使用できるようにするものです。たとえば、次のように指定します。

```
$INCLUDE /usr/etc/named.d/mailboxes
```

この行は、ファイル `/usr/etc/named.d/mailboxes` をロードすることを要求しています。$INCLUDE コマンドにより、別のゾーンやツリーにデータがロードされることはありません。これは、指定したゾーンのデータを別々のファイルで構成します。たとえば、この仕組みにより、メールボックス・データをホスト・データとは別のファイルで保守することが可能になります。

BIND リソース・レコードの $ORIGIN の指定

$ORIGIN は、データ・ファイルのオリジンを変更します。これは、第 1 カラムから開始し、ドメイン・オリジンがそれに続きます。これは、データ・ファイルに複数のドメインを記述するのに便利です。

```
$ORIGIN Berkeley.EDU.
```

BIND リソース・レコードの SOA（Start of Authority：権限の開始）の指定

```
name {ttl} addr-class SOA Source                Person-in-charge
@          IN          SOA ucbvax.Berkeley.EDU  kjd.ucbvax.Berkeley.EDU.
                       (
                       1994021501;Serial
                       10800   ;Refresh
                       3600    ;Retry
                       3600000 ;Expire
                       86400   ;Minimum
                       )
```

SOA レコードはゾーンの開始を示します。1つのゾーンに対して1つのSOAしか指定できません。

name はゾーン名です。ここには、Berkeley.EDU. などのフル・ドメイン名、または現時点での $ORIGIN に相対的な名前を指定します。アット・マーク（@）は現時点でのゾーン名を示し、`named.boot`ファイルのprimary行、または前の$ORIGIN行で指定されたものが用いられます。

`Source` はマスター・データ・ファイル存在するホストの名前であり、一般にはプライマリ・マスター・サーバになります。

`Person-in-charge` は、ネーム・サーバの責任者のメール・アドレスです。メール・アドレスはドメイン名の形式で表し、ホストとユーザ名を分けるアット・マーク（@）をピリオド（.）に置換えます。上の例では、kjd.ucbvax.berkeley.edu は kjd@ucbvax.berkeley.edu を置換えたものです。

`Serial` はデータ・ファイルのバージョン番号であり、データが変更されるたびに増分する必要があります。浮動小数点数（1.1 などの小数点以下がある数値）は使用できません。この値には、現在の日付をシリアル番号として用いるとよいでしょう。たとえば、*25 April 1994 edit #1* は次のように表します。

```
1994042501
```

1 日のうち何度もファイルを変更する場合は、編集番号を増分します。

`Refresh` は、セカンダリ・ネーム・サーバの更新が必要かどうかをプライマリ・ネーム・サーバに問い合せる間隔を秒単位で示します。

`Retry` は、セカンダリ・ネーム・サーバがリフレッシュの要求に対する応答を得られなかった場合に、再び要求を出すまでの時間を秒単位で示します。

`Expire` は、セカンダリ・ネーム・サーバがリフレッシュされずに、データを使用できる期間を秒単位で示します。

`Minimum` は、TTL を明示的に指定しなかった場合に、レコードの TTL フィールドで使用されるデフォルトの秒数を示します。

BIND リソース・レコードの NS（NameServer：ネーム・サーバ）の指定

```
{name}  {ttl}  addr-class  NS  Name server's_name
               IN          NS  ucbarpa.Berkeley.EDU.
```

NS レコードには、特定のドメインに対してドメイン・ネーム・サービスを提供するマシン名がリストされています。リソース名はドメイン名であり、データ部はこのドメインに関する情報を提供するホスト名です。ネーム・サービスを提供するワークステーションは、指定のドメインになくてもかまいません。ドメインのマスター・サーバ（プライマリまたはセカンダリ）ごとに 1 個の NS レコードが必要です。1 ゾーンに約 10 個から 15 個の NS レコードを使用すると、DNS データグラムのサイズの限界を超えてしまうことがあります。

ドメインの NS レコードは、そのドメインの代用ゾーンとそのドメイン自体のゾーンの両方に存在する必要があります。あるドメインに対するネーム・サーバ・ホストがそのドメイン自身の内部にある場合は、グルー・レコード（glue record）が必要です。グルー・レコードとは、サーバのアドレスを指定するアドレス（A）レコードのことです。グルー・レコードは、ドメインの代用サーバにだけ必要であり、ドメイン自体には必要ありません。たとえば、ドメイン SRI.COM のネーム・サーバが KL.SRI.COM の場合、NS レコードとグルー A レコードは次のようになります。

```
SRI.COM.     IN   NS      KL.SRI.COM.
KL.SRI.COM.  IN   A       10.1.0.2
```

代用ドメインと被代用ドメインの管理者は、NS レコードとグルー A レコードに矛盾がないように確認します。

BIND リソース・レコードの A（Address：アドレス）の指定

```
{name}    {ttl}    addr-class   A      address
ucbvax             IN           A      128.32.133.1
                   IN           A      128.32.130.12
```

A レコードには、ワークステーションのアドレスがリストされています。name フィールドはワークステーション名、address フィールドはネットワーク・アドレスです。ワークステーションのアドレスごとに 1 個の A レコードが必要です。

BIND リソース・レコードの HINFO（Host Information：ホスト情報）の指定

```
{name}  {ttl}  addr-class HINFO  Hardware         OS
               IN         HINFO  SGI-IRIS-INDY    IRIX
```

HINFO レコードは、ホスト固有のデータです。このレコードには、ハードウェアとホスト上で動作するオペレーティング・システムがリストされています。ハードウェアの情報とオペレーティング・システムの情報は、1 文字分のスペースで区切られます。ワークステーション名にスペースを挿入するには、名前を引用符で囲みます。ホストごとに 1 個の HINFO レコードが必要です。IRIS 4D シリーズのワークステーションやサーバの名前の現行リストは、`/usr/etc/named.d/README` にあります。ほかのハードウェアやオペレーティング・システムの名前は、最新の Assigned Numbers RFC（現在は RFC 1340）を参照してください。

BIND リソース・レコードの WKS（Well-Known Services：周知のサービス）の指定

```
{name} {ttl} addr-class WKS address      protocol services list
             IN         WKS 192.12.6.16  UDP   (who route
                                                timed domain)
             IN         WKS 192.12.63.16 TCP   (echo telnet
                                                chargen ftp
                                                smtp time
                                                domain bootp
                                                finger sunrpc)
```

