Oracle® Solaris の管理: IP サービス

Part No: E38855–03
2013 年 3 月

ORACLE

130701@25097

目次

はじめに

『Oracle Solaris の管理: IP サービス』へようこそ。このドキュメントは、Oracle Solaris システム管理の重要な部分を説明する 14 巻から成るドキュメントセットの一部です。このドキュメントの記述は、Oracle Solaris OS がインストール済みであることが前提です。ネットワークを構成する準備ができているか、またはネットワーク上で必要なネットワークソフトウェアを構成する準備ができているようにしてください。

注 – この Oracle Solaris のリリースでは、SPARC および x86 系列のプロセッサアーキテクチャーを使用するシステムをサポートしています。サポートされるシステムは、Oracle Solaris OS: Hardware Compatibility Lists に記載されています。このドキュメントでは、プラットフォームにより実装が異なる場合は、それを特記します。

このドキュメントの x86 に関連する用語については、次を参照してください。

- x86 は、64 ビットおよび 32 ビットの x86 互換製品系列を指します。
- x64 は特に 64 ビット x86 互換 CPU を指します。
- 「32 ビット x86」は、x86 をベースとするシステムに関する 32 ビット特有の情報を指します。

サポートされるシステムについては、Oracle Solaris OS: Hardware Compatibility Lists を参照してください。

対象読者

このドキュメントは、Oracle Solaris が動作しており、ネットワークに構成されているシステムの管理を行うユーザーを対象としています。このドキュメントを利用するにあたっては、UNIXのシステム管理について少なくとも 2 年の経験が必要です。UNIX システム管理のトレーニングコースに参加することも役に立ちます。

Solaris システム管理マニュアルセットの構成

システム管理ガイドセットに含まれる各ガイドとその内容は、次のとおりです。

ドキュメントのタイトル	トピック
『Oracle Solaris の管理: 基本管理』	Oracle Solaris コマンドの使用、システムのブートとシャットダウン、サーバーとクライアントのサポート、ユーザーアカウントとグループの管理、サービスの管理、システム情報、システムリソース、およびシステム性能、ソフトウェアの管理、コンソールと端末、およびシステムとソフトウェアの問題のトラブルシューティング
『Solaris のシステム管理 (上級編)』	端末とモデムの設定、システムリソースの管理 (ディスク割り当て、アカウンティング、および crontab ファイルの管理)、システムプロセスの管理、および Oracle Solaris ソフトウェアの障害追跡
『Oracle Solaris の管理: デバイスとファイルシステム』	リムーバブルメディア、ディスクとデバイス、ファイルシステム、およびデータのバックアップと復元
『Oracle Solaris の管理: IP サービス』	TCP/IP ネットワークの管理、IPv4 および IPv6 アドレスの管理、DHCP、IP セキュリティー、IKE、IP フィルタ、IP ネットワークのマルチパス化 (IPMP)、および IPQoS
『Solaris のシステム管理 (ネーミングとディレクトリサービス : DNS、NIS、LDAP 編)』	DNS、NIS、および LDAP のネーミングとディレクトリサービス (NIS から LDAP への移行、および NIS+ から LDAP への移行を含む)
『Solaris のシステム管理 (ネーミングとディレクトリサービス : NIS+ 編)』	NIS+ のネーミングとディレクトリサービス
『Solaris のシステム管理 (ネットワークサービス) 』	Web キャッシュサーバー、時間関連サービス、ネットワークファイルシステム (NFS と autofs)、メール、SLP、および PPP
『Oracle Solaris の管理: Oracle Solaris コンテナ - リソース管理と Oracle Solaris ゾーン』	リソース管理に関連する計画とタスク、拡張アカウンティング、リソース制御、フェアシェアスケジューラ (FSS)、リソース上限デーモン (rcapd) による物理メモリーの制御、およびリソースプール (Solaris Zones ソフトウェア区分技術と lx ブランドゾーンによる仮想化)
『Solaris のシステム管理ガイド (印刷)』	印刷に関するトピックや、サービス、ツール、プロトコル、およびテクノロジを使って印刷サービスおよびプリンタを設定および管理する方法
『Solaris のシステム管理: セキュリティーサービス』	監査、デバイス管理、ファイルセキュリティー、BART、Kerberos サービス、PAM、暗号化フレームワーク、鍵管理、特権、RBAC、SASL、および Secure Shell

ドキュメントのタイトル	トピック
『Oracle Solaris ZFS 管理ガイド』	ZFS ストレージプールおよびファイルシステムの作成と管理、スナップショット、クローン、バックアップ、アクセス制御リスト (ACL) による ZFS ファイルの保護、ゾーンがインストールされた Solaris システム上での ZFS の使用、エミュレートされたボリューム、およびトラブルシューティングとデータ回復
『Trusted Extensions 管理者の手順』	Trusted Extensions 固有のシステム管理
『Oracle Solaris Trusted Extensions 構成ガイド』	Solaris 10 5/08 リリース以降での、Trusted Extensions の計画、有効化、および初期構成の方法

関連情報

このドキュメントは、次のマニュアルを参照します。

- Stevens, W. Richard 著『TCP/IP Illustrated, Volume 1, The Protocols』 Addison Wesley 発行、1994 年
- Hunt Craig 著『TCP/IP Network Administration, 3rd Edition』 O'Reilly 発行、2002 年
- Perkins, Charles E. 著『Mobile IP Design Principles and Practices』 Massachusetts、Addison-Wesley Publishing Company 発行、1998 年
- Solomon, James D. 著『Mobile IP: The Internet Unplugged』 New Jersey、Prentice-Hall, Inc. 発行、1998 年
- Ferguson, Paul および Geoff Huston 著『Quality of Service』 John Wiley & Sons, Inc. 発行、1998 年
- Kilkki, Kalevi 著『Differentiated Services for the Internet 』 Macmillan Technical Publishing 発行、1999 年

関連するサードパーティーの Web サイト情報

このドキュメントにはオラクル社およびその関連会社が所有または管理しない Web サイトへのリンクが含まれている場合があります。

Oracle Solaris の IP フィルタ機能は、オープンソースの IP Filter ソフトウェアから派生したものです。IP フィルタのライセンス条項、帰属、および著作権声明については、`/usr/lib/ipf/IPFILTER.LICENCE` を参照してください (デフォルト)。Oracle Solaris オペレーティングシステム がデフォルト以外の場所にインストールされている場合は、所定のパスを修正して、インストールした場所にあるファイルにアクセスします。

Oracle サポートへのアクセス

Oracle のお客様は、My Oracle Support を通じて電子的なサポートを利用することができます。詳細は、http://www.oracle.com/pls/topic/lookup?ctx=acc&id=info を参照してください。聴覚に障害をお持ちの場合は、http://www.oracle.com/pls/topic/lookup?ctx=acc&id=trs を参照してください。

表記上の規則

次の表では、このマニュアルで使用される表記上の規則について説明します。

表 P–1　表記上の規則

字体	説明	例
`AaBbCc123`	コマンド名、ファイル名、ディレクトリ名、画面上のコンピュータ出力、コード例を示します。	`.login` ファイルを編集します。 `ls -a` を使用してすべてのファイルを表示します。 `machine_name% you have mail.`
`AaBbCc123`	ユーザーが入力する文字を、画面上のコンピュータ出力と区別して示します。	`machine_name%` **`su`** `Password:`
aabbcc123	Placeholder: 実際に使用する特定の名前または値で置き換えます。	ファイルを削除するには、`rm` *filename* と入力します。
AaBbCc123	書名、新しい単語、および強調する単語を示します。	『ユーザーズガイド』の第 6 章を参照してください。 キャッシュは、ローカルに格納されるコピーです。 ファイルを保存しないでください。 注: いくつかの強調された項目は、オンラインでは太字で表示されます。

コマンド例のシェルプロンプト

Oracle Solaris OS に含まれるシェルで使用する、UNIX のシステムプロンプトとスーパーユーザープロンプトを次に示します。コマンド例では、シェルプロンプトはコマンドが標準ユーザーまたは特権ユーザーのどちらによって実行されるべきかを示しています。

表 P–2 シェルプロンプト

シェル	プロンプト
Bash シェル、Korn シェル、および Bourne シェル	`$`
Bash シェル、Korn シェル、および Bourne シェルのスーパーユーザー	`#`
C シェル	`machine_name%`
C シェルのスーパーユーザー	`machine_name#`

パート I

システム管理の概要:IP サービス

ここでは、TCP/IP プロトコル群と、Oracle Solaris におけるその実装についての概要情報を示します。

◆ ◆ ◆ 第 1 章

1 Oracle Solaris TCP/IP プロトコル群 (概要)

この章では、Oracle Solaris 実装の TCP/IP ネットワークプロトコル群を紹介します。ここでの説明は、TCP/IP の基本概念をまだよく理解していないシステム管理者およびネットワーク管理者を対象としています。このドキュメントの残りの部分では、読者がそれらの概念をよく理解していることを前提としています。

この章では、次の内容について説明します。

このリリースの最新情報

Solaris 10 5/08 以降では、モバイル IP 機能が削除されました。モバイル IP は、Solaris 10 OS 8/07 以前のリリースで利用できます。

TCP/IP プロトコル群の概要

この節では、TCP/IP に含まれるプロトコルについて詳しく説明します。ここに示す情報は概念的なものですが、各プロトコルの名前を学習できます。また、各プロトコルがどのように機能するかも理解できます。

「TCP/IP」は、「インターネットプロトコル群」を形成するネットワークプロトコルの集合を示す頭字語です。多くの書籍では、「インターネット」という用語は、プロトコル群と広域ネットワークの両方を表すものとして使用されています。このドキュメントでは、「TCP/IP」は特にインターネットプロトコル群を表し、「インターネット」は広域ネットワークとインターネットを運営する組織を表すものとします。

TCP/IP ネットワークとほかのネットワークを相互接続するには、そのネットワーク用に一意の IP アドレスを取得する必要があります。このドキュメントの執筆時点では、このアドレスはインターネットサービスプロバイダ (ISP) から取得します。

ネットワーク上のホストがインターネットのドメインネームシステム (DNS) に参加する場合は、一意のドメイン名を取得して登録する必要があります。InterNIC は、世界中のレジストリのグループによってドメイン名の登録を調整します。DNS についての詳細は、『Solaris のシステム管理 (ネーミングとディレクトリサービス : DNS、NIS、LDAP 編)』を参照してください。

プロトコル層および開放型相互接続モデル

大部分のネットワークプロトコル群は一連の層から構成されており、まとめて「プロトコルスタック」と呼ばれる場合もあります。各層はそれぞれ特定の目的のために設計されていて、送信側システムと受信側システムの両方に存在しています。1 台のシステムの特定の層は、別のシステムの「ピアプロセス」が送受信するのとまったく同じ オブジェクトを送受信します。このような動作は、問題の層の上下の層で進行していることとは独立して行われます。つまり、システムの各層は、同じシステムのほかの層から独立して、ほかのシステムの同じ層と協調して働きます。

OSI 参照モデル

ほとんどのネットワークプロトコル群は層状に構造化されています。国際標準化機構 (ISO) は構造化された層を使用する開放型相互接続 (OSI) 参照モデルを設計しました。OSI モデルは、ネットワーク活動が 7 つの層から成る構造を持つものと規定しています。各層には、1 つ以上の プロトコルが関連付けられています。層は、連携するネットワーク相互間でのすべての種類のデータ転送に共通するデータ転送操作を表します。

OSI モデルでは、プロトコル層を上 (第 7 層) から下 (第 1 層) へ並べて表します。次の表にモデルを示します。

表 1–1　OSI (開放型相互接続) 参照モデル

層の番号	層の名前	説明
7	アプリケーション	誰でも使用できる標準の通信サービスとアプリケーションで構成されています。
6	プレゼンテーション	情報が解読可能な形で受信側システムに渡されるようにします。
5	セッション	連携システム間の接続と終了を管理します。
4	トランスポート	データの転送を管理します。また、受信されたデータと送信されたデータが同じになることを保証します。

表 1–1 OSI (開放型相互接続) 参照モデル (続き)

層の番号	層の名前	説明
3	ネットワーク	ネットワーク間でのデータのアドレス指定と配送を管理します。
2	データリンク	ネットワークメディアを通過するデータの転送を取り扱います。
1	物理	ネットワークハードウェアの特性を定義します。

OSI モデルで定義する概念上の操作は、特定のネットワークプロトコル群に固有のものではありません。たとえば、OSI ネットワークプロトコル群は、OSI モデルの 7 つの層をすべて実装しています。TCP/IP は、OSI モデル層のいくつかを使用し、その他の層を合併しています。その他のネットワークプロトコル、たとえば SNA では、8 番目の層が追加されています。

TCP/IP プロトコルアーキテクチャーモデル

OSI モデルは、一群のプロトコルによる理想的なネットワーク通信を定義します。TCP/IP は、このモデルに直接対応するわけではありません。TCP/IP では、複数の OSI 層が 1 つに合併されたり、まったく使用されない層があったりします。次の表は、Oracle Solaris 実装の TCP/IP の層を示しています。最上位層 (アプリケーション) から最下位層 (物理ネットワーク) まで並べてあります。

表 1–2 TCP/IP プロトコルスタック

OSI 参照の層の番号	対応する OSI 層	TCP/IP 層	TCP/IP プロトコルの例
5,6,7	アプリケーション、セッション、プレゼンテーション	アプリケーション	NFS、NIS、DNS、LDAP、`telnet`、`ftp`、`rlogin`、`rsh`、`rcp`、RIP、RDISC、SNMP、その他
4	トランスポート	トランスポート	TCP、UDP、SCTP
3	ネットワーク	インターネット	IPv4、IPv6、ARP、ICMP
2	データリンク	データリンク	PPP、IEEE 802.2
1	物理	物理ネットワーク	Ethernet (IEEE 802.3) トークンリング、RS-232、FDDI、その他

TCP/IP プロトコル層と相当する OSI モデルを表に示します。また、TCP/IP プロトコルスタックの各レベルで使用可能なプロトコルの例も示します。通信トランザクションに関与する各システムは、それぞれ固有の実装によるプロトコルスタックを実行します。

物理ネットワーク層

「物理ネットワーク層」は、ネットワークで使用されるハードウェアの特性を規定します。たとえば、通信メディアの物理特性を規定します。TCP/IP の物理層はハードウェア規格を意味しています。たとえば、Ethernet ネットワークメディアの仕様である IEEE 802.3 や、標準ピンコネクタの仕様である RS-232 などです。

データリンク層

「データリンク層」は、パケットのネットワークプロトコルの種類 (この例ではTCP/IP) を識別します。データリンク層は、エラー制御と「フレーミング」も行います。データリンク層プロトコルの例としては、Ethernet IEEE 802.2 フレーミングおよびポイントツーポイントプロトコル (PPP) フレーミングがあります。

インターネット層

「ネットワーク層」または「*IP* 層」とも呼ばれるインターネット層は、ネットワークのためにパケットを送受信します。この層には、強力なインターネットプロトコル (IP)、アドレス解決プロトコル (ARP)、インターネット制御メッセージプロトコル (ICMP) が組み込まれています。

IP プロトコル

IP プロトコルと関連するルーティングプロトコルは、TCP/IP スイート全体でもっとも重要である可能性があります。IP は次の機能を受け持ちます。

- **IP** アドレス指定 – IP アドレス指定規約は、IP プロトコルの一部です。58 ページの「IPv4 アドレス指定スキームの設計」 は IPv4 アドレス指定を紹介し、76 ページの「IPv6 アドレス指定の概要」 は IPv6 アドレス指定を紹介します。
- ホスト間通信 – IP は、受信側システムの IP アドレスに基づいてパケットが通る必要のあるパスを決定します。
- パケットの形式設定 – IP は、パケットを「データグラム」と呼ばれる単位にまとめます。データグラムについては、46 ページの「インターネット層: パケットの送信準備」 に詳しい説明があります。
- 断片化 – ネットワークメディアで転送するにはパケットが大きすぎる場合、送信側システムの IP は、パケットを小さなフラグメントに分割します。受信側システム の IP は、これらのフラグメントを組み立てて元のパケットに戻します。

Oracle Solaris は、IPv4 アドレス指定形式と IPv6 アドレス指定の両方をサポートしています。これらの形式については、このドキュメントで説明します。インターネットプロトコルについて言及するときに混乱を避けるため、次の規則を適用します。

- 「IP」という用語を使用している場合、その説明は IPv4 と IPv6 の両方に適用されます。
- 「IPv4」という用語を使用している場合、その説明は IPv4 のみに適用されます。

- 「IPv6」という用語を使用している場合、その説明は IPv6 のみに適用されます。

ARP プロトコル

アドレス解決プロトコル (ARP) は、概念上、データリンク層とインターネット層の間に存在します。ARP は、Ethernet アドレス (48 ビット長) を既知の IP アドレス (32 ビット長) にマッピングし、IP はこの情報に基づいてデータグラムを正しい受信側システムに向けることができます。

ICMP プロトコル

インターネット制御メッセージプロトコル (ICMP) は、ネットワークエラー状態を検出し、報告します。ICMP は次の事項について報告します。

- ドロップされたパケット – 速く到着しすぎたために、処理できないパケット
- 接続障害 – 宛先システムに到達できない
- 方向の変更 – 送信側システムが別のルーターを使用するように変更する

第 8 章「TCP/IP ネットワークの管理 (手順)」には、ICMP をエラー検出に使用する Oracle Solaris コマンドに関するより詳しい情報が含まれています。

トランスポート層

TCP/IP の「トランスポート層」は、データ受信の肯定応答を交換し、消失したパケットを再送することによって、パケットが順番にかつエラーなしで到着することを保証します。このような通信は、「エンドツーエンド」と呼ばれます。このレベルのトランスポート層プロトコルには、伝送制御プロトコル (TCP)、ユーザーデータグラムプロトコル (UDP)、およびストリームコントロール伝送プロトコル (SCTP) があります。TCP と SCTP は、信頼性の高いエンドツーエンドサービスを提供します。UDP は、信頼性の低いデータグラムサービスを提供します。

TCP プロトコル

TCP では、アプリケーション同士が物理的な回路で接続されているかのような相互通信を可能にします。TCP は、独立したパケットの形ではなく、文字単位で転送されているような形でデータを送信します。この転送は、次の点で構成されます。

- 接続を開始する開始点
- バイト順の転送全体
- 接続を終了する終了点

TCP は、転送するデータにヘッダーを添付します。このヘッダーには、送信側システム上のプロセスが受信側システム上の対等プロセスに接続できるようにするための、多数のパラメータが含まれています。

TCP は、送信側ホストと受信側ホストとの間に終端間接続を確立することにより、パケットが宛先に到達したことを確認します。したがって、TCP は、「信頼性の高い接続指向型」プロトコルとみなすことができます。

SCTP プロトコル

SCTP は、TCP から使用可能なアプリケーションに同じサービスを提供する、信頼性の高い接続指向のトランスポート層プロトコルです。さらに、SCTP は、複数のアドレスを持つ、つまり「マルチホーム」のシステム間の接続をサポートできます。送信側システムと受信側システム間の SCTP 接続は、「アソシエーション」と呼ばれます。アソシエーションのデータは、グループ別に整理されます。SCTP はマルチホームをサポートしているため、一部のアプリケーション、特に電気通信業界で使用されるアプリケーションは、TCP ではなく SCTP で実行する必要があります。

UDP プロトコル

UDP は、データグラム送信サービスを提供します。UDP は、受信側ホストと送信側ホストとの間の接続の検査は行いません。UDP は接続の確立と検査を省略するので、少量のデータを送信するアプリケーションは、UDP を使用します。

アプリケーション層

「アプリケーション層」は、だれでも使用できる標準インターネットサービスとネットワークアプリケーションを定義します。これらのサービスとトランスポート層の両方の働きにより、データの送受信が行われます。アプリケーション層のプロトコルは多数存在します。

次に、アプリケーション層プロトコルの例を示します。

- 標準 TCP/IP サービス。たとえば、`ftp`、`tftp`、`telnet` コマンドなど
- UNIX の “r” (リモート) コマンド。たとえば、 `rlogin` や `rsh` など
- ネームサービス。たとえば、NIS やドメインネームシステム (DNS) など
- ディレクトリサービス (LDAP)
- ファイルサービス。たとえば NFS サービスなど
- SNMP (ネットワーク管理用プロトコルの一種。Simple Network Mamagement Protocol の略)
- ルーター発見サーバープロトコル (RDISC) と経路制御情報プロトコル (RIP) の経路制御プロトコル

標準 TCP/IP サービス

- **FTP** と匿名 **FTP** – ファイル転送プロトコル (FTP) は、リモートネットワークとの間でファイルを送受信します。このプロトコルには、`ftp` コマンドと `in.ftpd` デーモンが含まれます。ユーザーは、リモートホストの名前とファイル転送コマンドのオプションを、ローカルホストのコマンド行に指定します。リモートホスト上の `in.ftpd` デーモンがそのあとローカルホストからの要求を処理します。 `rcp` とは違って、`ftp` は、リモートコンピュータのオペレーティングシステムが UNIX

ベースでない場合でも動作します。リモートシステムが匿名 FTP を認めるように構成されている場合を除いて、ftp 接続を行うときにユーザーはリモートシステムにログインする必要があります。

インターネットに接続されている「匿名 *FTP* サーバー」からは、莫大な量の資料を入手できます。大学その他の研究機関がこれらのサーバーを設定して、ソフトウェア、研究報告、その他の情報をパブリックドメインに公開しています。このようなサーバーにログインするときは、anonymous というログイン名を使用するため、「匿名 (anonymous) FTP サーバー」と呼ばれます。

匿名 FTP の使用法と匿名 FTP サーバーの設定については、このマニュアルでは説明しません。しかし、たとえば『*The Whole Internet User's Guide & Catalog*』など、匿名 FTP について詳しく説明している多数の書籍が市販されています。FTP の使用手順については、『Solaris のシステム管理 (ネットワークサービス)』に記載されています。ftp(1) のマニュアルページには、コマンドインタプリタによって呼び出されるすべての ftp コマンドオプションについての説明があります。ftpd(1M) のマニュアルページには、in.ftpd デーモンが提供するサービスについての説明があります。

- **Telnet** – Telnet プロトコルを使用すると、TCP/IP を実行しているネットワーク上で端末および端末指向のプロセスが通信できます。このプロトコルは、ローカルシステムでは telnet プログラムとして、リモートマシンでは in.telnetd デーモンとして実装されます。Telnet は、2 つのホストが文字単位または行単位で通信できるようなユーザーインタフェースを提供します。Telnet にはコマンドのセットが含まれていますが、これについては、telnet(1) のマニュアルページに詳しい説明があります。
- **TFTP** – 簡易ファイル転送プロトコル (tftp) は、ftp と同じような機能を提供しますが、ftp のような相互接続は確立しません。したがって、ユーザーは、ディレクトリの内容を表示したり、ディレクトリを変更したりすることはできません。ユーザーは、コピーするファイルのフルネームを知っている必要があります。tftp のコマンドセットについては、tftp(1) のマニュアルページに説明があります。

UNIX の "r" コマンド

UNIX の “r” コマンドを使用すると、ユーザーは自分のローカルマシンからコマンドを発行して、そのコマンドをリモートホストで実行できます。

この種のコマンドには次のものがあります。

- rcp
- rlogin
- rsh

これらのコマンドの使用方法は、rcp(1)、rlogin(1)、および rsh(1) のマニュアルページに記載があります。

ネームサービス

Oracle Solaris は、次のネームサービスを提供します。

- **DNS** – ドメインネームシステム (DNS) は、インターネットが TCP/IP ネットワーク用に提供するネームサービスです。DNS は、ホスト名から IP アドレスに変換するサービスを提供します。また、メール管理用のデータベースとしての働きもします。このサービスの詳細は、『Solaris のシステム管理 (ネーミングとディレクトリサービス : DNS、NIS、LDAP 編)』を参照してください。`resolver(3RESOLV)` のマニュアルページも参照してください。
- `/etc` ファイル - ホストベースの UNIX ネームシステムは、最初はスタンドアロンの UNIX マシン用に開発されたあと、ネットワークで使用されるように改良されました。UNIX オペレーティングシステムの旧バージョンの多くや UNIX マシンでは、現在でもこのシステムが使用されていますが、大規模で複雑なネットワークにはあまり適切ではありません。
- **NIS** – ネットワーク情報サービス (NIS) は DNS とは独立して開発され、目的はやや異なっています。DNS は数値 IP アドレスの代わりにマシン名を使うことによって、通信を簡略化することに焦点を当てているのに対して、NIS の場合は、多様なネットワーク情報を集中管理することによりネットワーク管理機能を高めることに焦点を絞っています。NIS には、マシンの名前とアドレス、ユーザー、ネットワークそのもの、ネットワークサービスについての情報も格納されます。NIS 名前空間情報は NIS マップに格納されています。NIS アーキテクチャーと NIS 管理の詳細は、『Solaris のシステム管理 (ネーミングとディレクトリサービス : DNS、NIS、LDAP 編)』を参照してください。

ディレクトリサービス

Oracle Solaris は、Sun ONE (Sun Open Net Environment) およびほかの LDAP Directory Server を使用する場合、LDAP (Lightweight Directory Access Protocol) をサポートします。ネームサービスとディレクトリサービスの違いは、拡張機能の差です。ディレクトリサービスはネームサービスと同じ機能のほかに、追加機能を提供します。『Solaris のシステム管理 (ネーミングとディレクトリサービス : DNS、NIS、LDAP 編)』を参照してください。

ファイルサービス

NFS アプリケーション層プロトコルは、Oracle Solaris 用にファイルサービスを提供します。NFS サービスに関する詳しい情報は、『Solaris のシステム管理 (ネットワークサービス)』に記載されています。

ネットワーク管理

シンプルネットワーク管理プロトコル (SNMP) を使用すると、ネットワークのレイアウトおよび主要マシンのステータスを参照できます。また、グラフィカル

ユーザーインタフェース (GUI) ベースのソフトウェアで複雑なネットワーク統計情報を参照できます。多くの企業が、SNMP を実装するネットワーク管理パッケージを提供しています。

ルーティングプロトコル

経路制御情報プロトコル (RIP) およびルーター発見サーバープロトコル (RDISC) は、2 つとも、TCP/IP ネットワーク用の経路制御プロトコルです。Oracle Solaris で使用できるルーティングプロトコルの一覧については、表 5–1 および 表 5–2 を参照してください。

TCP/IP プロトコルがデータ通信を行う方法

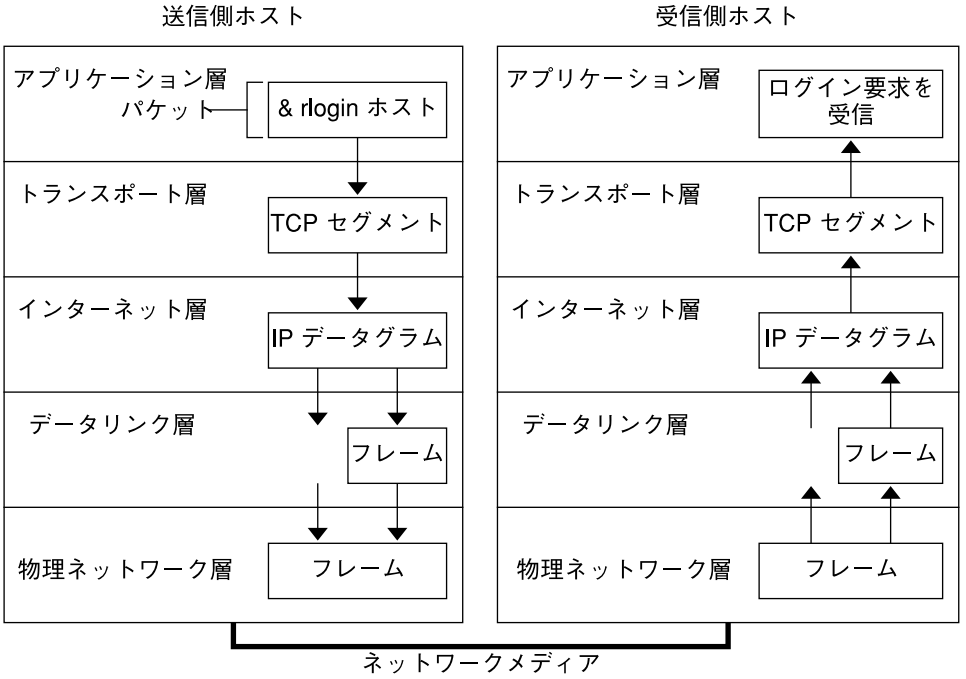
ユーザーが TCP/IP アプリケーション層プロトコルを使用するコマンドを発行すると、一連のイベントが開始されます。ユーザーのコマンドまたはメッセージはローカルシステム上の TCP/IP プロトコルスタックを通過します。次に、ネットワークメディアを通過して、リモートシステムのプロトコルに到達します。送信側ホストの各層のプロトコルにより、オリジナルのデータに情報が付加されていきます。

送信側ホストの各層のプロトコルは、受信側ホストのそれぞれの対等プロトコルとの間で対話します。図 1–1 に、この対話を示します。

データのカプセル化と TCP/IP プロトコルスタック

パケットとは、ネットワーク間で転送される情報の基本単位のことです。基本パケットは、送信側システムと受信側システムのアドレスを含むヘッダー、転送されるデータを含む本体、つまり「ペイロード」で構成されます。パケットが TCP/IP プロトコルスタックを通過するとき、各層のプロトコルは、基本ヘッダーにフィールドを追加したり、そこからフィールドを削除したりします。送信側システムのプロトコルがパケットヘッダーにデータを追加する場合、そのプロセスを「データのカプセル化」と呼びます。また、変更後のパケットを表す言葉は、次の図に示すように層によって異なります。

図 1–1　パケットが TCP/IP スタックを通過する方法



この節では、パケットのライフサイクルについて要約します。ライフサイクルは、ユーザーがコマンドを発行するか、またはメッセージを送信することによって開始します。受信側システムの該当アプリケーションがパケットを受信するとライフサイクルは終了します。

アプリケーション層:通信の発生

パケットの処理は、あるシステム上のユーザーがリモートシステムへのアクセスを必要とするメッセージの送信やコマンドの発行をしたときから始まります。アプリケーションプロトコルは、対応する TCP か UDP のどちらかのトランスポート層プロトコルがそのパケットを取り扱えるように、パケットの形式を設定します。

たとえば、あるユーザーが `rlogin` コマンドを発行して、リモートシステムにログインしようとしたとします (図 1–1 を参照)。`rlogin` コマンドは TCP トランスポート層プロトコルを使用します。TCP は、コマンド内の情報を含むデータをバイトストリーム形式で受け取るものと仮定しています。したがって、`rlogin` はこのデータを TCP ストリームとして送信します。

トランスポート層:データのカプセル化の開始

データがトランスポート層に到達すると、トランスポート層のプロトコルは、データのカプセル化プロセスを開始します。トランスポート層は、アプリケーションデータをトランスポートプロトコルのデータ単位にカプセル化します。

トランスポート層プロトコルは、転送ポート番号で区別される送信側アプリケーションと受信側アプリケーション間に仮想のデータフローを作成します。ポート番号は、メモリー内のデータ送受信専用の場所である「ポート」を識別します。さらに、トランスポートプロトコル層は、信頼性の高い順序どおりのデータ転送など、その他のサービスを提供する場合もあります。最終的な結果は、TCP、SCTP または UDP のどれで情報を処理したかによって異なります。

TCP のセグメンテーション

TCP はデータを確実に受信側ホストに送信できるため、「接続指向」のプロトコルと呼ばれます。図 1–1 は、TCP プロトコルがどのように `rlogin` コマンドからのストリームを受信するかを示しています。次に TCP は、アプリケーション層から受け取ったデータをセグメントに分割し、各セグメントにヘッダーを添付します。

セグメントヘッダーには、送信側と受信側のポート、セグメント順序に関する情報、「検査合計」と呼ばれるデータフィールドが含まれています。両方のホストの TCP プロトコルがこの検査合計データを使用して、データがエラーなしに転送されたかどうかを判別します。

TCP 接続の確立

TCP は、受信側システムでデータを受信する準備ができているかどうかを、セグメントを使用して判断します。まず、送信側 TCP は「*SYN*」というセグメントを受信側ホストの TCP プロトコルに送信して、接続を確立することを知らせます。次に、受信側 TCP は「*ACK*」というセグメントを戻して、セグメントを正しく受信したことを知らせます。送信側 TCP は新たな ACK セグメントを送信して、それからデータの送信を開始します。このような制御情報の交換を「*3* 相ハンドシェーク」と呼びます。

UDP パケット

UDPは「接続のない」プロトコルです。TCP の場合と異なり、UDP は、受信側ホストにデータが到達したかどうかを確認しません。その代わりに、UDP は、アプリケーション層から受信したメッセージを「*UDP* パケット」の形式に設定します。UDP は、各パケットにヘッダーを付加します。ヘッダーには、送信側ポートと受信側ポート、パケットの長さを示すフィールド、検査合計が含まれます。

送信側の UDP プロセスは、受信側ホストのピア UDP プロセスにパケットを送信しようとします。アプリケーション層は、受信側 UDP プロセスが、パケットを受信したことを示す肯定応答を戻すかどうかを判別します。UDP は受領の通知を必要としません。UDP は 3 相ハンドシェークを使用しません。

インターネット層: パケットの送信準備

トランスポートプロトコル TCP、UDP、および SCTP がセグメントとパケットをインターネット層に渡すと、これらのセグメントとパケットはインターネット層の IP プロトコルによって処理されます。IP は、これらのセグメントとパケットを「*IP* データグラム」と呼ばれる単位に形式化して、送信できるように準備します。次に、IP はデータグラムの IP アドレスを判別して、受信側ホストへの効率的な配送ができるようにします。

IP データグラム

IP は、TCP や UDP が追加した情報に加えて、「*IP* ヘッダー」をセグメントまたはパケットのヘッダーに添付します。 IP ヘッダーには、送信側ホストと受信側ホストの IP アドレス、データグラムの長さ、データグラムのシーケンス番号が含まれます。これらの情報が付加されるのは、データグラムがネットワークパケットとしての許容バイトサイズを超過してフラグメント化が必要になった場合に備えるためです。

データリンク層: フレーミングの実行

データリンク層のプロトコル (PPP など) は、IP データグラムを「フレーム」という形式に設定します。これらのプロトコルは、第 3 のヘッダーとフッターを付加することにより、データグラムを「フレーミング」します。フレームヘッダーには、フレームがネットワークメディアを通過するときのエラーを検査するための、「巡回冗長検査」(CRC) フィールドが含まれています。次に、データリンク層は物理層にフレームを渡します。

物理ネットワーク層: フレームの送受信

送信側ホストの物理ネットワーク層は、フレームを受信し、IP アドレスをネットワークメディアに適切なハードウェアアドレスに変換します。次に、物理ネットワーク層は、フレームをネットワークメディアに送り出します。

受信側ホストでのパケットの取り扱い

パケットが受信側ホストに到達すると、パケットは TCP/IP プロトコルスタックを送信時とは逆順に通過します。このパスを図 1–1 に示します。受信側ホストの各プロトコルは、送信側ホストの対等プロトコルがパケットに付加したヘッダー情報を取り除きます。