WKS レコードには、指定したアドレスでサービスされるプロトコルがリストされています。リストはサービスとポート番号から構成され、これは `/etc/services` で指定したサービス・リストを使用しています。WKS はアドレスごと、プロトコルごとに 1 個しか使用できません。

BIND リソース・レコードの CNAME（Canonical Name：正式名）の指定

```
aliases  {ttl}  addr-class  CNAME  Canonical name
ucbmonet        IN          CNAME  monet
```

CNAME レコードは、ホストの正式名のエリアスを指定します。1 つのエリアスに対してレコードが 1 個必要です。ほかのすべてのレコードは正式名に対応付けられ、エリアスには対応付けられません。ドメイン名を値として含むすべてのレコード（NS や MX など）は、エリアスではなく正式名で表します。

エリアスは、ホスト名を変更する場合にも有用です。つまり、CNAME レコードを使用すると、古い名前を使用している場合でもユーザは正しいホスト名を使用できます。

BIND リソース・レコードの PTR（Domain Name Pointer：ドメイン名ポインタ）の指定

```
name  {ttl}  addr-class  PTR  real name
6.130        IN          PTR  monet.Berkeley.EDU.
```

PTR レコードは、特別な名前を使ってドメインの別の場所を指します。上の例では、特別な IN-ADDR.ARPA ドメインの逆引きポインタを設定するのに PTR レコードを使用しています。PTR 名はゾーンに対して一意です。実際の名前に付加されているピリオド（.）は、named が現在のドメイン名を付加させないようにするためのものです。

BIND リソース・レコードの MB（Mailbox：メールボックス）の指定

```
name  {ttl}  addr-class  MB  Machine
ben          IN          MB  franklin.Berkeley.EDU.
```

MB レコードには、ユーザがメールを受信するワークステーションがリストされています。name フィールドはユーザのログイン名です。machine フィールドは、メールの配信先であるワークステーションです。メールボックス名は、ゾーンに対して一意にします。

BIND リソース・レコードの MR（Mail Rename Name：メール名変更）の指定

```
name {ttl} addr-class MR  corresponding_MB
Postmaster IN         MR  ben
```

MR レコードには、ユーザのエリアスがリストされています。name フィールドは、最後のフィールドにある名前のエリアスです。この名前には、それに対応する MB レコードが必要です。

BIND リソース・レコードの MINFO（Mail Information：メール情報）の指定

```
name  {ttl}  addr-class MINFO  requests      maintainer
BIND         IN         MINFO  BIND-REQUEST  kjd.Berkeley.EDU
```

MINFO レコードは、メール・リストのメール・グループを作成します。このレコードは、通常はメール・グループ（MG）レコードと対応していますが、メールボックス（MB）レコードと併せて使用することもできます。name フィールドは、メールボックス名です。requests フィールドは、メール・グループに追加する要求などのメールの配信先を示します。maintainer は、エラー・メッセージを受取るメールボックスです。これは、メンバー名のエラーを送信者以外のだれかに報告する場合に有用です。

BIND リソース・レコードの MG（Mail Group Member：メール・グループ・メンバー）の指定

```
{mail group name}  {ttl}    addr-class  MG  member name
                            IN          MG  Bloom
```

MG レコードには、メール・グループのメンバーがリストされています。次に示すのは、メール・リストの一例です。

```
Bind  IN  MINFO    Bind-Request        kjd.Berkeley.EDU.
      IN  MG       Ralph.Berkeley.EDU.
      IN  MG       Zhou.Berkeley.EDU.
```

BIND リソース・レコードの MX（Mail Exchanger：メール・エクスチェンジャ）の指定

```
                          preference  mail
name {ttl}      addr-class MX  value       exchanger
Munnari.OZ.AU. IN         MX  10          Seismo.CSS.GOV.
*.IL.          IN         MX  10          CUNYVM.CUNY.EDU.
```

MX レコードは、ネットワークには直接接続されていないワークステーションにメールを配信するワークステーションを指定します。最初の例では、Seismo.CSS.GOV がメールを Munnari.OZ.AU に配信するメール・ゲートウェイです。ネットワーク上にあるほかのシステムでは、メールを直接 Munnari に送ることはできません。Seismo と Munnari にはプライベートな接続を行うか、または別の伝送媒体を使用します。preference value は、同じワークステーションにメールを配信する経路が複数ある場合に、メーラが実行すべき順番を指定します。詳細については、RFC 974 を参照してください。

MX レコードでメールをルーティングする際に、アスタリスク（*）を含むワイルド・カード名を使用できます。ネットワーク上のサーバは、特定のドメインへのメールをリレー・ステーションを介して配信するように指示できます。2 番目の例では、ドメイン IL にあるホストへのすべてのメールは、CUNYVM.CUNY.EDU を経由して配信されます。これはワイルド・カードを使った MX レコードであり、*.IL が CUNYVM.CUNY.EDU. の MX を持っていることを意味します。

BIND リソース・レコードの RP（Responsible Person：責任者）の指定

```
owner {ttl} addr RP mbox_domain_name        TXT_domain_name
franklin    IN   RP franklin.berkeley.edu  admin.berkeley.edu.
```

RP レコードは、ホストの責任者の名前またはグループ名を示します。これは、特定のホストの責任者を識別するのに使用します。このレコードを指定しないと、ホストがダウンしたり正しく動作しなくなった場合に、問題の解決やホストの修理の責任者がわからなくなります。

mbox_domain_name フィールドは、責任者のメールボックスを指定するドメイン名です。マスター・ファイルのメールボックスの形式は、メールボックスのコード化に DNS 方式を用いており、これは SOA レコードの Person-in-charge メールボックス・フィールドに使用されているものと同じです。上の例で、mbox_domain_name は ben@franklin.berkeley.edu をコード化した

ものです。ルート・ドメイン名 “.” を指定すれば、利用できるメールボックスがないことを示すこともできます。

最後のフィールドは、TXT RR が存在するドメイン名です。次の問い合わせを行うと、TXT ドメイン名で対応する TXT リソース・レコードを検索できます。TXT レコードを検索すると、DNS の複数の場所から責任者を参照できる経路がわかります。TXT ドメイン名にルート・ドメイン名 “.” を指定すれば、対応する TXT RR が存在しないことを示すこともできます。上の例では、sysadmins.berkeley.edu は TXT レコードの名前であり、名前と電話番号の入ったテキストを入れることもできます。

RP レコードの形式は、クラスに依存しません。したがって、同じ名前の RP レコードが複数のデータベースに存在することもあります。これらのレコードは、同じ TTL を持ちます。