この処理の順序を次に示します。

1. 物理ネットワーク層は、フレーム形式のパケットを受信します。パケットの CRC を計算し、データリンク層にフレームを送信します。
2. データリンク層は、フレームの CRC が正しいことを確認すると、フレームのヘッダーとその CRC を取り除きます。最後に、データリンクプロトコルは、インターネット層にフレームを送ります。
3. インターネット層は、ヘッダーにある情報を読み取って、転送を識別します。そして、パケットがフラグメントであるかどうかを判別します。その転送がフラグメントである場合は、IP は、フラグメントを組み立て直して、オリジナルのデータグラムに戻します。そして、IP ヘッダーを取り除いてから、データグラムをトランスポート層プロトコルに渡します。
4. トランスポート層 (TCP、SCTP および UDP) は、ヘッダーを読み取って、どのアプリケーション層プロトコルでデータを受信する必要があるかを決定します。次に、TCP、SCTP または UDP は、自分に関連するヘッダーを取り除き、メッセージまたはストリームを受信アプリケーションに送信します。
5. アプリケーション層はメッセージを受信して、送信側ホストから要求された操作を実行します。

TCP/IP 内部トレース機能のサポート

TCP/IP は、RST パケットにより接続が終了したときに、TCP 通信のログを記録することで内部トレースをサポートします。RST パケットが送信または受信されたときに、直前に送受信された最大 10 パケットの情報が接続情報とともにログに記録されます。

TCP/IP とインターネットについてもっと詳しく知るには

TCP/IP やインターネットに関する情報は、幅広く入手できます。このドキュメントで説明していない特別な情報は、次に挙げる情報源からも入手できます。

TCP/IP に関するコンピュータ関連の書籍

TCP/IP やインターネットに関する多くの書籍が近所の図書館やコンピュータ関係の書店で入手できます。

次の 2 冊は、TCP/IP に関する基本的な書籍と考えられています。

- Craig Hunt 著『TCP/IP Network Administration 』 - 異種 TCP/IP ネットワークの管理について、ある程度の理論と、豊富な実践的情報が含まれています。
- W. Richard Stevens 著『TCP/IP Illustrated, Volume I』 - TCP/IP のプロトコルが詳細に解説されています。これは、TCP/IP に関する技術的な背景知識を必要とするネットワーク管理者およびネットワークプログラマにとって最適です。

TCP/IP とネットワーク設定に関する Web サイト

インターネットには、TCP/IP プロトコルとその管理に特化した多数の Web サイトおよびユーザーグループがあります。Oracle Corporation などの多くの製造元は、一般的な TCP/IP 情報に関する Web ベースの参考情報を提供しています。次に、TCP/IP 情報と一般的なシステム管理情報に関する有益な Web 上の参考情報を示します。次の表に、関連する Web サイトの一覧と、各サイトが提供するネットワーク情報についての説明を示します。

Web サイト	説明
The Internet Engineering Task Force (IETF) の Web サイト (`http://www.ietf.org/home.html`)	IETF は、インターネットのアーキテクチャーおよび管理に対して責任を負う団体です。IETF の Web サイトには、IETF のさまざまな活動に関する情報が含まれています。また、IETF の主な出版物へのリンクも含まれています。
Oracle Corporation's BigAdmin Portal (`http://www.oracle.com/technetwork/systems/index.html`)	BigAdmin は、Sun コンピュータの管理に関する情報を提供しています。このサイトには、FAQ、リソース、ディスカッション、ドキュメントへのリンクなど、ネットワーク設定を含めた Oracle Solaris の管理に関する情報があります。

RFC と Internet Draft

インターネットエンジニアリングタスクフォース (IETF) ワーキンググループは、「*Requests for Comments*」(RFC) として知られる標準文書を公開しています。作成途中の標準は、「Internet Draft」で公開されています。IAB (Internet Architecture Board) は、パブリックドメインで公開する前にすべての RFC を承認する必要があります。通常、RFC と Internet Draft は、開発者および非常に技術力の高い読者向けです。しかしながら、TCP/IP に関する項目に関係する RFC の多くには、システム管理者にとって有益な情報が含まれています。これらの RFC は、このドキュメントのさまざまな場所で引用されています。

通常、参考情報 (FYI) 文書は RFC のサブセットとして公開されています。FYI には、インターネット規格を取り扱うような内容は含まれていません。むしろ、インターネットのもっと一般的な性格に関する情報を扱うものです。たとえば、FYI 文書には、TCP/IP の入門書や資料の目録、また、あらゆるインターネット関連のソフトウェアツールを網羅した要覧、インターネットと一般的なネットワーキングに関する用語集などが含まれています。

このドキュメントおよび Oracle Solaris System Administrator Collection のほかのマニュアルのさまざまな場所に、関連する RFC への参照が含まれています。

パート II

TCP/IP の管理

このパートでは、TCP/IP ネットワークを構成、管理、および障害追跡するための手順および概念情報について説明します。

◆ ◆ ◆ 第 2 章

TCP/IP ネットワークの計画 (手順)

この章では、コスト効率のよい整然とした方法でネットワークを構築するために解決しておく必要のある事柄について説明します。これらの事柄を解決後、ネットワークを構成し管理するための計画を立てることができます。

この章では、次の内容について説明します。

- 55 ページの「ネットワークハードウェアの決定」
- 58 ページの「ネットワークの IP 番号の取得」
- 55 ページの「ネットワークの IP アドレス指定形式の決定」
- 64 ページの「ネットワーク上のエンティティーへの名前付け」
- 67 ページの「ネットワーク上でのルーターの計画」

ネットワークを構成するタスクについては、第 5 章「TCP/IP ネットワークサービスと IPv4 アドレス指定の構成 (作業)」を参照してください。

ネットワーク計画 (タスクマップ)

次の表に、ネットワークを構成するための各種作業の一覧を示します。表では、各タスクで実行する内容の説明と、タスクの具体的な実行手順が詳しく説明されている現在のマニュアル内のセクションを示しています。

タスク	説明	参照先
1. ハードウェア要件とネットワークトポロジを計画します	必要な機器の種類とサイトの機器レイアウトを決定します。	■ ネットワークトポロジに関する一般的な疑問点については、55 ページの「ネットワークハードウェアの決定」を参照してください。 ■ IPv6 トポロジの計画については、91 ページの「IPv6 をサポートするためのネットワークトポロジの準備」を参照してください。 ■ 特定の種類の機器については、機器メーカーの文書を参照してください。
2. ネットワークの登録 IP アドレスを取得します	インターネットなど、自分のローカルネットワークの外部と通信を行う計画の場合、ネットワークには一意の IP アドレスが必要となります。	58 ページの「ネットワークの IP 番号の取得」を参照してください。
3. IPv4 ネットワーク接頭辞または IPv6 サイト接頭辞に基づいたシステムの IP アドレス指定スキームを作成します。	サイトでのアドレスの割り振りを決定します。	55 ページの「ネットワークの IP アドレス指定形式の決定」または95 ページの「IPv6 アドレス指定計画の準備」を参照してください。
4. ネットワーク上のすべてのマシンの IP アドレスとホスト名を含むリストを作成します。	このリストを使用してネットワークデータベースを構築します	65 ページの「ネットワークデータベース」を参照してください
5. ネットワークで、どのネームサービスを使用するかを決定します。	ローカルの /etc ディレクトリで、NIS、LDAP、DNS、またはネットワークデータベースを使用するかどうかを決定します。	65 ページの「ネームサービスとディレクトリサービスの選択」を参照してください
6. 必要に応じて、管理作業を分担するための区分を設定します。	サイトで、管理作業を分担するための区分にネットワークを分割する必要があるかどうかを決定します	66 ページの「管理作業の分化」を参照してください
7. ネットワーク設計でのルーターの位置を決定します。	ネットワークがルーターを必要とする大きさの場合は、ルーターをサポートするネットワークトポロジを作成します。	67 ページの「ネットワーク上でのルーターの計画」を参照してください

タスク	説明	参照先
8. 必要な場合、サブネット戦略を策定します。	IP アドレス空間を管理するためや、ユーザーが使用できる IP アドレスを増やすために、サブネットの作成が必要な場合があります。	IPv4 でのサブネット計画については、248 ページの「サブネット化とは」を参照してください IPv6 サブネットの計画については、96 ページの「サブネット用の番号付けスキームの作成」を参照してください

ネットワークハードウェアの決定

ネットワークを設計する場合、組織のニーズにもっとも合うネットワークの種類を決定する必要があります。計画段階の決定事項には、次のネットワークハードウェアを含めてください。

- ネットワークトポロジ、ネットワークハードウェアのレイアウトと接続
- ネットワークがサポートするホストシステムの数
- ネットワークがサポートするホストの種類
- 必要になる可能性があるサーバーの種類
- 使用するネットワークメディアの種類。 たとえば、Ethernet、トークンリング、FDDI など
- このメディアを拡張してローカルネットワークを外部ネットワークに接続するためにブリッジまたはルーターが必要かどうか
- いくつかのシステムで、組み込みインタフェースに加えて別途購入したインタフェースが必要かどうか

これらの要因に基づいて、ローカルエリアネットワークのサイズを決定できます。

注 – ネットワークハードウェアの計画方法については、このマニュアルでは取り上げません。ハードウェアに付属しているマニュアルを参照してください。

ネットワークの IP アドレス指定形式の決定

サポートする予定のシステムの数は、ネットワークの構成方法に影響を与えます。組織によっては、1 つの階または 1 つのビルの中にある数十台のスタンドアロンシステムから成る小さいネットワークが必要な場合もあります。また、複数のビルに散在する 1000 以上のシステムを持つネットワークの設定が必要な場合もあります。このような大きい設定の場合は、ネットワークを「サブネット」と呼ばれる小区分に分割することが必要になる場合もあります。

ネットワークアドレス指定スキームを計画する場合は、次の要因を考慮してください。

- 使用する IP アドレスの種類 (IPv4 または IPv6)
- ネットワーク上の潜在的なシステムの数
- 独立した IP アドレスを持つ複数のネットワークインタフェースカード (NIC) を必要とする、マルチホームまたはルーターとなるシステムの数
- ネットワークでプライベートアドレスを使用するかどうか
- IPv4 アドレスのプールを管理する DHCP サーバーを使用するかどうか

1990 年以来、世界中にインターネットが広まり、使用可能な IP アドレスが不足してきました。この状況を改善するために、インターネットエンジニアリングタスクフォース (IETF) は、多数の代替 IP アドレス指定を開発してきました。

組織に 2 つ以上のネットワーク用 IP アドレスが割り当てられている、または組織がサブネットを使用している場合は、組織内の中央の管理者にネットワーク IP アドレスの割り当てを担当させます。この責任者が、割り当てられたネットワーク IP アドレスのプールを管理する権限を保持し、ネットワーク、サブネット、ホストアドレスを必要に応じて割り当てます。問題の発生を避けるために、組織内に重複したネットワーク番号や無秩序なネットワーク番号が生じないことを確認してください。今日使用されている IP アドレスの種類には、次のようなものがあります。

IPv4 アドレス

この 32 ビットのアドレスは、TCP/IP 向けに設計された元々の IP アドレス指定書式です。もともと、IP ネットワークには A、B、C という 3 つのクラスがあります。ネットワークに割り当てられるネットワーク番号は、このクラス指定にホストを表す 8 ビット以上を加えたものになります。クラスベースの IPv4 アドレスでは、ネットワーク番号のネットマスクを構成する必要があります。さらに、ローカルネットワーク上のシステムで使用可能なアドレスを増やす目的で、これらのアドレスは、多くの場合サブネットに分割されます。

今日、IP アドレスは「*IPv4* アドレス」と呼ばれます。ISP からクラスベースの IPv4 ネットワーク番号を入手することはできなくなりましたが、既存ネットワークの多くは、まだこの番号を持っています。IPv4 アドレスの管理については、60 ページの「IPv4 アドレス指定スキームの設計」を参照してください。

CIDR 書式の IPv4 アドレス

IETF は、短期または中期的に IPv4 アドレスの不足に対処するために CIDR (Classless Inter-Domain Routing) アドレスを開発しました。さらに、CIDR 書式は、世界的なインターネット経路制御テーブルの容量不足の解決も意図しています。CIDR 表記の

IPv4 アドレスは、長さが 32 ビットで、同じドット付き 10 進数の書式を持っています。ただし、CIDR は、IPv4 アドレスのネットワーク部を定義するために、右端のバイトのあとに接頭辞指定を追加します。詳細については、61 ページの「IPv4 CIDR アドレス指定スキームの設計」を参照してください。

DHCP アドレス

動的ホスト構成プロトコル (DHCP) を使用すると、システムは、ブートプロセスの一環として、IP アドレスなどの構成情報を DHCP サーバーから受け取ることができます。DHCP サーバーは、DHCP クライアントに割り当てるアドレスの入った IP アドレスのプールを格納しています。そのため、DHCP を使用する 1 つのサイト用の IP アドレスプールは、すべてのクライアントに常時 IP アドレスを割り当てた場合に比べて、小さくなります。DHCP サービスを設定すると、サイトの IP アドレスまたはアドレスの一部を管理できます。詳細については、第 12 章「DHCP について (概要)」を参照してください。

IPv6 アドレス

使用可能な IPv4 アドレスの不足の長期的解決策として、IETF は 128 ビットの IPv6 アドレスを導入しました。IPv6 アドレスは、IPv4 で使用できるものより大きなアドレス空間を提供します。Oracle Solaris では、デュアルスタック TCP/IP を使用することで、同一ホスト上での IPv4 と IPv6 のアドレス指定をサポートしています。CIDR 形式の IPv4 アドレスと同様、IPv6 アドレスにはネットワーククラスまたはネットマスクの考え方はありません。CIDR と同様、IPv6 アドレスは接頭辞を使用して、サイトのネットワークを定義するアドレスの部分を指定します。IPv6 の紹介については、76 ページの「IPv6 アドレス指定の概要」を参照してください。

プライベートアドレスとドキュメントの接頭辞

IANA では、プライベートネットワークで使用するために、IPv4 アドレスのブロックと IPv6 サイト接頭辞が予約されています。これらのアドレスは、企業ネットワーク内のシステムに割り振ることができますが、プライベートアドレスを持つパケットは、インターネットでは経路制御できないことに注意してください。プライベートアドレスの詳細については、63 ページの「プライベート IPv4 アドレスの使用」を参照してください。

注 – プライベート IPv4 アドレスは、文書化目的でも予約されています。このドキュメントの例では、プライベート IPv4 アドレスと予約 IPv6 文書接頭辞を使用します。

ネットワークの IP 番号の取得

IPv4 ネットワークは、IPv4 ネットワーク番号とネットワークマスク、つまり「ネットマスク」を組み合わせて定義されます。IPv6 ネットワークは、「サイト接頭辞」、およびサブネット化されている場合は、「サブネット接頭辞」で定義されます。

ネットワークを永遠にプライベートにするつもりでない限り、そのネットワークのローカルユーザーは、大部分の場合、ローカルネットワークの外と通信する必要があります。したがって、ネットワークが外部と通信できるように、適切な組織からネットワークの登録 IP 番号を取得する必要があります。取得したアドレスが、IPv4 アドレス指定スキームのネットワーク番号または IPv6 アドレス指定スキームのサイト接頭辞となります。

インターネットサービスプロバイダは、複数のサービスレベルを基準にした課金体系によって、ネットワークの IP アドレスを提供します。各 ISP を調査して、どこが自分のネットワークに最も合ったサービスを提供しているのかを決定します。一般的に ISP は、企業に対して動的に割り当てられるアドレスまたは静的 IP アドレスを提供します。IPv4 アドレスと IPv6 アドレスの両方を提供する ISP もあります。

自分が ISP の場合は、自分のロケールのインターネットレジストリ (IR) から、顧客用の IP アドレスを取得します。インターネットアサインドナンバーオーソリティー (IANA) は、世界中で登録 IP アドレスの IR への委託に対して最終的な責任を負います。各 IR には、IR がサービスを提供するロケールの登録情報とテンプレートが含まれています。IANA とその IR については、IANA の IP Address Service のページ (`http://www.iana.org/ipaddress/ip-addresses.htm`)を参照してください。

注 – 現在、使用しているネットワークを外部の TCP/IP ネットワークに接続していなくても、勝手な IP アドレスを割り当てないでください。代わりに、63 ページの「プライベート IPv4 アドレスの使用」に説明があるプライベートアドレスを使用してください。

IPv4 アドレス指定スキームの設計

注 – IPv6 アドレスの計画については、95 ページの「IPv6 アドレス指定計画の準備」を参照してください。

このセクションでは、IP v 4 アドレス指定計画を設計する助けとなるように、IP v 4 アドレス指定の概要について説明します。IPv6 アドレスについては、76 ページの「IPv6 アドレス指定の概要」を参照してください。DHCP アドレスについては、第 12 章「DHCP について (概要)」を参照してください。

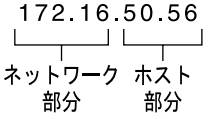
IPv4 ベースの各ネットワークには、次のものが必要です。

- ISP、IR によって、または IANA で登録された旧式のネットワークに対して割り当てられた一意のネットワーク番号。プライベートアドレスを使用する予定の場合、作成するネットワーク番号は組織内で一意でなければなりません。
- ネットワーク上の各システムのインタフェースの一意の IPv4 アドレス。
- ネットワークマスク。

IPv4 アドレスは 32 ビットの数値で、63 ページの「ネットワークインタフェースへの IP アドレスの適用法」に説明されているように、システム上のネットワークインタフェースを一意に識別します。IPv4 アドレスは 10 進数で表され、ピリオドで区切った 4 つの 8 ビットフィールドに分割されます。個々の 8 ビットフィールドは、それぞれ IPv4 アドレスの 1 バイトを表します。このような形式で IPv4 アドレスのバイトを表す方式を「ドット化 *10* 進形式」と呼びます。

次の図に、IPv4 アドレス `172.16.50.56` のコンポーネント部を示します。

図 2–1　IPv4 アドレス形式



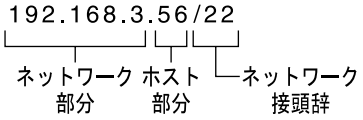
`172.16`　登録 IPv4 ネットワーク番号。クラスベースの IPv4 表記では、この番号が IP ネットワーククラスも定義します。この例では、クラスは、IANA によって登録されたクラス B です。

`50.56`　IPv4 アドレスのホスト部。ホスト部は、ネットワーク上のシステムのインタフェースを一意に識別します。ローカルネットワーク上の各インタフェースについて、アドレスのネットワーク部は同じで、ホスト部はそれぞれ異なる必要があります。

クラスベースの IPv4 ネットワークをサブネット化する予定の場合は、サブネットマスク、つまり、「ネットマスク」を定義する必要があります (248 ページの「`netmasks` データベース」を参照)。

次の例は、CIDR 形式のアドレス `192.168.3.56/22` を示します。

図 2–2 CIDR 形式の IPv4 アドレス



192.168.3 ISP または IR から受け取った IPv4 ネットワーク番号で構成されるネットワーク部。

56 システムのインタフェースに割り当てるホスト部。

/22 ネットワーク番号を構成するアドレスのビット数を定義するネットワーク接頭辞。ネットワーク接頭辞は、IP アドレスのサブネットマスクも提供します。また、ISP または IR によっても割り当てられます。

Oracle Solaris ベースのネットワークでは、標準の IPv4 アドレス、CIDR 形式の IPv4 アドレス、DHCP アドレス、IPv6 アドレス、およびプライベート IPv4 アドレスを組み合わせることができます。

IPv4 アドレス指定スキームの設計

この節では、標準 IPv4 アドレスを整理するクラスについて説明します。IANA はクラスベースのネットワーク番号の配布をやめていますが、これらのネットワーク番号は今でも多くのネットワークで使用されています。クラスベースのネットワーク番号を持つサイトでは、アドレス空間の管理が必要な場合もあります。IPv4 ネットワーククラスの詳細については、262 ページの「ネットワーククラス」を参照してください。

次の表は、標準 IPv4 アドレスがどのようにネットワークアドレス空間とホストアドレス空間に分かれるかを示しています。どのクラスについても、「範囲」の欄は、ネットワーク番号の最初のバイトの 10 進数値の範囲を示しています。「ネットワークアドレス」は、IPv4 アドレスの中でネットワーク部の働きをするバイト数を示します。*xxx* は 1 バイトを表します。「ホストアドレス」は、アドレスのホスト部を表すバイト数を示します。たとえばクラス A ネットワークアドレスの場合は、最初の 1 バイトがネットワーク番号で、残りの 3 バイトがホスト番号です。クラス C ネットワークの場合はこの関係が逆になり、最初の 3 バイトがネットワーク番号で、残りの 1 バイトがホスト番号です。

表 2–1 IPv4 クラスの分割

クラス	バイトの範囲	ネットワーク番号	ホストアドレス
A	0–127	*xxx*	*xxx.xxx. xxx*
B	128–191	*xxx.xxx*	*xxx.xxx*

表 2–1 IPv4 クラスの分割 (続き)

クラス	バイトの範囲	ネットワーク番号	ホストアドレス
C	192–223	*xxx.xxx. xxx*	*xxx*

IP v 4 アドレスで 1 番目のバイトの番号は、ネットワークがクラス A、B、C のいずれであるのかを定義します。残りの 3 バイトの範囲は 0 - 255 です。0 と 255 の 2 つは予約されています。1 から 254 までの数字は、IANA によってネットワークに割り当てられたネットワーククラスに従って、各バイトに割り当てることができます。

次の表は、IPv4 アドレスのどのバイトがユーザーに割り当てられているかを示しています。また、ホストへの割り当てが可能な、各バイト内の値の範囲を示します。

表 2–2 割り当て可能な IPv4 クラスの範囲

ネットワーククラス	バイト 1 の範囲	バイト 2 の範囲	バイト 3 の範囲	バイト 4 の範囲
A	0–127	1–254	1–254	1–254
B	128–191	IANA によって事前割り当て	1–254	1–254
C	192–223	IANA によって事前割り当て	IANA によって事前割り当て	1–254

IPv4 サブネット番号

多数のホストを持つローカルネットワークは、サブネットに分割される場合があります。IPv4 ネットワーク番号をサブネットに分割した場合は、ネットワーク識別子を各サブネットに割り当てる必要があります。IPv4 アドレスのホスト部の一部のビットをネットワーク識別子として使用することで、IPv4 アドレス空間の有効率を最大限にできます。ネットワーク識別子として使用した場合、アドレスの指定した部分がサブネット番号になります。サブネット番号は、ネットマスクを使って作成します。ネットマスクは、IPv4 アドレスのネットワーク部とサブネット部を選択するビットマスクです。詳細は、249 ページの「IPv4 アドレス用のネットワークマスクの作成」を参照してください。

IPv4 CIDR アドレス指定スキームの設計

元々 IPv4 を構成していたネットワーククラスは、現在インターネットでは使用されていません。現在は、IANA は、クラスを持たない CIDR 形式のアドレスを世界中のレジストリに配布しています。ISP から提供される IPv4 アドレスはすべて CIDR 形式です (図 2–2 を参照)。

CIDR アドレスのネットワーク接頭辞は、ネットワーク上のホストに対して割り当て可能な IPv4 アドレスの数を示しています。これらのホストアドレスは、ホストのインタフェースに割り当てられます。ホストに複数の物理インタフェースがある場合は、使用されているすべての物理インタフェースにホストアドレスを割り当てる必要があります。

CIDR アドレスのネットワーク接頭辞は、サブネットマスクの長さも定義します。大部分の Oracle Solaris コマンドは、ネットワークのサブネットマスクの CIDR 接頭辞の指定を認識します。ただし、Oracle Solaris インストールプログラムおよび `/etc/netmask` ファイルでは、ドット付き 10 進数表現を使用して、サブネットマスクを設定する必要があります。この 2 つの場合、次の表に示されているように、CIDR ネットワーク接頭辞のドット付き 10 進数表現を使用してください。

表 2–3　CIDR 接頭辞と 10 進数での表現

CIDR ネットワーク接頭辞	使用可能な IP アドレス	ドット付き 10 進数でのサブネット表現
/19	8,192	255.255.224.0
/20	4,096	255.255.240.0
/21	2,048	255.255.248.0
/22	1024	255.255.252.0
/23	512	255.255.254.0
/24	256	255.255.255.0
/25	128	255.255.255.128
/26	64	255.255.255.192
/27	32	255.255.255.224

CIDR アドレスについては、次の文書を参照してください。

- CIDR の技術的な詳細については、RFC 1519, Classless Inter-Domain Routing (CIDR): an Address Assignment and Aggregation Strategy (`http://www.ietf.org/rfc/rfc1519.txt?number=1519`) を参照してください。
- CIDR に関するより一般的な情報は、Pacific Bell のサイトの Classless Inter-Domain Routing (CIDR) Overview (`http://www.wirelesstek.com/cidr.htm`) で入手できます。
- Wikipedia の記事『"Classless inter-domain routing" (`http://en.wikipedia.org/wiki/Classless_inter-domain_routing`)』にも CIDR の概要が記載されています。

プライベート IPv4 アドレスの使用

IANA は、企業がプライベートネットワークのために使用できるように IPv4 アドレスの 3 つのブロックを予約しています。これらのアドレスは、RFC 1918, Address Allocation for Private Internets (`http://www.ietf.org/rfc/rfc1918.txt?number=1918`) に定義されています。1918 アドレスとしても知られるこれらの「プライベートアドレス」は、企業のイントラネット内のローカルネットワーク上のシステムに使用できます。ただし、インターネット上ではプライベートアドレスは無効です。ローカルネットワークの外部とも通信する必要があるシステムでは使用しないでください。

次の表に、プライベート IPv4 アドレスの範囲と、各範囲に対応するネットマスクの一覧を示します。

IPv4 アドレス範囲	ネットマスク
10.0.0.0 - 10.255.255.255	10.0.0.0
172.16.0.0 - 172.31.255.255	172.16.0.0
192.168.0.0 - 192.168.255.255	192.168.0.0

ネットワークインタフェースへの IP アドレスの適用法

ネットワークと接続するには、システムに 1 つ以上の「物理ネットワークインタフェース」が必要になります。各ネットワークインタフェースは、それぞれ一意な IP アドレスを持っていなければなりません。Oracle Solaris のインストール時には、インストールプログラムが最初に検出したインタフェースの IP アドレスを設定する必要があります。通常、そのインタフェースは、*eri0* または `hme0` などの `device-name0` という名前を持ちます。このインタフェースは、「プライマリネットワークインタフェース」とみなされます。

ホストに 2 番目のネットワークインタフェースを追加する場合は、そのインタフェースにも一意の IP アドレスが必要です。2 番目のネットワークインタフェースを追加すると、ホストは「マルチホーム」になります。一方、2 番目のネットワークインタフェースをホストに追加し、IP 転送を有効にした場合は、そのホストはルーターになります。説明は、124 ページの「IPv4 ルーターの構成」を参照してください。

`/devices` ディレクトリには、各ネットワークインタフェースのデバイス名、デバイスドライバ、関連するデバイスファイルが入っています。ネットワークインタフェースのデバイス名には、たとえば `le0` または `smc0` などがあります。これらは、よく使用される 2 つの Ethernet インタフェースのデバイス名です。

インタフェースに関する情報および作業については、第 6 章「ネットワークインタフェースの管理 (作業)」を参照してください。

注 – このドキュメントは、システムに Ethernet ネットワークインタフェースがあることを想定しています。別のネットワークメディアを使用する予定の場合は、そのネットワークインタフェースのマニュアルの中の構成に関する情報を参照してください。

ネットワーク上のエンティティーへの名前付け

割り当てられたネットワーク IP アドレスを受け取り、その IP アドレスでシステムの NIC をすべて構成すると、次のタスクはホストへ名前を割り当てることです。ここで、ネットワーク上のネームサービスをどのように扱うかを決める必要があります。これらの名称は最初にネットワークを設定する場合や、後日ルーター、ブリッジまたは PPP を使ってネットワークを拡張する場合に使います。

TCP/IP は、ネットワーク上の特定のシステムを見つけるときに、そのシステムの IP アドレスを使用します。ただし、認識可能な名前を使用すると、そのシステムを簡単に識別できます。したがって、TCP/IP プロトコル (および Oracle Solaris) では、システムを一意なものとして識別するために、IP アドレスとホスト名の両方が必要です。

TCP/IP の視点から見れば、ネットワークは名前が付けられたエンティティーの集合です。ホストは名前が付けられた 1 個のエンティティーです。ルーターも名前が付けられた 1 個のエンティティーです。さらに、ネットワークも名前が付けられた 1 個のエンティティーです。ネットワークがインストールされているグループや部門にも、名前を付けることができます。部課、地区、会社も同様です。理論的には、ネットワークを識別するために使用できる名前の階層については、事実上まったく制限はありません。このドメイン名で「ドメイン」が特定されます。

ホスト名の管理

多くのサイトでは、ユーザーはマシンのホスト名を選択できます。サーバーにも少なくとも 1 つのホスト名が必要で、このホスト名はプライマリネットワークインタフェースの IP アドレスに関連付けられます。

システム管理者は、自己の管轄ドメイン内のすべてのホスト名が一意なものであることを確認する必要があります。言い換えると、ネットワーク上の 2 つのマシンが同じ「fred」という名前を持つことはできません。ただし、「fred」というマシンが複数の IP アドレスを持つことは可能です。

ネットワークの計画を立てるときは、IP アドレスとそれぞれのホスト名のリストを作って、設定工程中に各マシンに簡単にアクセスできるようにしてください。このリストは、すべてのホスト名が一意かどうかを検査するために役立ちます。

ネームサービスとディレクトリサービスの選択

Oracle Solaris を使用すると、3 種類のネームサービスを使用できます。3 つのネームサービスとは、ローカルファイル、NIS、DNS です。ネームサービスは、ネットワーク上のマシンに関する重要な情報、たとえばホスト名、IP アドレス、Ethernet アドレスなどを保持しています。Oracle Solaris では、ネームサービスに加えて、またはネームサービスの代わりに LDAP ディレクトリサービスを使用することもできます。Oracle Solaris のネームサービスについては、『Solaris のシステム管理 (ネーミングとディレクトリサービス : DNS、NIS、LDAP 編)』のパート I「ネームサービスとディレクトリサービスについて」を参照してください。

ネットワークデータベース

オペレーティングシステムをインストールするときに、手順の一環として、サーバー、クライアント、スタンドアロンシステムのホスト名および IP アドレスを指定します。Oracle Solaris インストールプログラムは、この情報を `hosts` に追加し、Solaris 10 11/06 以前の Solaris 10 リリースでは `ipnodes` ネットワークデータベースにも追加します。このデータベースは、ネットワーク上の TCP/IP の動作に必要な情報を含んでいるネットワークデータベースセットの一部です。管理者が自己のネットワーク用として選択したネームサービスは、これらのデータベースを読み取ります。

ネットワークデータベースの構成は重要です。したがって、ネットワーク計画工程の一環として、どのネームサービスを使用するかを決定する必要があります。ネームサービスの使用の決定は、ネットワークを管理ドメインとして編成するかどうかにも影響を与えます。252 ページの「ネットワークデータベースと `nsswitch.conf` ファイル」は、ネットワークデータベースのセットを詳細に説明しています。

ネームサービスとしての NIS または DNS の使用

NIS および DNS ネームサービスは、ネットワーク上の複数のサーバーのネットワークデータベースを格納しています。これらのネームサービス、およびデータベースの構成方法については、『Solaris のシステム管理 (ネーミングとディレクトリサービス : DNS、NIS、LDAP 編)』を参照してください。このガイドでは、「名前空間」と「管理ドメイン」の概念についても詳しく説明されています。

ネームサービスとしてのローカルファイルの使用

NIS、LDAP または DNS を実行しないと、ネットワークは、「ローカルファイル」を使用してネームサービスを提供します。「ローカルファイル」とは、ネットワークデータベースが使用するためのものとして `/etc` ディレクトリに入っている一連のファイルのことです。このドキュメントに示す手順では、特に断らない限り、ネームサービスとしてローカルファイルを使用しているものとします。

注 – ネットワーク用のネームサービスとしてローカルファイルを使用することに決めた場合、後日、別のネームサービスを設定することもできます。

ドメイン名

多くのネットワークは、ホストとルーターを管理ドメインの階層で整理しています。NIS または DNS のネームサービスを使用する場合は、所属組織のドメイン名として、全世界の中で一意な名前を選択する必要があります。ドメイン名が一意であることを確認するには、そのドメイン名を InterNIC に登録する必要があります。DNS を使う予定がある場合は、必ず選択したドメイン名も登録します。

ドメイン名は階層構造になっています。一般に、新規のドメインは、既存の関連するドメインの下に配置されます。たとえば、子会社のドメイン名はその親会社のドメイン名の下に配置されます。特にほかとの関連性のない組織のドメイン名は、既存の最上位ドメインのいずれかの下に直接配置できます。

次に最上位ドメインの例を 2、3 示します。

- `.com` – 民間企業 (世界規模)
- `.edu` – 教育機関 (世界規模)
- `.gov` – アメリカ政府機関
- `.fr` – フランス

その名前が一意であるという制限を守って、自分の組織を識別する名前を選択します。

管理作業の分化

管理作業の分業化の問題は、規模と制御の問題に関係します。ネットワーク内のホストとサーバーの数が増えるに従って、管理タスクはますます複雑になります。このような状況に対処するための方法としては、管理部門を増設することが考えられます。そのためには、特定のクラスのネットワークを増設したり、既存のネットワークをサブネットに分割したりします。

ネットワーク管理の作業を分化するかどうかは、次の要因によって判断します。

- ネットワークの規模

  数百台のホストから構成される単一のネットワークは、すべてのホストが物理的に同じ場所にありしかも同じ管理サービスを必要とする場合は、1 つの管理部門で対処できます。しかし、場合によっては、複数の管理部門を設立する必要があります。サブネットを持つ小規模ネットワークが地理的に広い範囲に散在している場合は、複数の管理部門を設立する方法が効率的です。

- ネットワーク上のユーザーのニーズが共通しているかどうか

たとえば、1 つのビル内にあり、比較的少数のマシンをサポートするネットワークを使用しているとします。これらのマシンはいくつかのサブネットワークに分割されています。各サブネットワークは、異なるニーズを持つユーザーのグループをサポートします。このような場合は、サブネットごとに管理部門を設立するとよいでしょう。

ネットワーク上でのルーターの計画

TCP/IP では、ネットワークに 2 種類のエンティティーがあったことを思い出してください。 つまりホストとルーターだけです。ホストはすべてのネットワークに必要ですが、ルーターはすべてのネットワークに必要なわけではありません。ネットワークの物理的なトポロジによってルーターを使用する必要があるかどうかが決まります。この節では、ネットワークトポロジとルーティングの概念を紹介します。これらの概念は、既存のネットワーク環境に別のネットワークを追加すると決めた場合に重要になります。