RP レコードは試験的なレコードであり、すべての DNS サーバがそれを実装したり、認識するわけではありません。

BIND リソース・レコードの TXT（Text：テキスト）の指定

```
text-name   {ttl}   addr-class   TXT  text-data
location            IN           TXT  "Berkeley, CA"
```

TXT レコードには、テキストが入ります。テキストの意味は、それが存在するドメインによって異なります。

IRIX sendmail リファレンス

この付録は、`sendmail` に関する参考情報です。この付録では、以下について説明します。

sendmail コマンド行フラグ

コマンド行でフラグを指定すると、sendmail の動作を設定できます。ここでは、頻繁に使用するフラグについて説明します。

コマンド行で指定するフラグの詳細については、297 ページの「sendmail のフラグ、オプションおよびファイル」を参照してください。

sendmail 設定オプション値の変更

-o フラグは設定ファイルの設定内容をオーバーライドします。オーバーライドは、現在のセッションに対してのみ有効です。次の例では、そのセッションにかぎり **T**（タイムアウト）オプションを 2 分間に設定しています。

```
/usr/lib/sendmail -oT2m
```

設定オプションの詳細については、297 ページの「sendmail のフラグ、オプションおよびファイル」を参照してください。

sendmail 配信モードの指定

コマンド行で頻繁に変更されるオプションに `sendmail` の配信モードを指定する **d** オプションがあります。配信モードの設定によって、メールの配信速度を変更することができます。

i　対話形式（同期）で配信します。

b　バックグラウンド（非同期）で配信します。

q　待ち行列に登録するだけで、配信は行いません。

オプションにはトレードオフがあります。**i** モードでは最大量の情報を送信者に渡せますが、これが必要となることはほとんどありません。

q モードではステーションにかかる負荷が最小限になりますが、待ち行列の処理間隔だけ配信が遅れる場合があります。

b モードは適当な配信方法といえます。ただし、このモードでは、メッセージの配信に時間がかかるメーラを使用していると `sendmail` が多くのプロセスを生成してしまう場合があります。

sendmail 待ち行列モードの指定

-q フラグは `sendmail` に一定間隔でメールの待ち行列を処理するように指示します。このオプションは、次の構文で記述します。*time* には待ち行列処理の間隔を指定します。

-q [*time*]

時間は分単位で指定します。たとえば、15m は間隔を 15 分に設定します。*time* が省略されると、*sendmail* は待ち行列を 1 回処理して戻ります。通常、**-q** フラグは次に説明するデーモン・モードと合わせて使います。

処理間隔の指定方法と形式については、291 ページの「sendmail のタイムアウトと処理間隔の省略文字」を参照してください。

sendmail デーモン・モードの指定

ソケットで受信するメールを処理するには、デーモンを実行する必要があります。**-bd** フラグを指定すると、*sendmail* がデーモン・モードで動作します。次に示すように、**-bd** と **-q** フラグを一緒に指定することができます。

```
/usr/lib/sendmail -bd -q30m
```

このコマンドを実行すると、sendmail がデーモン・モードで動作し、サブデーモンをフォークして 30 分おきに待ち行列を処理します。

IRIX のスクリプトでは、次のコマンド行で sendmail を起動します。

```
/usr/lib/sendmail -bd -q15m
```

sendmail 検証モードの指定

-bv フラグを指定すると、sendmail がアドレス、エリアス、およびメール・リストの妥当性を検証します。このモードでは、sendmail は妥当性だけを検証し、メッセージの収集や配信は行いません。sendmail はすべてのエリアスを展開し、重複を削除し、展開された名前のリストを表示します。また、sendmail は個々の名前に対してメッセージを実際にその配信先に送れるかどうかを検証します。

sendmail テスト・モードの指定

-bt フラグを指定すると、sendmail がテスト・モードになり、現在の設定ではアドレスがどのように書換えられるのかを示します。テスト・モードは、`/usr/lib/sendmail.cf` 設定ファイルに加えた変更内容をデバッグするのに特に有用です。

sendmail デバッグ・フラグの指定

sendmail にはいくつかのデバッグ用フラグが組込まれています。フラグごとに番号とレベルがあり、番号はデバッグ・フラグを示し、レベル（デフォルトは 1）は表示する情報量を示します。低レベルは最小限の情報を表示し、高レベルはより広範囲な情報を表示します。通常、9 を超えるレベルはお薦めしません。それほど多くの情報を表示してもあまり意味がないからです。デバッグ・フラグの構文は、次のとおりです。

-d *debug-list*

デバッグ・リストには、次の例に示すようにフラグ番号とフラグ・レベルが記述されています。

- フラグ番号 13 をレベル 1 に設定します。

  -d13

- フラグ番号 13 をレベル 3 に設定します。

  -d13.3

- フラグ番号 5 から 18 までをレベル 1 に設定します。

  -d5-18

- フラグ番号 5 から 18 までをレベル 4 に設定します。

  -d5-18.4

通常の *sendmail* ユーザがデバッグ・フラグを使用することはあまりありません。しかし、このオプションには、原因のよく分からない問題を調べるのに有用なものもあります。デバッグ・オプションのリストについては、297 ページの「sendmail のフラグ、オプションおよびファイル」を参照してください。

sendmail の設定の変更

設定パラメータを編集すれば、sendmail をサイトの要求に合わせて設定できます。これらの設定パラメータは、設定ファイルのオプションで設定します。たとえば、文字列 **T3d** は、**T**（タイムアウト）オプションを **3d**（3 日）に設定します。処理間隔の指定については、291 ページの「sendmail のタイムアウトと処理間隔の省略文字」を参照してください。

大半のオプションにはデフォルト値があり、それらを変更しなくても十分サイトの要求を満たすことができます。しかし、メールの負荷が重いサイトでは、その負荷に対応できるようにパラメータを変更する必要があります。特に、短いメッセージを大量に複数の受信者に送信するサイトでは、待ち行列の優先順位のパラメータを変更します。