注 – IPv4 ネットワークでのルーター構成の詳細と手順については、117 ページの「IPv4 ネットワーク上でのパケット転送と経路制御」を参照してください。IPv6 ネットワークでのルーター構成の詳細とタスクについては、183 ページの「IPv6 ルーターの構成」を参照してください。

ネットワークトポロジの概要

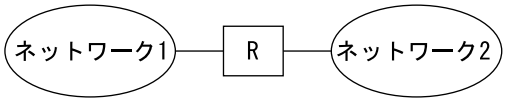
ネットワークトポロジは、ネットワークの組み合わせ方を定義します。ルーターは、ネットワークを相互に接続するエンティティーです。ルーターは、複数のネットワークインタフェースを持ち、IP 転送を実行するマシンです。ただし、124 ページの「IPv4 ルーターの構成」で説明されているとおりに正しく構成されるまで、システムはルーターとして機能できません。

ルーターは、複数のネットワークに接続して、より大きなインターネットワークを形成します。ルーターは、隣接する 2 つのネットワーク間でパケットの受け渡しをするように構成する必要があります。さらに、隣接するネットワークを越えた位置にあるネットワークに、パケットを渡す機能も備えられている必要があります。

次の図に、ネットワークトポロジの基本部分を示します。最初の図は、2 つのネットワークを 1 台のルーターで接続した単純な構成です。2 番目の図は、3 つのネットワークを 2 台のルーターで相互接続した構成を示しています。最初の例では、ルーター R がネットワーク 1 とネットワーク 2 を連結して、より大きなインターネットワークを作っています。2 番目の例では、ルーター R1 はネットワーク 1 と 2 に接続し、ルーター R2 は、ネットワーク 2 と 3 に接続しています。この接続で、ネットワーク 1、2、3 を含むネットワークが形成されます。

図2–3 基本的なネットワークトポロジ

1 つのルーターによって接続されている 2 つのネットワーク

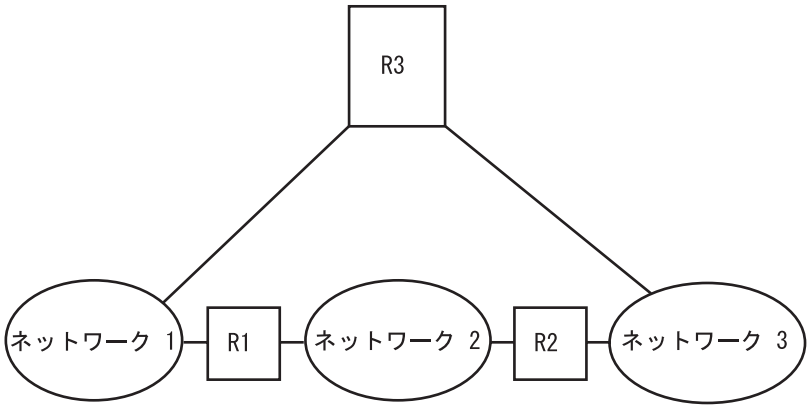


2 つのルーターによって接続されている 3 つのネットワーク

ネットワークをインターネットワークに結合したあと、ルーターは、宛先ネットワークのアドレスを基にネットワーク間でパケットの経路制御を行います。インターネットワークがより複雑になるにつれて、ルーターがパケットの宛先を決定する回数は増加します。

次の図にさらに複雑な例を示します。ルーター R3 は、ネットワーク 1 と 3 に直接接続されており、この冗長性によって信頼性が向上します。ネットワーク 2 が停止しても、ルーター R3 は、ネットワーク 1 と 3 の間にルートを提供できます。 多くのネットワークを相互接続することが可能です。ただし、相互接続するネットワークは、同じネットワークプロトコルを使う必要があります。

図 2–4 ネットワーク間に追加パスを提供するネットワークトポロジ



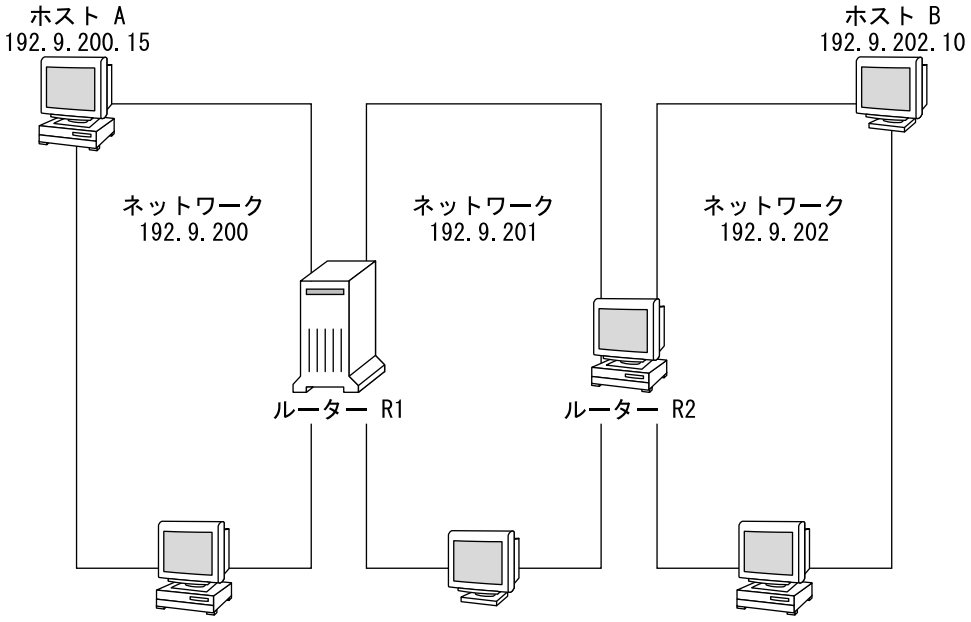
ルーターがどのようにパケットを転送するか

受信側の IP アドレスはパケットヘッダーに含まれ、パケットのルーティング方法を決定します。このアドレスにローカルネットワークのネットワーク番号が含まれている場合は、その IP アドレスを持つホストに直接パケットが送られます。ネットワーク番号がローカルネットワークではない場合は、パケットはローカルネットワーク上のルーターに送られます。

ルーターは、「ルーティングテーブル」にルーティング情報を格納します。このテーブルには、ルーターが接続されているネットワーク上のホストとルーターの IP アドレスが含まれています。また、それらのネットワークを指すポインタも含まれています。ルーターは、パケットを受信すると、ルーティングテーブルを調べて、ヘッダー内の宛先アドレスがテーブルにリストされているかどうかを確認します。テーブルにその宛先アドレスが含まれていない場合は、ルーターは、ルーティングテーブルにリストされているほかのルーターにパケットを転送します。ルーターの詳細については、124 ページの「IPv4 ルーターの構成」を参照してください。

次の図は、2 つのルーターにより接続された 3 つのネットワークのネットワークトポロジを示します。

図 2–5 3つの相互接続ネットワークを持つネットワークトポロジ



ルーター R1 は、ネットワーク 192.9.200 とネットワーク 192.9.201 を接続しています。ルーター R2 は、ネットワーク 192.9.201 とネットワーク 192.9.202 と接続しています。

ネットワーク 192.9.200 のホスト A がネットワーク 192.9.202 のホスト B にメッセージを送る場合、次のイベントが発生します。

1. ホスト A は、ネットワーク 192.9.200 にパケットを送り出します。パケットヘッダーには、受信側ホスト B の IPv4 アドレスである 192.9.202.10 が含まれています。
2. ネットワーク 192.9.200 には、192.9.202.10 の IPv4 アドレスを持つマシンはありません。したがって、ルーター R1 がパケットを受け取ります。
3. ルーター R1 は自己のルーティングテーブルを調べます。ネットワーク 192.9.201 には、アドレスが 192.9.202.10 であるマシンはありません。ただし、ルーティングテーブルにはルーター R2 がリストされています。
4. R1 は「次のホップ」ルーターとして R2 を選択し、パケットを R2 に送信します。
5. R2 はネットワーク 192.9.201 を 192.9.202 に接続するため、R2 にはホスト B のルーティング情報が含まれています。その後、ルーター R2 はパケットをネットワーク 192.9.202 に転送し、その場所でホスト B がそのパケットを受け入れます。

◆ ◆ ◆ 第 3 章

IPv6 の紹介 (概要)

この章では、Oracle Solaris Internet Protocol version 6 (IPv6) 実装の概要について説明します。この実装には、IPv6 アドレス空間をサポートする関連のデーモンやユーティリティーも含まれます。

IPv6 アドレスと IPv4 アドレスは Oracle Solaris ネットワーク環境に共存します。IPv4 アドレスがすでに存在する場合、IPv6 アドレスで構成されたシステムは IPv4 アドレスを保持します。IPv6 アドレスに関連する操作が IPv4 の操作に悪影響を与えることはなく、IPv4 の操作が IP v 6 の操作に悪影響を与えることはありません。

この章では、主に次の内容について説明します。

IPv6 の詳細については、次の章を参照してください。

- IPv6 ネットワークの計画 – 第 4 章「IPv6 ネットワークの計画 (手順)」
- IPv6 関連のタスク – 第 7 章「IPv6 ネットワークの構成 (手順)」および第 8 章「TCP/IP ネットワークの管理 (手順)」
- IPv6 の詳細 – 第 11 章「IPv6 の詳細 (リファレンス)」

IPv6 の主な特長

IPv6 の主な特長は、IPv4 と比べてアドレス空間が増えたことです。IPv6 はまた、インターネット機能をさまざまな分野で強化します。この節では、この強化された機能について概説します。

拡張されたアドレス

IPv6 では、IP アドレスのサイズが IPv4 の 32 ビットから 128 ビットに拡張され、より多くのレベルの指定階層をサポートできます。IPv6 はさらに、より多くのアドレス可能な IPv6 システムを提供します。詳細については、76 ページの「IPv6 アドレス指定の概要」を参照してください。

アドレスの自動構成と近傍検索

IPv6 の「近傍検索 *(ND)*」プロトコルは、IPv6 アドレスの自動構成を容易にします。「自動構成」とは、IPv6 ホストが自分の IPv6 アドレスを自動的に生成できるという機能のことです。自動構成を使用すると、アドレス管理が簡単になり、時間もかからなくなります。詳細については、84 ページの「IPv6 アドレスの自動構成」を参照してください。

近傍検索プロトコルは、次に示す IPv4 プロトコルの組み合わせに対応します。 つまり、Address Resolution Protocol (ARP)、Internet Control Message Protocol (ICMP)、Router Discovery (RDISC)、および ICMP Redirect です。IPv6 ルーターは近傍検索を使用して、IPv6 サイト接頭辞を通知します。IPv6 ホストは、さまざまな目的のために近傍検索を使用します。これには、IPv6 ルーターからの接頭辞の要請が含まれます。詳細については、83 ページの「IPv6 近傍検索プロトコルの概要」を参照してください。

ヘッダーフォーマットの簡略化

IPv6 ヘッダーフォーマットは、一部の IPv4 のヘッダーフィールドを削除したり、任意にしたりします。IPv6 のアドレスのサイズは増えましたが、ヘッダーフォーマットの簡略によって、IPv6 のヘッダーのサイズは極力減りました。IPv6 アドレスの長さは、IPv4 アドレスの 4 倍ですが、IPv6 ヘッダーのサイズは IPv4 の 2 倍に抑えられています。

IP ヘッダーオプションのサポートの強化

IP ヘッダーオプションをコード化する方法を変更したため、転送がより効率よく行われるようになりました。また、IPv6 オプションの長さに関する制限が緩和されました。この変更によって、将来新しいオプションを導入する際の柔軟性が高くなりました。

IPv6 アドレス指定のアプリケーションのサポート

多くの Oracle Solaris の重要なネットワークサービスは、IPv6 アドレスを認識およびサポートします。

- たとえば、DNS、LDAP、および NIS などのネームサービスです。これらのネームサービスにおける IPv6 のサポートの詳細については、『Solaris のシステム管理 (ネーミングとディレクトリサービス : DNS、NIS、LDAP 編)』を参照してください。
- 認証および機密アプリケーション。IP Security Architecture (IPsec) や Internet Key Exchange (IKE) など。詳細については、パート IV「IP セキュリティー」を参照してください。
- 差別化サービス。IP Quality of Service (IPQoS) で提供されます。詳細については、パート VI「IP サービス品質 (IPQoS)」を参照してください。
- フェイルオーバー検出。IP ネットワークマルチパス (IPMP) で提供されます。詳細については、パート V「IPMP」を参照してください。

IPv6 に関する追加リソース

このパートのほかにも、IPv6 については、次の情報も参照してください。

IPv6 RFC と Internet Draft

IPv6 については、多数の RFC が入手可能です。次の表に、本書執筆時点での IPv6 に関する主な記事と、その記事が掲載されている IETF (Internet Engineering Task Force) の Web サイトを示します。

表 3–1　IPv6 関連の RFC と Internet Draft

RFC または Internet Draft	件名	場所
RFC 2461、Neighbor Discovery for IP Version 6 (IPv6)	IPv6 近傍検索プロトコルの特長と機能について説明します。	http://www.ietf.org/rfc/rfc2461.txt$number=2461 (http://www.ietf.org/rfc/rfc2461.txt?number-2461)

表 3–1 IPv6 関連の RFC と Internet Draft (続き)

RFC または Internet Draft	件名	場所
RFC 3306、Unicast-Prefix-Based IPv6 Multicast Addresses	IPv6 マルチキャストアドレスのフォーマットとタイプについて説明します。	ftp://ftp.rfc-editor.org/in-notes/rfc3306.txt (ftp://ftp.rfc-editor.org/in-notes/rfc3306.txt)
RFC 3484: Default Address Selection for Internet Protocol version 6 (IPv6)	IPv6 のデフォルトのアドレスを選択するときに使用されるアルゴリズムについて説明します。	http://www.ietf.org/rfc/rfc3484?number=3484 (http://www.ietf.org/rfc/rfc3484.txt?number=3484)
RFC 3513、Internet Protocol version 6 (IPv6) Addressing Architecture	IPv6 アドレスのタイプについて詳細に説明し、その例を示します。	http://www.ietf.org/rfc/rfc3513.txt?number=3513 (http://www.ietf.org/rfc/rfc3513.txt?number=3513)
RFC 3587、 IPv6 Global Unicast Address Format	IPv6 ユニキャストアドレスの標準フォーマットを定義します。	http://www.ietf.org/rfc/rfc3587.txt?number=3587 (http://www.ietf.org/rfc/rfc3587.txt?number=3587)

Web サイト

次の Web サイトには、IPv6 に関する有用な情報があります。

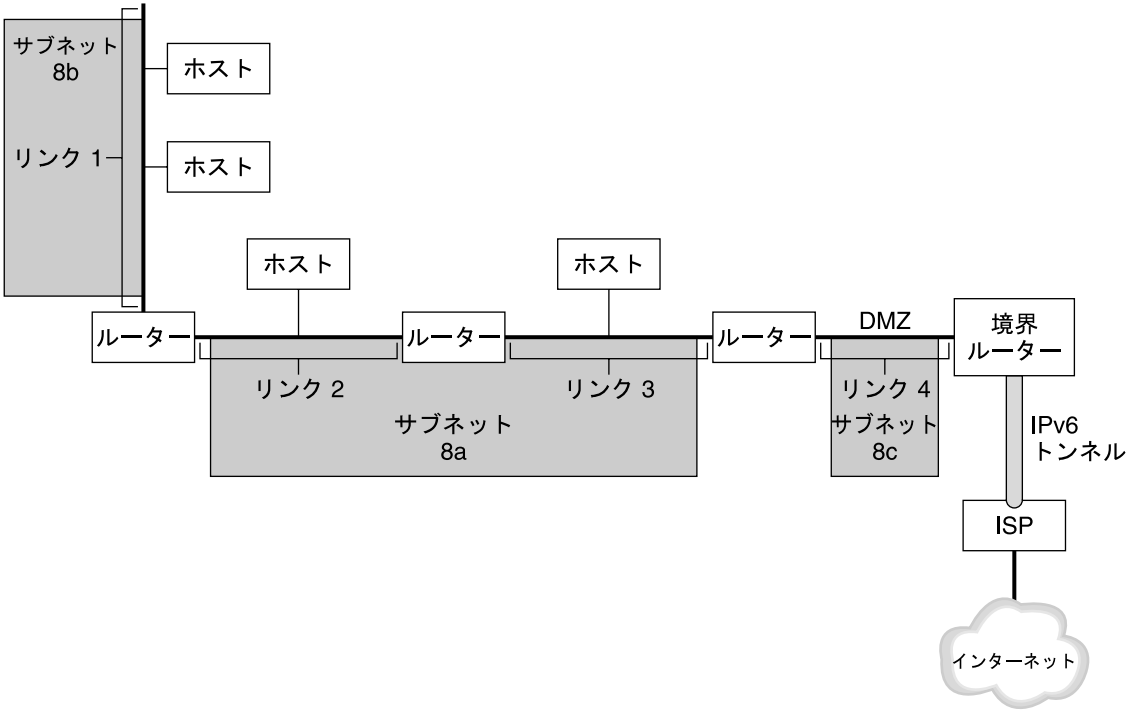
表 3–2 IPv6 関連の Web サイト

Web サイト	説明	場所
IPv6 Forum	この Web サイトからは、世界中の IPv6 関連のプレゼンテーション、イベント、クラス、および実装へのリンクが利用できます。	http://www.ipv6forum.com
Internet Educational Task Force IPv6 Working Group	この Web サイトからは、すべての関連する IPv6 RFC および Internet Draft へのリンクが利用できます。	http://www.ietf.org/html.charters/ipv6-charter.html

IPv6 ネットワークの概要

この節では、IPv6 ネットワークトポロジの基本である用語を紹介します。次の図に、IPv6 ネットワークの基本部分を示します。

図 3–1 IPv6 ネットワークの基本部分



この図は、IPv6 ネットワークとその ISP への接続を示します。内部ネットワークはリンク 1、リンク 2、リンク 3、およびリンク 4 から構成されます。各リンクはホストによって生成され、ルーターによって終了します。リンク 4 は内部ネットワークの DMZ であり、片方の終端は境界ルーターによって終了します。境界ルーターは ISP への IPv6 トンネルを実行して、内部ネットワークにインターネット接続を提供します。リンク 2 とリンク 3 はサブネット 8a として管理されます。サブネット 8b はリンク 1 上のシステムだけから構成されます。サブネット 8c はリンク 4 で DMZ につながります。

図 3–1 に示すとおり、IPv6 ネットワークのコンポーネントは IPv4 ネットワークと本質的に同じです。しかし、IPv6 の専門用語は IPv4 と少しだけ異なります。次に、IPv6 で使用されるネットワーク構成要素の専門用語を示します。

ノード
: IPv6 アドレスと IPv6 サポート用に構成されたインタフェースを持つシステム。この用語は汎用であり、ホストとルーターの両方に使用されます。

IPv6 ルーター
: IPv6 パケットを転送するノード。少なくとも、ルーターのインタフェースの 1 つは IPv6 サポート用に構成されている必要があります。また IPv6 ルーターは、内部ネットワーク経由で、企業の登録済み IPv6 サイト接頭辞を通知できます。

IPv6 ホスト	IPv6 アドレスを持つノード。IPv6 ホストは、IPv6 サポート用に構成されたインタフェースを複数持つことができます。IPv4 と同様に、IPv6 ホストはパケットを転送しません。
リンク	単一の連続するネットワークメディア。一方の終端にルーターが存在します。
近傍	ローカルノードと同じリンク上にある IPv6 ノード。
IPv6 サブネット	IPv6 ネットワークの管理セグメント。IPv4 と同様に、IPv6 サブネットの構成要素はリンク上のすべてのノードに直接対応付けることができます。必要に応じて、リンク上のノードは別個のサブネット内で管理できます。さらに、IPv6 はマルチリンクサブネットをサポートします。マルチリンクサブネットでは、複数のリンク上の複数のノードを単一のサブネットの構成要素にできます。図 3–1 のリンク 2 とリンク 3 は、マルチリンクサブネット 8a のコンポーネントです。
IPv6 トンネル	ある IPv6 ノードとほかの IPv6 ノードのエンドポイント間に仮想 P2P パスを提供するトンネル。IPv6 は、手動構成可能なトンネルと自動 6to4 トンネルをサポートします。
境界ルーター	ネットワークの終端にあり、IPv6 トンネルの片側をローカルネットワーク外のエンドポイントに提供するルーター。境界ルーターは、内部ネットワークへの IPv6 インタフェースを少なくとも 1 つ持っている必要があります。外部ネットワークのルーターは、IPv6 インタフェースまたは IPv4 インタフェースのどちらを持っていてもかまいません。

IPv6 アドレス指定の概要

1 つのノードが複数のインタフェースを持つことができるため、IPv6 アドレスは、ノードではなく、インタフェースに割り当てます。さらに、1 つのインタフェースに複数の IPv6 アドレスを割り当てることも可能です。

注 – IPv6 アドレスフォーマットについての技術的な詳細情報については、RFC 2374、IPv6 Global Unicast Address Format (`http://www.ietf.org/rfc/rfc2374.txt?number=2374`) を参照してください。

IPv6 は 3 つのタイプのアドレスを定義します。

ユニキャスト	個々のノードの 1 つのインタフェースを識別します。

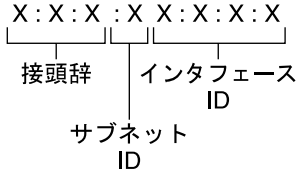
マルチキャスト　複数のインタフェースの 1 つのグループ (通常は、異なるノードにある) を識別します。マルチキャストアドレスに送信されたパケットは、この「マルチキャストグループ」のすべてのメンバーに送信されます。

エニーキャスト　複数のインタフェースの 1 つのグループ (通常は、異なるノードにある) を識別します。エニーキャストアドレスに送信されたパケットは、この「エニーキャストグループ」のうち、送信者に物理的に最も近いメンバーに送信されます。

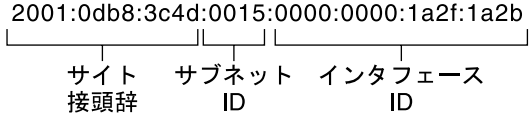
IPv6 アドレスの構成部分

IPv6 アドレスは、長さが 128 ビットで、8 つの 16 ビットフィールドから構成され、各フィールドはコロンで区切られます。各フィールドには、IPv4 アドレスのドット区切り表記ではなく、16 進数値で指定する必要があります。次の図において、x は 16 進数値を表します。

図 3–2　基本的な IPv6 アドレスフォーマット



例:



左の 3 つのフィールド (48 ビット) には、「サイト接頭辞」が含まれます。サイト接頭辞は「公開トポロジ」を表します。公開トポロジは、通常、ISP または RIR (Regional Internet Registry) がサイトに割り当てます。

真ん中の 1 つのフィールド (16 ビット) には、「サブネット *ID*」が入ります。サブネット ID は、管理者 (またはほかの管理者) が自分のサイトに割り当てます。サブネット ID は「プライベートトポロジ」を表します。サイトの内部を示すため、「サイトトポロジ」と呼ぶこともあります。

いちばん右の 4 つのフィールド (64 ビット) にはインタフェース *ID* (トークンとも呼ばれる) が格納されます。インタフェース ID は、インタフェースの MAC アドレスから自動的に構成されるか、EUI-64 フォーマットで手動で構成されます。

もう一度、図 3–2を検討します。

```
2001:0db8:3c4d:0015:0000:0000:1a2f:1a2b
```

この例は、IPv6 アドレスの 128 ビットすべてを示します。最初の 48 ビット `2001:0db8:3c4d` はサイト接頭辞を含み、公開トポロジを表します。次の 16 ビット `0015` はサブネット ID を含み、そのサイトのプライベートトポロジを表します。右端の最後の 64 ビット `0000:0000:1a2f:1a2b` はインタフェース ID を含みます。

IPv6 アドレスの省略

ほとんどの IPv6 アドレスは、128 ビットすべてが使用されるわけではありません。結果として、フィールド内の使用されていないビットはゼロで埋められます (ゼロだけが含まれます)。

IPv6 アドレスアーキテクチャーでは、2 つのコロン (`::`) 表記で、連続するゼロだけの 16 ビットフィールドを表すことができます。たとえば、インタフェース ID にある 2 つの連続するゼロのフィールドを 2 つのコロンで置き換えることで、図 3–2 の IPv6 アドレスを短縮できる可能性があります。その結果、アドレスは `2001:0db8:3c4d:0015::1a2f:1a2b` のようになります。その他のゼロだけのフィールドは、単一の 0 として表現できます。また、フィールド内の先頭のゼロは省略できます。たとえば、`0db8` は `db8` のように省略できます。

したがって、`2001:0db8:3c4d:0015:0000:0000:1a2f:1a2b` というアドレスは、`2001:db8:3c4d:15::1a2f:1a2b` のように省略できます。

2 つのコロン表記を使用すると、IPv6 アドレス内のゼロだけの連続するフィールドを表現できます。たとえば、 `2001:0db8:3c4d:0015:0000:d234::3eee:0000` という IPv6 アドレスは `2001:db8:3c4d:15:0:d234:3eee::` のように省略できます。

IPv6 の接頭辞

IPv6 アドレスの左側は接頭辞を含みます。この接頭辞は、IPv6 パケットの経路制御に使用されます。IPv6 接頭辞のフォーマットは次のとおりです。

接頭辞/長さ *(ビット数)*

接頭辞の長さは CIDR (クラスレス相互ドメイン経路制御) 表記で表現されます。CIDR 表記とは、アドレスの終わりにスラッシュを付けて、その後ろに接頭辞の

長さ (ビット数) を表します。CIDR フォーマットの IP アドレスについては、61 ページの「IPv4 CIDR アドレス指定スキームの設計」を参照してください。

IPv6 アドレスの「サイト接頭辞」は、IPv6 アドレスの左側の 48 ビット (最大) です。たとえば、`2001:db8:3c4d:0015:0000:0000:1a2f:1a2b/48` という IPv6 アドレスのサイト接頭辞は、左側の 48 ビットである `2001:db8:3c4d` に含まれています。この接頭辞を表現するには、ゼロを圧縮して、次のような表記を使用します。

```
2001:db8:3c4d::/48
```

注 – `2001:db8::/32` という接頭辞は、この例だけで使用される特別な IPv6 接頭辞です。

「サブネット接頭辞」を指定すると、ルーターまでの内部ネットワークトポロジも定義できます。次に、IPv6 のサブネット接頭辞の例を示します。

```
2001:db8:3c4d:15::/64
```

サブネット接頭辞は常に 64 ビットを含んでいます。サブネット接頭辞の 64 ビットには、サイト接頭辞の 48 ビットとサブネット ID の 16 ビットが含まれます。

次の接頭辞は、特別な目的のために予約されています。

`2002::/16` 6to4 経路制御接頭辞が続くことを示します。

`fe80::/10` リンクローカルアドレスが続くことを示します。

`ff00::/8` マルチキャストアドレスが続くことを示します。

ユニキャストアドレス

IPv6 のユニキャストアドレスには、2 つの種類があります。

- グローバルユニキャストアドレス
- リンクローカルアドレス

ユニキャストアドレスの種類は、アドレス内の左側の連続するビット (接頭辞が含まれるところ) によって判断されます。

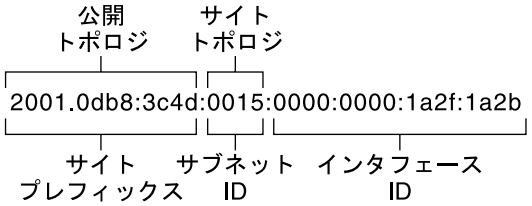
ユニキャストアドレスフォーマットは、次の階層から構成されます。

- 公開トポロジ
- サイト (プライベート) トポロジ
- インタフェース ID

グローバルユニキャストアドレス

グローバルユニキャストアドレスはインターネット内でグローバルに一意です。78 ページの「IPv6 の接頭辞」に、グローバルユニキャストアドレスである IPv6 アドレスの例を示します。次の図に、IPv6 アドレスの一部と比較して、グローバルユニキャストアドレスの有効範囲を示します。

図 3–3 グローバルユニキャストアドレスの一部



公開トポロジ

サイト接頭辞は、ルーターまでのネットワークの「公開トポロジ」を定義します。企業のサイト接頭辞は、ISP または RIR (Regional Internet Registry) から取得します。

サイトトポロジと IPv6 サブネット

IPv6 では、「サブネット *ID*」はネットワークの管理サブネットを定義し、その長さは 16 ビット (最大) です。サブネット ID は、IPv6 ネットワーク構成の一部として割り当てます。「サブネット接頭辞」は、サブネットが割り当てられる特定のリンクを指定することによって、ルーターへのサイトトポロジを定義します。

IPv6 サブネットは、概念的には IPv4 サブネットと同じであり、各サブネットは通常、単一のハードウェアリンクに関連付けられます。しかし、IPv6 サブネット ID は、ドット付き 10 進数表記ではなく、16 進数表記で表現されます。

インタフェース ID

「インタフェース *ID*」は、特定のノードのインタフェースを識別します。インタフェース ID はサブネット内で一意である必要があります。IPv6 ホストは、近傍検索プロトコルを使用して、自分のインタフェース ID を自動的に生成します。近傍検索は、ホストのインタフェースの MAC アドレスまたは EUI-64 アドレスに基づいて、インタフェース ID を自動的に生成します。インタフェース ID は手動でも割り当てることができます。IPv6 ルーターと、IPv6 が有効なサーバーには、インタ

フェース ID を手動で割り当てることが推奨されます。手動で EUI-3513 アドレスを作成する方法については、RFC 3513、Internet Protocol Version 6 (IPv6) Addressing Architecture を参照してください。

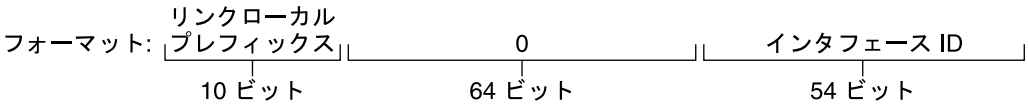
移行用グローバルユニキャストアドレス

移行のため、IPv6 プロトコルは IPv4 アドレスを IPv6 アドレス内に埋め込む機能を持っています。この種類の IPv4 アドレスは、既存の IPv4 ネットワークへの IPv6 パケットのトンネルを容易にします。移行用グローバルユニキャストアドレスの例としては、6to4 アドレスがあります。6to4 アドレスの詳細については、299 ページの「6to4 自動トンネル」を参照してください。

リンクローカルユニキャストアドレス

リンクローカルユニキャストアドレスを使用できるのは、ローカルネットワークリンク上だけです。リンクローカルアドレスは、企業の外では有効でなく、認識されません。次に、リンクローカルアドレスのフォーマットの例を示します。

例 3–1　リンクローカルユニキャストアドレスの一部



例: fe80::123e:456d

「リンクローカル接頭辞」のフォーマットは、次のとおりです。

`fe80::`*interface-ID*`/10`

次に、リンクローカルアドレスの例を示します。

```
fe80::23a1:b152
```

`fe80`	10 ビットの 2 進数接頭辞 1111111010 の 16 進数表記。この接頭辞は、IPv6 アドレスの種類がリンクローカルであると識別します。
interface-ID	インタフェースの 16 進数アドレス。通常は 48 ビットの MAC アドレスから生成されます。

Oracle Solaris インストール中に IPv6 を有効にした場合、ローカルマシン上の最も低い番号のインタフェースはリンクローカルアドレスで構成されます。ローカルリンクにおいて、あるリンクをほかのリンクから識別するために、各インタフェースは少なくとも 1 つのリンクローカルアドレスを必要とします。したがって、ノードに

インタフェースを追加した場合、リンクローカルアドレスを手動で構成する必要があります。構成後、このノードはそのリンクローカルアドレスを使用して、アドレスの自動構成や近傍検索を行います。

マルチキャストアドレス

IPv6 は、マルチキャストアドレスの使用をサポートします。マルチキャストアドレスは、「マルチキャストグループ」を識別します。マルチキャストグループとは、通常は異なるノード上にあるインタフェースのグループのことです。1 つのインタフェースが所属できるマルチキャストグループは複数設定できます。IPv6 アドレスの最初の 16 ビットが `ff00`*n* である場合、そのアドレスはマルチキャストアドレスです。

マルチキャストアドレスは、マルチキャストグループのメンバーとして定義されているすべてのインタフェースに情報やサービスを送信するのに使用されます。たとえば、ローカルリンク上のすべての IPv6 ノードと通信するときなどです。

あるインタフェースに IPv6 ユニキャストアドレスを作成するとき、カーネルは自動的にそのインタフェースを特定のマルチキャストグループのメンバーにします。たとえば、カーネルはすべてのノードを Solicited Node マルチキャストグループのメンバーにします。このグループは、近傍検索プロトコルが到達できるかどうかを検出するときに使用されます。また、カーネルはすべてのノードを All-Nodes または All Ruters マルチキャストグループのメンバーにします。

マルチキャストアドレスの詳細については、268 ページの「IPv6 マルチキャストアドレスの詳細」を参照してください。技術的な情報については、RFC 3306, Unicast-Prefix-based IPv6 Multicast Addresses (`ftp://ftp.rfc-editor.org/in-notes/rfc3306.txt`) を参照してください。この RFC は、マルチキャストアドレスのフォーマットについて説明します。マルチキャストアドレスとマルチキャストグループの適切な使用方法の詳細については、RFC 3307, Allocation Guidelines for IPv6 Multicast Addresses (`ftp://ftp.rfc-editor.org/in-notes/rfc3307.txt`) を参照してください。

エニーキャストアドレスとエニーキャストグループ

IPv6 エニーキャストアドレスは、異なる IPv6 ノード上のインタフェースのグループを識別します。このようなインタフェースの各グループのことを「エニーキャストグループ」と呼びます。エニーキャストアドレスにパケットが送信されると、エニーキャストグループのうち、送信者に物理的に最も近いメンバーがそのパケットを受信します。

注 – IPv6 の Oracle Solaris 実装は、エニーキャストアドレスやグループの作成をサポートしていません。しかし、Oracle Solaris IPv6 ノードはエニーキャストアドレスにパケットを送信できます。詳細については、301 ページの「6to4 リレールーターとの間のトンネルについての考慮事項」を参照してください。

IPv6 近傍検索プロトコルの概要

IPv6 は、メッセージングを近傍ノード間の対話を処理する手段として使用する近傍検索機能を導入します。「近傍ノード」とは、同じリンク上にある IPv6 ノードのことです。たとえば、近傍検索関係のメッセージを発行すると、ノードは近傍ノードのリンクローカルアドレスを知ることができます。