次に、設定パラメータについて説明します。

- 「sendmail のタイムアウトと処理間隔の省略文字」（291 ページ）
- 「sendmail メール待ち行列の処理間隔設定」（292 ページ）
- 「sendmail 読込みのタイムアウト設定」（292 ページ）
- 「sendmail 待ち行列メッセージのタイムアウト設定」（293 ページ）
- 「sendmail 待ち行列実行中のフォーク」（294 ページ）
- 「sendmail 待ち行列の優先順位」（294 ページ）
- 「sendmail 負荷の最大値」（295 ページ）
- 「sendmail ログ・レベル」（295 ページ）

sendmail のタイムアウトと処理間隔の省略文字

処理間隔は、次の省略文字で指定します。

s	秒
m	分
h	時間
d	日数
w	週

たとえば、10m は 10 分を表し、2h30m は 2 時間 30 分を表します。

sendmail メール待ち行列の処理間隔設定

-q フラグの引数は、メール待ち行列の処理間隔を指定します。sendmail デーモンを `/etc/init.d/mail` スクリプトで起動した場合、待ち行列の処理間隔は 15 分に設定されています。

sendmail が配信モード **b** で動作しており、受信ホストがダウンしているときなどメッセージが配信されなかった場合のみ、そのメッセージが待ち行列に追加されます。このとき、待ち行列を処理する必要性は低いので、処理間隔を長く設定できます。この値は、ダウンしていたホストが回復した場合のみ重要になります。

sendmail が送信モード **q** で動作している場合、待ち行列の処理間隔は短く設定します。この値が、処理するまでメッセージをローカルの待ち行列に入れておく最長時間を指定します。

sendmail 読込みのタイムアウト設定

sendmail プログラムが標準入力を読込んでいるときやリモートの SMTP サーバから情報を読込んでいるときに、タイムアウトになる場合があります。技術的には、発行済みのプロトコル内でタイムアウトは有効ではありません。ただし、読込みのタイムアウト・オプションに大きな値（たとえば 1 時間）を設定すると、アイドル状態のデーモンが大量にシステム上に存在することがなくなります。このオプションは、次の形式で指定します。

r*timeout.suboption=value*

timeout は、291 ページの「sendmail のタイムアウトと処理間隔の省略文字」で説明されているタイムアウトの形式に従って起動中の `sendmail` で指定します。次では、読込みタイムアウトについてさらに詳しく説明します。

表 B-1 sendmail 読込みタイムアウトのサブオプション

サブオプション	説明
command	次のコマンドを待ちます。
connect	connect(2) が戻るのを待ちます。
datablock	各データ・ブロックを読むのを待ちます。
datafinal	最終ドットの認知を待ちます。

表 B-1 sendmail 読込みタイムアウトのサブオプション（続き）

サブオプション	説明
datainit	データの認知を待ちます。
fileopen	NFS ファイルが開くのを待ちます。
helo	HELO または EHLO を待ちます。
hoststatus	ホスト状態の持続時間
iconnect	最初の connect(2) を待ちます。
initial	最初のグリーティング・メッセージを待ちます。
mail	MAIL の認知を待ちます。
misc	ほかの SMTP コマンドを待ちます。
queuereturn	未配信の場合はバウンスします。
queuewarn	未配信の場合は警告します。
quit	QUIT の認知を待ちます。
rcpt	RCPT の認知を待ちます。
rset	RSET の認知を待ちます。

sendmail 待ち行列メッセージのタイムアウト設定

sendmail は、指定した時間が過ぎると待ち行列のメッセージをタイムアウトします。このため、メッセージが送信されなかったことが送信者に知らされます。デフォルトのタイムアウト値は、1 週間（7 日間）です。このオプションの値は、**T** で指定します。

待ち行列ではメッセージが待ち行列に登録された時刻を記録しており、タイムアウトまでの残り時間は記録していません。したがって、待ち行列でメッセージのタイムアウトを短くすれば、短期間残っていたメッセージを sendmail がフラッシュできます。次の例は、待ち行列の処理方法と前日のすべてのメッセージをフラッシュする方法を示しています。

```
/usr/lib/sendmail -oT1d -q
```

sendmail 待ち行列実行中のフォーク

Y オプションを設定すると、待ち行列を処理する前に sendmail が個々のメッセージをフォークします。大量のメモリが消費されることがなくなるので、これはメモリの少ない環境では有用です。**Y** オプションが設定されていないと、sendmail は待ち行列の実行中にダウンしたホストを記録します。このため、パフォーマンスが大幅に向上します。

sendmail 待ち行列の優先順位

sendmail プログラムは、各メッセージがそれぞれ最初に生成されたときに優先順位を付けます。これはその優先順位とメッセージの生成時刻（1970 年 1 月 1 日以降の秒単位の時刻）に基づき、待ち行列の順序を決めます。最も低い優先順位番号を持つメッセージが最初に処理されます。このアルゴリズムでは、次の情報を基にメッセージの優先順位を決定します。

メッセージ・サイズ（バイト単位）
: 短いメッセージは長いメッセージより優先順位が低くなります。これによって、待ち行列の効率が向上します。

メッセージ・クラス
: Precedence: フィールドとその値がメッセージにある場合、*sendmail* はこの値で設定ファイルからメッセージ・クラスを選択します。一般的な値は、first-class または bulk です。

クラスの重み係数
: この係数は、設定ファイルで **z** オプションを使って設定します。デフォルト値は 1800 です。

受信者の数
: 受信者の数は、メッセージがシステムに及ぼす負荷に影響を及ぼします。受信者が 1 人しかいないメッセージは、受信者リストの多いメッセージよりも優先順位が低くなります。

受信者の重み係数
: この係数は、設定ファイルで **y** オプションを使って設定します。デフォルト値は 1000 です。

次に示すのは、優先順位のアルゴリズムです。

```
priority=message_size-(message_class * z)+(num_recipients * y)
```

sendmail はメッセージの優先順位を決めた後、次の式で待ち行列の順番を決めます。

```
ordering = priority + creation_time
```

メッセージの優先順位は、メッセージが配信されるたびに変わります。回数の重み係数（**Z** オプションで設定）は優先順位を上げる値です。これは、何回も配信に失敗したメッセージはその後も失敗する可能性が大きいという仮定に基づいています。

sendmail 負荷の最大値

sendmail はシステムのロード・アベレージが指定した最大値を超えると、メールを待ち行列に登録して配信を試みません。この最大値は *x* オプションで指定します。ロード・アベレージがこの最大値を超えると、*sendmail* は、次のアルゴリズムでメッセージの優先順位をテストします。次の例では、*q* は **q** オプションの値、*x* は **x** オプションの値です。

q `/ (load_average -` *x* `+ 1)`

最終的なロード・アベレージを計算した後、sendmail はその結果を各メッセージの優先順位と比較します。優先順位の方が大きければ、sendmail は配信モードを **q**（待ち行列に登録のみ）に設定します。

q オプションのデフォルト値は 10000 であり、1 ポイントのロード・アベレージは 10000 優先順位・ポイントに相当します。**X** オプションは、sendmail にネットワーク接続を受付けさせないロード・アベレージを指定します。ただし、ローカルに生成されたメールは、UUCP メールも含めて受付けられます。

sendmail ログ・レベル

sendmail は、エラーとイベントを記録します。設定ファイルの **L**（ログ・レベル）オプションは、どの程度詳しい情報をログに書込むかを指定します。デフォルト値は 1 です。次のレベルを指定できます。

0	ログを取りません。
1	重要な問題だけを記録します。
2	受信したメッセージと失敗した配信のログを取ります。

3	成功した配信のログを取ります。
4	ホストがダウンした場合など､配信が遅れているメッセージのログを取ります。
5	通常の待ち行列動作のログを取ります。
6	ロックされた待ち行列ファイルを処理しようとする試みなど、異常だけれども重大ではない問題のログを取ります。
9	内部待ち行列の ID と外部メッセージの ID との対応付けのログを取ります。メッセージがいくつかのホストを経由する場合などにそのメッセージをトレースするのに有用です。
12	デバッグ時に必要ないくつかのメッセージのログを取ります。
16	待ち行列に関する詳細情報のログを取ります。
20	ロックされている待ち行列ファイルを処理しようとする試みのログを取ります。
21	ロード・アベレージの計測値とプロセス数のログを取ります。

Sendmail 設定ファイル — sendmail.cf

`sendmail.cf` ファイルは、メールの動作やルーティングの方法を sendmail に伝える設定ファイルです。`sendmail.cf` ファイルを生成する場合、大半のユーザが `sendmail.mc` ファイル内のパラメータを定義し、さらに `/usr/lib/sendmail.cf_m4` ディレクトリ内のプロトタイプ・ファイルを使用してこのファイルを設定することをお勧めします。

`sendmail.cf` ファイルの修正については、Bryan Costales、Eric Allman 共著の『sendmail, 2nd Edition』（ISBN: 1-56592-222-0）に詳しい情報があります。出版社は O'Reilly & Associates Inc で、この本は次の URL から注文できます。

`http://www.oreilly.com/catalog/sendmail2`

sendmail のフラグ、オプションおよびファイル

ここでは、次の項目について説明します。

sendmail コマンド行フラグ

フラグはアドレスの前に置きます。フラグには次のものがあります。

-b*x* 操作モードを *x* に設定します。操作モードには、次のものがあります。

a	ARPANET モードで実行します。 ARPANET の特別な処理には、次のものがあります。ヘッダから From: と Sender: 行を読込み、送信者を特定します。ARPANET 形式（FTP プロトコルと互換性をとるために、メッセージの前に 3 桁の応答コードを記述）のメッセージを出力します。エラー・メッセージの行を <CRLF> で終了します。
d	デーモンとして実行します。
D	d と同じですが、フォアグラウンドで実行します。
h	ホスト状態のデータベースを出力します。
H	ホスト状態のデータベースを消去します。
i	エリアス・データベースを初期化します。
m	通常の方法でメールを配布します（デフォルト）。
p	メールの待ち行列を出力します。
s	入力側で SMTP を実行します。
t	テスト・モードで実行します。
v	メールの受信と送信は行わず、アドレスの検証のみ行います。
z	設定ファイルを凍結します。これができるのは特権ユーザだけです。

-C*file* デフォルトではない設定ファイルを使用します。このフラグを指定すると、sendmail が root としてではなく、呼出しユーザのアカウントで動作します。

-d*flag*[-*flag*][.*level*]
デバッグ・フラグまたはフラグの範囲を指定したレベルにセットします。デフォルト値は 1 です。308 ページの「sendmail のデバッグ・フラグ」を参照してください。

-F*name* 送信者のフルネームを *name* に設定します。

-f*name* *From*（メールの送信者）の名前を設定します。このフラグは、ユーザが認可ユーザ・リストに登録されていない場合、または *name* ユーザ名と同じでない場合は無視されます。

-h*cnt* ホップ・カウントを *cnt* に設定します。ホップ・カウントは、メールが処理されるたびに増加します。上限値に達すると、メールはエラー・メッセージを伴って返され、エリアス・ループに陥りません。

-i 受信メッセージ上のドットだけを無視します。

-N 配信状態の通知を設定します。

-n エリアスの設定を行いません。

-o*x value* 設定オプション *x* を指定した *value* に設定します。このオプションについては、299 ページの「sendmail の設定オプション」を参照してください。

-p*protocol* メッセージ受信プロトコルを設定します。プロトコル名だけ、または UUCP:ucbvax のように、プロトコル名とホスト名の両方を指定します。

-q[*time*] 待ち行列のメールを処理します。time を指定すると、sendmail は指定した間隔で待ち行列を調べ、メールを配信します。time を指定しないと、1 回しか待ち行列を処理しません。288 ページの「sendmail 待ち行列モードの指定」を参照してください。

-R*return* メールの配信に失敗した場合に、返信するメッセージの数を設定します。たとえば *full* を設定するとすべてのメッセージが返信され、*hdrs* を設定するとヘッダのみが返信されます。

-r *name* *-f* の古い形式であり、実行する内容は *-f* と同じです。

`-t` ヘッダの To:、Cc:、Bcc: の各行を読込み、そこにリストされている受信者全員にメッセージを送信します。Bcc: 行は送信前に削除されます。引数ベクトルのすべてのアドレスが送信リストから削除されます。

-U 最初のユーザ登録を設定します。Mail または exmh などのようなユーザ・エージェントを使用するときは、必ずこのオプションを設定します。rmail ネットワーク配信エージェントでは、絶対にこのオプションを設定しないでください。

-V*envid* オリジナルのエンベロープ ID を設定します。

`-v` 冗長モードになります。エリアスの展開情報などが表示されます。

X*logfile* *logfile* に対するメーラのすべてのトラフィックをログに記録します。データが大量に蓄積されてしまうため、デバッグが困難な場合にだけ使用してください。

-Z*file* *file* の代替の凍結ファイルです。

sendmail の設定オプション

コマンド行で *-o* フラグを指定するか、または設定ファイルに *O* 行を指定すると、次のオプションを設定できます。これらのオプションの大半は、認可ユーザでなければ指定できません。

AliasFile=*file*

A*file* *file* をエリアス・ファイルとして使用します。file を指定しないと、現在のディレクトリにあるエリアスを使用します。

AliasWait=*N*

a*N* 起動前にエリアス・データベースに @:@ エントリが現れるのを最大 *N* 分待ちます。*N* 分以内に現れない場合、データベースを再構築するか（*D* オプションも設定されている場合）、または警告を出力します。