近傍検索は、主に、次のような IPv6 ローカルリンク上の活動を制御します。

- ルーターの発見 – ホストがローカルリンク上のルーターを検出できるようにします。
- アドレスの自動構成 – ノードが自分のインタフェースの IPv6 アドレスを自動構成できるようにします。
- 接頭辞の検出 – ノードがリンクに割り当てられている既知のサブネット接頭辞を検出できるようにします。ノードはサブネット接頭辞を使用して、ローカルリンク上の宛先と、ルーター経由でのみ到達可能な宛先を区別します。
- アドレスの解決 – ノードが宛先の IP アドレスだけから近傍ノードのリンクローカルアドレスを判断できるようにします。
- 次のホップの判断 – ノードがローカルリンク外のパケット受信ノードの IP アドレスを判断できるようにします。次のホップは、ルーターまたは宛先ノードのどちらかである可能性があります。
- 近傍不到達の検索 – ノードが近傍に到達できないことを判断できるようにします。ルーターとホストの場合、アドレスの解決は繰り返すことができます。
- 重複アドレス検出 – 使用したいアドレスが使用中でないことを、ノードが判断できるようにします。
- リダイレクト – 特定の宛先に到達するために使用するとよい最初のホップノードを、ルーターがホストに通知できるようにします。

近傍検索は、リンク上のノード間で通信するとき、次の ICMP メッセージタイプを使用します。

- ルーターの要請
- ルーター広告 (RA: Router Advertisement)
- 近傍の要請
- 近傍の通知

- リダイレクト

近傍検索メッセージや近傍検索プロトコルの詳細については、287 ページの「IPv6 近傍検索プロトコル」を参照してください。近傍検索の技術的な情報については、RFC 2461, Neighbor Discovery for IP Version 6 (IPv6) (`http://www.ietf.org/rfc/rfc2461.txt?number=2461`) を参照してください。

IPv6 アドレスの自動構成

IPv6 の主な特長は、ホストがインタフェースを自動構成できることです。ホストは近傍検索を使用して、ローカルリンク上の IPv6 ルーターを見つけて、サイト接頭辞を要求します。自動構成プロセスの一部として、ホストは次のことを行います。

- 各インタフェースのリンクローカルアドレスを作成します。リンク上にルーターは必要ありません。
- リンク上におけるアドレスの一意性を確認します。リンク上にルーターは必要ありません。
- グローバルアドレスを取得するときに、ステートレスメカニズム、ステートフルメカニズム、またはその両方のメカニズムを使用するかどうかを判断します。リンク上にルーターが必要です。

ステートレス自動構成の概要

ステートレス自動構成では、手動によるホストの構成は不要です。ルーターは最小限の構成 (あれば) ですみ、サーバーの追加も不要です。ステートレスメカニズムにより、ホストは独自のアドレスを生成できます。ステートレスメカニズムは、アドレスを生成するために、ローカルな情報とルーターが通知する情報を使用します。

インタフェースには、一時アドレスを実装できます。この一時アドレスも自動構成されます。一時アドレストークンは、ホスト上の 1 つまたは複数のインタフェースに対して有効にできます。しかし、標準の自動構成された IPv6 アドレスとは異なり、一時アドレスは、サイト接頭辞とランダムに生成された 64 ビット数から構成されます。このランダムな数は、IPv6 アドレスのインタフェース ID 部分になります。リンクローカルアドレスでは、一時アドレスはインタフェース ID としては生成されません。

ルーターは、リンクに割り当てられているすべての接頭辞を通知します。IPv6 ホストは近傍検索を使用して、ローカルルーターからサブネット接頭辞を取得します。ホストは、インタフェースの MAC アドレスから生成されたインタフェース ID とサブネット接頭辞を組み合わせることによって、自動的に IPv6 アドレスを作成し

ます。ルーターがない場合、ホストはリンクローカルアドレスだけを生成します。リンクローカルアドレスは、同じリンク上のノード間の通信でのみ使用できます。

注 – ステートレス自動構成を使用して、サーバーの IPv6 アドレスを作成しないでください。ホストは、自動構成中、ハードウェア固有の情報に基づいて、インタフェース ID を自動的に生成します。既存のインタフェースが新しいインタフェースに切り替わると、現在のインタフェース ID は無効になる可能性があります。

IPv6 トンネルの概要

ほとんどの企業では、既存の IPv4 ネットワークに IPv6 を導入するとき、徐々に段階的に行う必要があります。Oracle Solaris のデュアルスタックネットワーク環境は IPv4 と IPv6 の両方の機能をサポートします。ほとんどのネットワークは IPv4 プロトコルを使用するため、現在のところ、IPv6 ネットワークは IPv6 ネットワークの境界を越えて通信する方法を持つ必要があります。この通信のためには、IPv6 ネットワークはトンネルを使用します。

ほとんどの IPv6 トンネルシナリオでは、送信される IPv6 パケットは IPv4 パケット内にカプセル化されます。IPv6 ネットワークの境界ルーターは、宛先 IPv6 ネットワークの境界ルーターに向かうさまざまな IPv4 ネットワークにポイントツーポイントトンネルを設定します。パケットはトンネルを通って、宛先ネットワークの境界ルーターに到着し、その境界ルーターでカプセル化が解除されます。次に、その境界ルーターは個々の IPv6 パケットを宛先のノードに転送します。

Oracle Solaris IPv6 実装がサポートするトンネルシナリオは、次のとおりです。

- IPv4 ネットワーク経由で、2 つの IPv6 ネットワーク間に手動でトンネルを構成する場合。この場合、IPv4 ネットワークはインターネットでも、企業内のローカルネットワークでもかまいません。
- IPv6 ネットワーク経由で、IPv4 ネットワーク間に手動でトンネルを構成する場合 (通常は企業内)。
- 企業の IPv4 ネットワークまたはインターネット経由で、2 つの IPv6 ネットワーク間に自動 6to4 トンネルを動的に構成する場合。

IPv6 トンネルの詳細については、295 ページの「IPv6 トンネル」を参照してください。IPv4- to-IPv4 トンネルと VPN については、520 ページの「仮想プライベートネットワークと IPsec」を参照してください。

◆ ◆ ◆ 第 4 章

IPv6 ネットワークの計画 (手順)

新しいネットワークまたは既存のネットワークに IPv6 を配備するには、大規模な計画を立案する必要があります。この章では、IPv6 を自分のサイトに構成する前に行う必要がある計画タスクについて説明します。既存のネットワークへの IPv6 の配備は、ゆっくりと着実に行う必要があります。この章では、IPv4 専用ネットワークに IPv6 を段階的に導入する方法を説明します。

この章の内容は次のとおりです。

IPv6 の概念の概要については、第 3 章「IPv6 の紹介 (概要)」を参照してください。詳細については、第 11 章「IPv6 の詳細 (リファレンス)」を参照してください。

IPv6 の計画 (タスクマップ)

次のタスクマップに、IPv6 の配備に必要な計画タスクを達成するためのタスクを順番に示します。

次の表に、IPv6 ネットワークを構成するための各種作業の一覧を示します。表では、各タスクで実行する内容の説明と、タスクの具体的な実行手順が詳しく説明されている現在のマニュアル内のセクションを示しています。

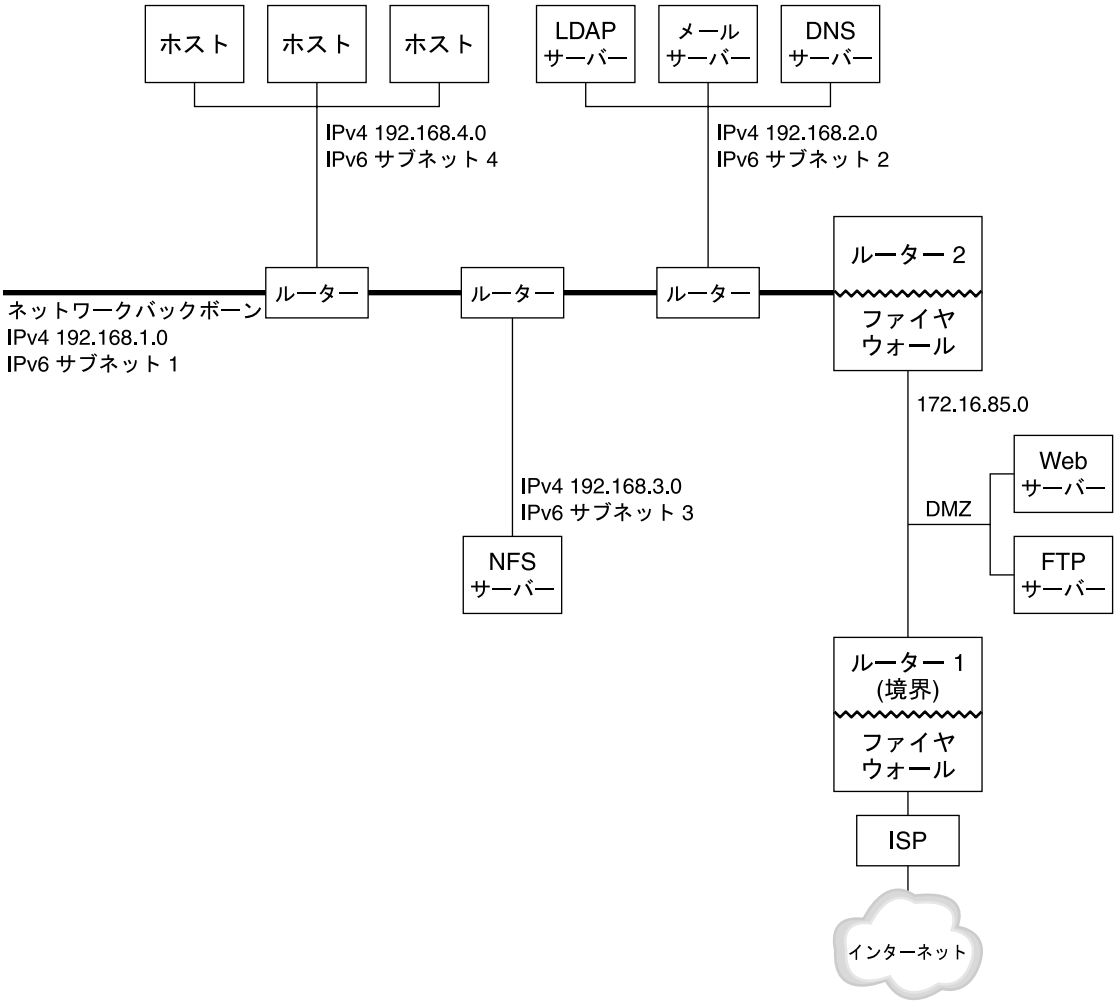
タスク	説明	説明
1. IPv6 をサポートするようにハードウェアを準備します。	IPv6 をサポートできるようにハードウェアをアップグレードします。	91 ページの「IPv6 をサポートするためのネットワークトポロジの準備」

タスク	説明	説明
2. IPv6 をサポートする ISP を取得します。	現在利用している ISP が IPv6 をサポートしていることを確認します。それ以外の場合、IPv6 をサポートできる ISP を探してください。IPv6 での通信用に 1 つ、IPv4 通信用に 1 つと、2 つの ISP を利用してもかまいません。	
3. アプリケーションが IPv6 をサポートすることを確認します。	使用するアプリケーションが IPv6 環境で動作できることを確認します。	92 ページの「IPv6 をサポートするためにネットワークサービスを準備する方法」
4. サイト接頭辞を取得します。	ISP または最も近い RIR から、当該サイト用の 48 ビットのサイト接頭辞を取得します。	95 ページの「サイト接頭辞の取得」
5. サブネットのアドレス指定を計画します。	IPv6 をネットワーク上のさまざまなノードに構成する前に、全体的な IPv6 ネットワークトポロジとアドレス指定スキームを計画する必要があります。	96 ページの「サブネット用の番号付けスキームの作成」
6. トンネルの使用について計画します。	ほかのサブネットまたは外部ネットワークへのトンネルを実行するルーターを判断します。	94 ページの「ネットワークトポロジにおけるトンネルの計画」
7. ネットワーク上のエンティティーへのアドレス指定を計画します。	IPv6 を構成する前に、サーバー、ルーター、およびホストへのアドレス指定スキームを計画する必要があります。	96 ページの「ノードの IPv6 アドレス指定計画の立案」
8. IPv6 セキュリティーポリシーを開発します。	IPv6 セキュリティーポリシーを開発すると同時に、IP Filter、IP セキュリティーアーキテクチャー (IPsec)、Internet Key Exchange (IKE) などの Oracle Solaris セキュリティー機能を調査します。	パート IV「IP セキュリティー」
9. (任意) DMZ を設定します。	IPv6 を構成する前に、セキュリティーのため、DMZ およびそのエンティティーへのアドレス指定を計画する必要があります。	94 ページの「IPv6 実装のセキュリティーについて」
10. IPv6 をサポートするようにノードを有効にします。	すべてのルーターおよびホスト上で IPv6 を構成します。	183 ページの「IPv6 ルーターの構成 (タスクマップ)」
11. ネットワークサービスを起動します。	既存のサーバーが IPv6 をサポートできることを確認します。	210 ページの「主な TCP/IP 管理タスク (タスクマップ)」
12. IPv6 をサポートするようにネームサーバーを更新します。	新しい IPv6 アドレスをサポートするように、DNS、NIS、および LDAP サーバーが更新されていることを確認します。	205 ページの「ネームサービスの IPv6 サポート用の構成」

IPv6 ネットワークトポロジのシナリオ

この章では、典型的な企業ネットワークで IPv6 サービスを計画する方法について説明します。次の図に、この章の説明で使用するネットワークを示します。実際に計画しようとしている IPv6 ネットワークには、この図で示されるネットワークリンクのすべてが含まれるとは限りません。

図 4–1 IPv6 ネットワークトポロジのシナリオ



この企業ネットワークシナリオでは、既存の IPv4 アドレスを持つサブネットが 5 つあります。ネットワークのリンクは管理サブネットに直接対応します。4 つの内部ネットワークは、RFC 1918 スタイルの IPv4 専用アドレスで表されています。このアドレスは、IPv4 アドレスの不足に対応するための一般的な解決方法です。このような内部ネットワークのアドレス指定スキームは次のとおりです。

- Subnet 1 は内部ネットワークバックボーン `192.168.1` です。
- Subnet 2 は内部ネットワーク `192.168.2` であり、LDAP、`sendmail`、および DNS サーバーが含まれます。
- Subnet 3 は内部ネットワーク `192.168.3` であり、企業の NFS サーバーが含まれます。
- Subnet 4 は内部ネットワーク `192.168.4` であり、企業の従業員用のホストが含まれます。

外部の公開ネットワーク `172.16.85` は、企業の DMZ として機能します。このネットワークには、Web サーバーや匿名 FTP サーバーなど、企業が外部に提供するリソースが含まれます。Router 2 はファイアウォールを実行して、公開ネットワーク `172.16.85` を内部バックボーンから分離します。DMZ のもう一方の終端では、Router 1 がファイアウォールを実行して、企業の境界サーバーとして機能します。

図 4–1 では、公開 DMZ は RFC 1918 専用アドレス `172.16.85` を持っています。実際には、公開 DMZ は登録済み IPv4 アドレスを持っている必要があります。ほとんどの IPv4 サイトは、公開アドレスと RFC 1918 専用アドレスの組み合わせを使用します。しかし、IPv6 を導入すると、公開アドレスと専用アドレスの概念が変わります。 IPv6 は巨大なアドレス空間を持つため、専用ネットワークにも、公開ネットワークにも、IPv6 公開アドレスを使用します。

IPv6 をサポートするための既存のネットワークの準備

注 – Oracle Solaris デュアルプロトコルスタックは、IPv4 と IPv6 の並行動作をサポートします。IPv6 をネットワークに配備している間も、そのあとでも、IPv4 関連の操作は正常に実行できます。

IPv6 は、既存のネットワークに新しい機能を追加します。したがって、IPv6 を初めて配備するときには、IPv4 関連の操作が中断されないことを確認する必要があります。この節では、IPv6 を着実に既存のネットワークに導入する方法について説明します。

IPv6 をサポートするためのネットワークトポロジの準備

IPv6 配備の最初の手順では、ネットワーク上の既存のエンティティーが IPv6 をサポートできるかどうかを判断することです。ほとんどの場合、ネットワークトポロジ、つまり、ワイヤー、ルーター、およびホストは、IPv6 の実装時に変更する必要はありません。ただし、既存のハードウェアおよびアプリケーションについては、実際に IPv6 アドレスをネットワークインタフェースに構成する前に、IPv6 をサポートするための準備を行う必要がある場合があります。

ネットワーク上のどのハードウェアを IPv6 向けにアップグレードできるかを確認します。たとえば、次のクラスのハードウェアについては、メーカーのマニュアルで IPv6 対応状況を調べてください。

- ルーター
- ファイアウォール
- サーバー
- スイッチ

注 – このパートで説明するすべての手順では、すべての装置 (特に、ルーター) が IPv6 向けにアップグレードできると仮定します。

IPv6 向けにアップグレードできないルーターモデルもあります。詳細と回避方法については、237 ページの「IPv4 ルーターを IPv6 用にアップグレードできない」を参照してください。

IPv6 をサポートするためのネットワークサービスの準備

現在の Oracle Solaris リリースにおいて、次の典型的な IPv4 ネットワークサービスは IPv6 をサポートできます。

- `sendmail`
- NFS
- HTTP (Apache 2.x または Orion)
- DNS
- LDAP

IMAP メールサービスは IPv4 専用です。

IPv6 向けに構成されたノードでも IPv4 サービスは実行できます。IPv6 を有効にしても、必ずしもすべてのサービスが IPv6 接続を受け入れるわけではありません。IPv6

向けに移植されたサービスだけが IPv6 接続を受け入れます。IPv6 向けに移植されていないサービスは、プロトコルスタックの IPv4 部分を使用して機能し続けることができます。

IPv6 向けにアップグレードしたあとで、いくつかの問題が発生する可能性があります。詳細については、238 ページの「サービスを IPv6 用にアップグレードしたあとの問題」を参照してください。