BlankSub=*c*

B*c* 空白置換文字を *c* に設定します。アドレスにある引用符なしのスペースは、この文字に置換えられます。

MinFreeBlocks=*nblocks*

b*nblocks[/maxsize]*

許可される最大メッセージサイズを設定するときに、メッセージのスプールに必要な最小の空きブロック数を設定します。デフォルトはそれぞれ、0（ゼロ）と無限です。

CheckpointInterval=*N*

C*N*

N の配信が成功した後、待ち行列ファイルを調べます。数多くの宛先にメールを送信中にシステム・クラッシュなどで送信が中断された場合、重複して送信するのを防ぐことができます。

HoldExpensive

c

送信メーラの負荷が重いと指定されている場合は、すぐに接続しません。このオプションを指定すると、待ち行列が実行されます。

AutoRebuildAliases

D

必要に応じてエリアス・データベースを再構築します。このオプションが指定されていないと、*sendmail -bi* で明示的に指定しないかぎり、sendmail はエリアス・データベースを再構築しません。

DeliveryMode=*x*

d*x*

モード *x* で配信します。使用できるモードは次のとおりです。

- i　対話形式（同期化）で配布します。
- b　バックグラウンド（非同期化）で配布します。
- q　メッセージを待ち行列に入れ、待ち行列を実行するまでそれを配布しません。

ErrorHeader=*/file*|*format*

E*/file*|*format*

エラー・メッセージにヘッダを追加します。値を／で始めると、エラー・ヘッダの形式が指定のファイルから読込まれます。

ErrorMode=*x*

e*x*

モード *x* でエラーを処理します。*x* には次の値を指定できます。

- p　エラー・メッセージを出力します（デフォルト）。
- q　メッセージを出力せず、終了の状態だけ報告します。

m　エラー・メッセージを返します。

w　ユーザがログインしていない場合は、エラー・メッセージを出力して配布します。

e　エラー・メッセージを返し、常に終了状態をゼロにします。

TempFileMode=*mode*

F*mode*　待ち行列ファイルや凍結された設定ファイルを作成する場合に使用する UNIX ファイル・モードを設定します。

SaveFromLine

f　ヘッダの先頭にある UNIX 形式の “From” 行を保存します。通常、この行は冗長とみなされて破棄されます。

MatchGecos

G　パスワード・ファイルの GECOS 部分に対してローカル名のマッチングを行います。

DefaultUser

g*n*　実行メーラのグループ ID を *n* に設定します。

HelpFile=*file*

H*file*　SMTP のヘルプ・ファイルを指定します。

MaxHopCount=*N*

h*N*　メッセージがループに分類されるまでにホップできる最大回数を設定します。

I　BIND ネーム・サーバを実行し、ホスト名と MX レコードを検索します。リゾルバからの ECONNREFUSED エラーを一時的なエラーとして処理します。通常、このオプションはネーム・サーバを実行している場合にだけ設定します。`/etc/hosts` ファイルにすべての既知のホストがない場合、または BIND ネーム・サーバの MX（メール・フォワード）機能を使用している場合に、このオプションを設定します。このオプションを設定していなくてもネーム・サーバに問い合わせを行いますが、ネーム・サーバが利用できない場合は、sendmail は `/etc/hosts` ファイルを読込みます。

IgnoreDots

i　　行のドットがメッセージ終端記号として解釈されません。

ForwardPath=*path*

J*path*　　ユーザの `.forward` ファイルの検索に使用するパスを指定します。デフォルトは `$z.forward` ですが、複数のパスをコロンで区切って指定することもできます。

SendMimeErrors

j　　エラー・メッセージを MIME 形式で送信します。

ConnectionCacheTimeout=*timeout*

K*timeout*　　キャッシュされた接続のアイドル・タイムの最大値を指定します。*timeout* は、秒は「s」、分は「m」、時間は「h」、日数は「d」、週は「w」として指定します。たとえば、「K1h30m」と「K90m」はいずれも *timeout* を 1 時間 30 分として指定します。

ConnectionCacheSize=*N*

k*N*　　キャッシュされるオープンな接続の最大数を定義します。デフォルトは 1 です。0 を指定すると、接続はただちに閉じられます。

LogLevel=*n*

L*n*　　ログ・レベルを *n* に設定します。*n* に設定可能な値は次のとおりです。

- 0　ログを取りません。
- 1　重大なシステム・エラー、潜在的なセキュリティの問題だけを記録します。
- 2　ネットワーク・エラーとセキュリティ上の問題だけを記録します。
- 3　配信および受信メッセージの失敗を記録します。
- 4　小さなエラーを記録します。
- 5　受信メッセージの統計情報を記録します。
- 6　エラー・メッセージの生成と、VRFY および EXPN コマンドを記録します。
- 7　失敗した配信を記録します。
- 8　成功した配信を記録します。

	9	ホストがダウンしているために配信が延期されているメッセージを記録します。
	10	エリアスによる配信拡張を記録します。
	12	ホストへの接続を記録します。
	20	ロックされている待ち行列ファイルを処理しようとする試みを記録します。
	21	アベレージ・ロードの計測値を監視します。

UseErrorsTo

l	Errors-To: ヘッダがある場合に、指定アドレスにエラー・メッセージを送信します。
M*x value*	マクロ *x* を *value* に設定します。このオプションは、コマンド行でしか指定できません。

MeTo

m	送信者がエリアス展開の中にあっても、「me」（送信者）に送ります。
N*netname*	ホーム（ローカル）ネットワークの名前を設定します。接続ホスト（**gethostbyaddr()** への呼出しで決定）の名前がドメインを含むフル・アドレスではない（ドットを含んでいない）場合、接続ホストの名前に 1 個のドットと *netname* が付加されます。 その後、接続ホストが発行した SMTP の HELO コマンドの引数と上記の接続ホスト名とを比較します。この 2 つがマッチしないと、接続ホスト名に Received: 行が追加され、メッセージが正確にトレースできるようになります。

CheckAliases

n	エリアス・データベース構築時に RHS を検証します。

OldStyleHeaders

o	古い形式（スペースで名前を区切る形式）のヘッダを受付けます。このオプションは、使用できるアルゴリズムを有効にします。つまり、受信者アドレスにカンマ、丸かっこ、またはかぎかっこがある場合は、カンマがすでに存在しているものとして処理します。このオプションが設定されていない場合、カンマだけが名前を区切ります。ヘッダでは、常に名前の間にカンマが挿入されます。

PostMasterCopy=*addr*

P*addr*　「postmaster」アドレス `addr` をすべてのエラー・メッセージの Cc: リストに追加します。

PrivacyOptions=*opt,opt,...*

p*opt,opt,*　プライバシー・オプションが SMTP プロトコルに対応するよう要求します。

authwarnings	X-Authenication-Warning ヘッダを追加します。
goaway	SMTP 状態の照会を禁止します。
needexpnhelo	EXPNの前にHELOまたはEHLOコマンドを置きます。
needmailhelo	MAILの前にHELOまたはEHLOコマンドを置きます。
needvrfyhelo	VERYの前にHELOまたはEHLOコマンドを置きます。
noexpn	EXPN を全面的に禁止します。
novrfy	VRFY を全面的に禁止します。
noreceipts	Return-Receipt-To: ヘッダを無視します。
public	自由なアクセスを許可します（デフォルト）。
restrictmailq	`mailq` コマンドを制限します。

Q*dir*　dir（ディレクトリ）を待ち行列ディレクトリとして使用します。

q*factor*　メッセージを送信せずに待ち行列に登録する時期を決定するための関数の乗数として、factor（係数）を使用します。この値を、現在のロード・アベレージとロード・アベレージの限界値（*x* オプション）との差で割り、送信メッセージの優先順位の最大値を決めます。デフォルト値は 10000 です。

R　明示的にルート内の最初のアドレスにルーティングします。

Timeout.suboption=*time*

r*time*　*time* に指定した時間経過後、読込みをタイムアウトにします。

StatusFile=*file*

S*file*　指定した *file* の統計を記録します。

SuperSafe

s　実行時、安全のために即時配信を試みても必ず待ち行列ファイルにそれを登録します。*sendmail* はどのような状況でも、待ち行列ファイルに登録してから制御をクライアントに返します。

QueueTimeout=*time*

T*time*　待ち行列のタイムアウトを *time* に設定します。指定した時間が経過すると、送信に失敗したというメッセージが送信者に返されます。

TimeZoneSpec

t*tzinfo*　*PST8PDT* などのローカルな時間帯を設定します。

DefaultUser=*U* [:*G*]

u*n*　メーラのデフォルトのユーザ ID を *n* に設定します。メーラ定義に S フラグのないメーラは、このユーザとして実行されます。

FallbackMXhost=*host*

V*host*　各ホスト上で *host* が最も優先順位の低い MX のように動作するように設定します。

Verbose

v　冗長モードで実行します。

TryNullMXList

w　最良の MX の場合、ホストを直接試します。これはデフォルトで設定されています。

QueueLA=*load*

x*load*　メッセージを送信せずに待ち行列に入れるかどうかの判断をするための、システムのロード・アベレージの上限を *load* で指定します。

RefuseLA=*load*

X*load*　システムのロード・アベレージが *load* を超えたときに、受信される SMTP 接続を拒否します。

Recipientfactor=*factor*

y*factor*　受信者数にこの factor（係数）を掛け、その値を優先順位に加算します。多くの受信者を持つメッセージほど、優先順位が低くなります。

ForkEachJob

Y　別々のプロセスにある待ち行列から実行されるジョブを配信します。デフォルトの設定では、待ち行列処理時にかなりのメモリを消費するので、メモリが不足している場合にこのオプションを使用します。

ClassFactor=*factor*

z*factor*　この factor（係数）をメッセージ・クラス（ユーザ・ヘッダの Precedence: フィールド、および設定ファイルの Precedence 行で決定）に掛け、それを優先順位から引きます。高いクラスのメッセージが優先されます。

RetryFactor=*factor*

Z*factor*　メッセージが処理されるたびに、この factor（係数）を優先順位に加算します。処理頻度の高いメッセージほど、優先順位が低くなります。

DialDelay=*sleeptime*

接続に失敗すると、*sleeptime* 秒間だけスリープ状態となってから接続を再試行します。ダイヤル・オン・デマンドで接続しているサイトなどで有用です。

NoRecipientAction=*action*

受信者ヘッダ（To:、Cc:、Bcc;）がない場合の動作を、*action* で設定します。*action* に指定できる値は次のとおりです。

none	メッセージを変更しないでそのままにします。
add-to	宛先の受取人のところに To: ヘッダを追加します。
add-apparently-to	宛先の受取人のところに Apparently-To ヘッダを追加します。
add-bcc	Bcc: ヘッダを空欄で追加します。
add-to-undisclosed	「To undisclosed-recipients」というヘッダを追加します。

sendmail のサポート・ファイル

ここでは、sendmail が作成するサポート・ファイルをまとめます。

/etc/aliases.db
: Berkeley db 形式のエリアス・ファイル

/etc/aliases.{pag,dir}
: ndbm 形式のエリアス・ファイル

/etc/init.d/mail
: sendmail デーモンを起動、停止するシェル・スクリプト

/etc/sendmail.cf
: テキスト形式の設定ファイル

/etc/sendmail.hf
: SMTP ヘルプ・ファイル

/etc/sendmail.killed
: 機能していないホスト（存在していないか、何らかの理由でメールを受信できないホスト）の名前を記述したファイル

/etc/sendmail.mc
: `sendmail.cf` ファイルの設定ファイル

//usr/bin/mail.local
: sendmail が「ローカル」メーラーとして使用するプログラム

/usr/bin/mailq
: メールの待ち行列をリストするプログラム。これを実行するのは、sendmail に *-bp* オプションを指定して実行するのと同じです。

/usr/bsd/newaliases
: */usr/lib/sendmail* へのリンク。エリアス・データベースを再構築します。このプログラムを実行することは、sendmail に **-bi** フラグを指定して実行するのと同じです。

/usr/lib/aliases
: エリアス・ファイルのテキスト形式のファイル

/usr/lib/sendmail
: sendmail プログラム

/usr/lib/sendmail.st
: 統計ファイル。なくてもかまいません

/var/spool/mqueue
メール待ち行列と一時ファイルが存在するディレクトリ

*/var/spool/mqueue/df**
データ・ファイル

*/var/spool/mqueue/nf**
一意な ID を作成する際に使用するファイル

*/var/spool/mqueue/qf**
メッセージの制御（待ち行列）ファイル

*/var/spool/mqueue/tf**
qf ファイルの一時ファイル。待ち行列ファイルのリビルド時に使用されます。

*/var/spool/mqueue/xf**
現在のセッションのトランスクリプト

sendmail のデバッグ・フラグ

次に、`sendmail` のデバッグ・フラグをまとめます。特に有用なフラグには、アスタリスク（*）を付けてあります。

0.1*	`sendmail` のバージョン情報を表示します。
0.4*	ローカル・ホストのすべての既知名を表示します。
0.15*	配信エージェントをダンプします。
0.20*	各インタフェースのネットワーク・アドレスを表示します。
0.44	**printav()** が要素のアドレスを表示します。
1.1*	ローカルに生成したメールの「From」アドレスを表示します。
2.1*	終了状態とエンベロープ・オプションを表示します。
4.80*	**enoughspace()** の状態をトレースします。
5.4	**tick()** 呼出しの引数を表示します。
5.5	**setevent()** 呼出しと **clrevent()** 呼出しの引数を表示します。
5.6	**tick()** 呼出し時にイベント待ち行列を表示します。

6.1*	**savemail()** または **returntosender()** エラー処理の呼出しを示します。
6.5	**savemail()** ステート・マシンで状態をトレースします。
7.1*	待ち行列ファイルに割当てたエンベロープに関する情報を表示します。
7.2*	指定した待ち行列ファイル名を表示します。
7.20*	指定した中間の待ち行列ファイル名を表示します。
8.1*	リゾルバ呼出しに関する各種の情報を表示します。
8.2*	正式に認められた標準ホスト名を **getcanonname**(3) に返します。
8.3*	ドロップしたローカル・ホスト名をトレースします。
8.5*	**getcanonname**(3) で試行中のホスト名を表示します。
8.7*	**getcanonname**(3) で試行中のホストに yes または no で応答します。
8.8*	間違ったタイプを送り返してきたときに、MX リゾルバのデバッグをオンにします。
9.1*	**gethostbyaddr()** 呼出しの結果を表示します。
10.1*	メッセージ配信情報を表示します。
11.1*	**openmailer()** への呼出しに使用されるメール・プログラムに関する情報を記録します。
11.2*	配信中に実行されるユーザとグループの ID を表示します。
12.1*	**remotename()** 入出力を表示します。
13.1*	**sendall()**— 送信先のアドレスを表示します。
13.3	**sendall()**— 失敗を探すループにある各アドレスを表示します。
13.4	**sendall()**— エラーの受信者を表示します。
14.2	**commaize()** 呼出しを示します。
15.1	**getrequests()** が使用するポート番号またはソケット番号を示します。
15.2	**getrequests()** をいつフォークするのか、またはいつ返すのかを示します。
15.15	**getrequests()** の DEBUG ソケット・オプションを設定します。

16.1*	**makeconnection()** で接続されているホスト、アドレス、およびソケットを示します。
16.14	**makeconnection()** 内で DEBUG ソケット・オプションを設定します。
18.1*	SMTP チャットを表示します。
18.100	各 SMTP 応答を読込んだ後、sendmail を中断します。
20.1*	**parseaddr()** 入出力を表示します。
21.2*	書換えルール・セット・サブルーチンの calls/returns、入出力を表示します。実行時のマクロ展開を表示します。
21.3*	書換えルール内からの書換えサブルーチン呼出しを示します。
21.4*	書換え結果を表示します。
21.10*	ルールの失敗を示します。
21.12*	ルールがマッチしたことを示し、アドレス書換えのステップを表示します。
21.15*	書換えたアドレスを表示します。
21.35	パターンおよびサブジェクトの要素を表示します。
22.1*	**prescan()** で分析する前に無効なアドレスを表示します。
22.11*	**prescan()** で分析する前に入力されたアドレスを表示します。
25.1*	To リストを表示します。
26.1*	配信先の重複抑制について表示します。
26.6*	受信者のパスワード照合処理を表示します。
27.1*	エリアスと転送の変換、およびエラーを表示します。
27.2*	インクルード・ファイル、セルフ・リファレンス、ホーム上エラー
27.3*	**aliaslookup()** の詳細情報を表示します。
27.4*	ほかのユーザのインクルード・ファイルを実行しようとしたときに、警告します。
27.9*	インクルード・ファイル読込み中に uid/gid を変更しようとしたときに、警告します。
28.1*	ユーザのデータベース・トランザクションをトレースします。

29.4*	あいまいな検索の結果得られたマッチングを表示します。
30.1	メッセージ収集時のヘッダ終了を示します。
30.2	**eatfrom()** 呼出しの引数を表示します。
30.3	メッセージに Apparently-To ヘッダをいつ追加するのかを示します。
31.2*	ヘッダ処理をトレースします。
31.6	**chompheader()** 呼出しと処理するヘッダを示します。
32.1	収集したヘッダを表示します。
33.1	**crackaddr()** 入出力を表示します。
34.11*	ヘッダがどのように生成され、スキップされたかをトレースします。
35.9*	マクロ定義を表示します。
35.24	マクロ展開を表示します。
36.5	シンボル・テーブルの処理を表示します。
36.9	シンボル・テーブル・ハッシュ関数の結果を表示します。
37.1*	オプションを集合として表示します。
37.2*	書換えクラスのロード状況を表示します。
37.8*	クラスに追加されたワードを表示します。
38.2*	開き、失敗したマップを表示します。
38.4*	開いたマップの結果を表示します。
38.20*	マップ検索をトレースします。
40.1*	メッセージの待ち行列登録を示し、待ち行列の内容を表示します。
40.4*	待ち行列制御ファイルの内容を表示します。
40.5*	メッセージを制御するユーザに関する情報を表示します。
41.1	待ち行列の順番を表示します。
41.2	**orderq()** が制御ファイルを開けなかったことを示します。
44.5*	**writable()** 呼出しをトレースします。

48.2*	ユーザ設定可能な **check_rule** セットへの呼出しをトレースします。
45.1	**setsender()** 呼出しを示します。
50.1	**dropenvelope()** 呼出しを示します。
51.4	トランスクリプト・ファイル（*qxAAXXXXX* ファイル）を削除しません。
52.1	**disconnect()** 呼出しを示し、I/O ファイル記述子を表示します。
52.5	接続を切りません。
60.1*	**rewrite()** 内マップの検索に関する情報を表示します。
61.1*	MX レコードの検索に関する情報を表示します。

索引

I

L

M

N

P

R

S

T

U

V

Y

い

お

か

き

く

け

こ

さ

し

な

ね

は

ひ

ふ

ほ

ま

め

も

ら

り

る

ろ