IPv6 をサポートするためのサーバーの準備

サーバーは IPv6 ホストであると考えられるため、デフォルトでは、IPv6 アドレスは自動的に近傍検索プロトコルによって構成されます。しかし、多くのサーバーは複数の NIC (Network Interface Card) を持っており、保守のために、そのいくつかを交換したい場合もあります。ある NIC を交換すると、近傍検索プロトコルは自動的に、その NIC に新しいインタフェース ID を生成します。サーバーによっては、この動作を受け入れることができない場合があります。

このような場合は、サーバーのインタフェースごとに、IPv6 アドレスのインタフェース ID 部分を手動で構成する方法を考えてください。手順については、192 ページの「ユーザー指定の IPv6 トークンを構成する方法」を参照してください。そのあとで既存の NIC を交換する必要がある場合、その NIC には以前構成した IPv6 アドレスが適用されます。

▼ IPv6 をサポートするためにネットワークサービスを準備する方法

1 **IPv6 をサポートするには、次のネットワークサービスを更新します。**

- メールサーバー
- NIS サーバー
- NFS

注 – LDAP は IPv6 をサポートします。IPv6 固有な構成タスクは必要ありません。

---

2 **ファイアウォールハードウェアが IPv6 をサポートできるかどうかを確認します。**

この手順については、ファイアウォール関連の適切なマニュアルを参照してください。

3 ネットワーク上のほかのサービスが **IPv6** 向けに移植されているかどうかを確認します。

詳細については、ソフトウェアに付属するマニュアルや関連するマニュアルを参照してください。

4 次のサービスを配備しているサイトでは、これらのサービスを適切に評価しているかどうかを確認します。

- ファイアウォール

  IPv6 をサポートするために、IPv4 向けに作成したポリシーを強化することを考えてください。セキュリティーの詳しい考慮事項については、94 ページの「IPv6 実装のセキュリティーについて」を参照してください。

- 郵便

  DNS の MX レコードにおいて、メールサーバーの IPv6 アドレスを追加することを考えてください。

- DNS

  DNS 固有の問題点については、93 ページの「IPv6 をサポートするために DNS を準備する方法」を参照してください。

- IPQoS

  ホストで IPv4 向けに使用していたのと同じ Diffserv ポリシーを使用します。詳細については、843 ページの「クラシファイアモジュール」を参照してください。

5 ノードを **IPv6** 向けに変更する前に、そのノードが提供するネットワークサービスを評価します。

▼ IPv6 をサポートするために DNS を準備する方法

現在の Oracle Solaris のリリースは、クライアント側とサーバー側の両方において、DNS による名前解決をサポートします。IPv6 をサポートするために DNS サービスを準備するには、次の手順を行います。

IPv6 のDNS サポートに関連する詳細については、『Solaris のシステム管理 (ネーミングとディレクトリサービス : DNS、NIS、LDAP 編)』を参照してください。

1 再帰的な名前解決を実行する **DNS** サーバーがデュアルスタックであるか (つまり、**IPv4** と **IPv6** 両用であるか)、あるいは、**IPv4** 専用であるかを判断します。

2 **DNS** サーバーでは、関連する **IPv6** データベース **AAAA** レコードを前進ゾーンで使用して、**DNS** データベースを作成します。

注 – 複数の基幹系のサービスを実行しているサーバーには、特に注意する必要があります。ネットワークが適切に機能していることを確認します。また、すべての基幹系のサービスが IPv6 向けに移植されていることを確認します。次に、そのサーバーの IPv6 アドレスを DNS データベースに追加します。

3 **AAAA** レコードの関連する **PTR** レコードを逆進ゾーンに追加します。

4 **IPv4** 専用データまたは **IPv6** と **IPv4** 両用データを、ゾーンを記述する **NS** レコードに追加します。

ネットワークトポロジにおけるトンネルの計画

IPv6 実装は、IPv4 と IPv6 が混在するネットワークへの移行メカニズムとして、多数のトンネル構成をサポートします。トンネルを使用すると、孤立した IPv6 ネットワークどうしが通信できるようになります。ほとんどのインターネットは IPv4 で動作しているため、自分のサイト (IPv6 ネットワーク) から宛先のサイト (IPv6 ネットワーク) に IPv6 パケットを送信するためには、インターネットにトンネルを開けて、そこを通す必要があります。

次に、IPv6 ネットワークトポロジにおいてトンネルを使用するいくつかのシナリオを示します。

- ISP から IPv6 サービスを購入すると、自分のサイトの境界ルーターから ISP ネットワークにトンネルを作成できます。図 4–1 に、このようなトンネルを示します。このようなシナリオの場合は、手動の IPv6 over IPv4 トンネルを実行します。
- 大規模な分散ネットワークを IPv4 接続で管理している場合。IPv6 を使用する分散サイトに接続するには、各サブネットの境界ルーターから自動 6to4 トンネルを実行します。
- 自分のインフラストラクチャー内のルーターを IPv6 向けにアップグレードできないこともあります。このような場合には、2 つの IPv6 ルーターをエンドポイントとして、IPv4 ルーターに手動トンネルを作成できます。

トンネルを構成する手順については、195 ページの「IPv6 サポート用にトンネルを構成するためのタスク (タスクマップ)」を参照してください。トンネルに関する概念の情報については、295 ページの「IPv6 トンネル」を参照してください。

IPv6 実装のセキュリティーについて

IPv6 を既存のネットワークに導入するとき、サイトのセキュリティーを損なわないように注意する必要があります。IPv6 を導入するときには、次のセキュリティーの問題点に注意してください。

- IPv6 パケットと IPv4 パケットには、両方とも、同じ量のフィルタリンクが必要です。
- IPv6 パケットは頻繁にファイアウォールにトンネルを開けます。したがって、次のシナリオのどちらかを実装する必要があります。
  - ファイアウォールでトンネル内部のコンテンツを検査すること。
  - トンネルの反対側にあるエンドポイントでも、同じような規則の IPv6 ファイアウォールを設置すること。
- IPv6 over UDP over IPv4 トンネルを使用するような移行メカニズムもあります。しかし、このようなメカニズムはファイアウォールを通らないため、危険であることが証明されています。
- IPv6 ノードは企業ネットワークの外からグローバルに到達できます。セキュリティーポリシーで公開アクセスを禁止する場合、ファイアウォールに対して、より厳しい規則を確立する必要があります。たとえば、ステートフルなファイアウォールを考えてください。

このドキュメントでは、IPv6 実装で使用できるセキュリティーについても説明しています。

- IP セキュリティーアーキテクチャー (IPsec) 機能を使用すると、IPv6 パケットを暗号化で保護できます。詳細については、第 19 章「IP セキュリティーアーキテクチャー (概要)」を参照してください。
- Internet Key Exchange (IKE) 機能を使用すると、IPv6 パケットに公開鍵認証を使用できます。詳細については、第 22 章「インターネット鍵交換 (概要)」を参照してください。

IPv6 アドレス指定計画の準備

IPv4 から IPv6 への移行の大部分は、アドレス指定計画の立案です。このタスクには、次の前準備が必要です。

- 95 ページの「サイト接頭辞の取得」
- 96 ページの「IPv6 番号付けスキームの作成」

サイト接頭辞の取得

IPv6 を構成する前に、サイト接頭辞を取得する必要があります。サイト接頭辞は、自分の IPv6 実装におけるすべてのノードの IPv6 アドレスを抽出するときに使用します。サイト接頭辞の概要については、78 ページの「IPv6 の接頭辞」を参照してください。

IPv6 をサポートする ISP は、48 ビットの IPv6 サイト接頭辞を提供できます。現在の ISP が IPv4 しかサポートしない場合、現在の ISP を IPv4 サポート用に残したまま、別

の ISP を IPv6 サポート用に使用できます。このような場合の回避方法は複数あります。詳細については、238 ページの「現在の ISP が IPv6 をサポートしない」を参照してください。

企業自身が ISP である場合、顧客のサイト接頭辞は適切なインターネットレジストリから取得します。詳細については、Internet Assigned Numbers Authority (IANA) (`http://www.iana.org`) を参照してください。

IPv6 番号付けスキームの作成

IPv6 ネットワークがまったく新しいものでない限り、既存の IPv4 トポロジを IPv6 番号付けスキームとして使用します。

サブネット用の番号付けスキームの作成

番号付けスキームを開始するには、まず、既存の IPv4 サブネットを等価な IPv6 サブネットにマッピングします。たとえば、図 4–1 で示したサブネットを考えてください。サブネット 1 からサブネット 4 までは、RFC 1918 の IPv4 専用アドレス指定を使用して、アドレスの最初の 16 ビットを指定し、さらに、1 から 4 までの数字を使用して、サブネットを指定しています。この例では、IPv6 接頭辞 `2001:db8:3c4d/48` がサイトに割り当てられていると仮定します。

次の表に、専用アドレスの IPv4 接頭辞から IPv6 接頭辞にマッピングする方法を示します。

IPv4 サブネット接頭辞	等価な IPv6 サブネット接頭辞
`192.168.1.0/24`	`2001:db8:3c4d:1::/64`
`192.168.2.0/24`	`2001:db8:3c4d:2::/64`
`192.168.3.0/24`	`2001:db8:3c4d:3::/64`
`192.168.4.0/24`	`2001:db8:3c4d:4::/64`

ノードの IPv6 アドレス指定計画の立案

ほとんどのホストにおいて、インタフェースに IPv6 アドレスを構成するのに適切で時間がかからない戦略は、ステートレス自動構成です。ホストが最も近いルーターからサイト接頭辞を受信したとき、近傍検索プロトコルは自動的に、ホストの各インタフェースに IPv6 アドレスを生成します。

サーバーは安定した IPv6 アドレスを持つ必要があります。サーバーの IPv6 アドレスを手動で構成しない場合、サーバーの NIC カードを交換したときには、新しい IPv6 アドレスが自動構成されます。サーバーのアドレスを作成するときには、次のことを覚えておいてください。

- サーバーには意味のある安定したインタフェース ID を指定してください。インタフェース ID の番号付けスキームを使用するときには、1 つの戦略だけを使用します。たとえば、図 4–1 の LDAP サーバーの内部インタフェースは `2001:db8:3c4d:2::2` になります。
- あるいは、IPv4 ネットワークの番号を定期的に変更しない場合、ルーターおよびサーバーの既存の IPv4 アドレスをそのインタフェース ID として使用することを考えてください。図 4–1 では、Router 1 の DMZ へのインタフェースは IPv4 アドレス `123.456.789.111` を持っていると仮定します。この IPv4 アドレスを 16 進数に変換すると、その結果をインタフェース ID として使用できます。つまり、新しいインタフェース ID は `::7bc8:156F` になります。

  この方法は、ISP から IPv4 アドレスを取得したのではなく、登録済み IPv4 アドレスを所有しているときだけに使用するようにしてください。ISP から取得した IPv4 アドレスを使用している場合、依存関係が発生し、ISP を変更する場合に問題が発生します。

IPv4 アドレスの数には制限があるため、ネットワーク設計者は、既に登録済みのグローバルアドレスや RFC 1918 専用アドレスをどのように使用するかを考える必要がありました。しかし、IPv4 のグローバルアドレスや専用アドレスの表記は IPv6 アドレスには適用されません。サイト接頭辞を含むグローバルユニキャストは、ネットワークのすべてのリンクで使用できます (公開 DMZ を含む)。

◆ ◆ ◆ 第 5 章

TCP/IP ネットワークサービスと IPv4 アドレス指定の構成 (作業)

TCP/IP ネットワークの管理は、2 つの段階で行います。最初の段階ではハードウェアを組み立てます。次に、TCP/IP プロトコルを実装するデーモンや、ファイル、サービスを構成します。

この章では、IPv4 アドレス指定とサービスを実装する TCP/IP をネットワークで構成する方法について説明します。

注 – この章のタスクの多くは、IPv4 のみをサポートするネットワークにも、IPv6 が有効なネットワークにも適用されます。構成タスクが 2 つのアドレス指定形式の間で異なる場合、この章には IPv4 の構成手順が記載されています。この章のタスクでは、同等の IPv6 作業 (第 7 章「IPv6 ネットワークの構成 (手順)」) を相互参照します。

この章では、次の内容について説明します。

この章で説明する新機能

Solaris 10 8/07 には、次の変更が加えられています。

- `routeadm` コマンドを使用する代わりに、サービス管理機能 (SMF) を介してルーティングを構成および管理できます。手順については、117 ページの「IPv4 ネットワーク上でのパケット転送と経路制御」の例および `routeadm(1M)` のマニュアルページを参照してください。
- `/etc/inet/ipnodes` ファイルは廃止されました。個々の手順で説明されているとおり、`/etc/inet/ipnodes` は以前の Solaris 10 リリースにのみ使用してください。

IPv4 ネットワークを構成する前に (作業マップ)

TCP/IP を構成する前に、次の表に示す作業を完了してください。表では、各タスクで実行する内容の説明と、タスクの具体的な実行手順が詳しく説明されている現在のマニュアル内のセクションを示しています。

タスク	説明	参照先
1. ネットワーク設計者の場合は、ネットワークトポロジを設計します。	ネットワークの物理レイアウトを決めます。	67 ページの「ネットワークトポロジの概要」および 121 ページの「IPv4 自律システムのトポロジ」
2. ISP または Regional Internet Registry (RIR) からネットワーク番号を入手します。	このサイトのシステムが外部と通信できるようにするために、登録されているネットワーク番号を入手します。	60 ページの「IPv4 アドレス指定スキームの設計」
3. ネットワークに対する IPv4 アドレス指定スキームの計画を立てます。必要に応じて、サブネットアドレス指定の計画も含めます。	ネットワーク番号をアドレス指定計画のベースとして使用します。	60 ページの「IPv4 アドレス指定スキームの設計」
4. ネットワークトポロジに従ってネットワークハードウェアを組み立てます。ハードウェアが正しく動作することを確認します。	ネットワークトポロジの設計に従ってシステムやネットワークメディア、ルーター、スイッチ、ハブ、ブリッジをセットアップします。	ハードウェアマニュアルと 67 ページの「ネットワークトポロジの概要」
5. ネットワークのすべてのシステムに IPv4 アドレスとホスト名を割り当てます。	Oracle Solaris のインストール時かインストール後に IPv4 アドレスを適切なファイルで指定します。	60 ページの「IPv4 アドレス指定スキームの設計」および 112 ページの「IPv4 アドレスおよびその他のネットワーク構成パラメータを変更する方法」

タスク	説明	参照先
6. ネットワークインタフェースとルーターが必要とする構成ソフトウェアがあれば、それを実行します。	ルーターとマルチホームホストを構成します。	ルーターについては、67 ページの「ネットワーク上でのルーターの計画」と 124 ページの「IPv4 ルーターの構成」を参照してください。
7. ネットワークでどのネームサービスまたはディレクトリサービスを使用するのかを指定します。(NIS、LDAP、DNS、またはローカルファイル)。	選択したネームサービスまたはディレクトリサービス (またはその両方) を構成します。	『Solaris のシステム管理 (ネーミングとディレクトリサービス : DNS、NIS、LDAP 編)』
8. 必要なら、ネットワークで使用するドメイン名を選択します。	ネットワークのドメイン名を選択し、InterNIC に登録します。	『Solaris のシステム管理 (ネーミングとディレクトリサービス : DNS、NIS、LDAP 編)』

ホスト構成モードの決定

ネットワーク管理者として、ホストやルーター (該当する場合) で TCP/IP が動作するように構成します。これらのシステムは、ローカルシステム上のファイルまたはネットワーク上のほかのシステムにあるファイルから構成情報を入手するように構成できます。

必要な構成情報を次に示します。

- 各システムのホスト名
- 各システムの IP アドレス
- 各システムが属しているドメインの名前
- デフォルトルーター
- 各システムのネットワークで使用されている IPv4 ネットマスク

TCP/IP 構成情報をローカルファイルから入手するシステムは「ローカルファイルモード」で動作します。TCP/IP 構成情報をリモートネットワークサーバーから入手するシステムは「ネットワーククライアントモード」で動作します。

ローカルファイルモードで実行するシステム

システムがローカルファイルモードで動作するには、TCP/IP 構成ファイルのローカルコピーを持っている必要があります。これらの構成ファイルについては、241 ページの「TCP/IP 構成ファイル」を参照してください。このシステムが専用のディスクを持っていることが望ましいですが、不可欠というわけではありません。

ほとんどのサーバーはローカルファイルモードで実行します。このようなサーバーの一部を次に示します。

- ネットワーク構成サーバー
- NFS サーバー
- NIS、LDAP、または DNS のサービスを提供するネームサーバー
- メールサーバー

また、ルーターはローカルファイルモードで実行する必要があります。

システムがプリントサーバー専用の場合は、ローカルファイルモードで実行する必要はありません。個々のホストをローカルファイルモードで実行する方がよいかどうかは、ネットワークの規模によって異なります。

ネットワークがきわめて小さい場合は、個々のホストのファイルを管理する作業は比較的簡単です。しかし、数百のホストから成るネットワークの場合は、そのネットワークがいくつかの管理サブドメインに分割されていたとしても、この作業は困難なものとなります。したがって、規模の大きいネットワークの場合は、ローカルファイルモードを使用しても一般に効率は上がりません。ただし、ルーターとサーバーはそれぞれ自身で構成されるものなので、ローカルファイルモードで構成する必要があります。

ネットワーク構成サーバー

ネットワーク構成サーバーとは、ネットワーククライアントモードで構成されているホストに TCP/IP 構成情報を提供するサーバーのことです。この種のサーバーは、次の 3 つのブートプロトコルをサポートしています。

- RARP – Reverse Address Resolution Protocol (RARP) では、Ethernet アドレス (48 ビット) を IPv4 アドレス (32 ビット) にマップします。この操作は ARP の逆です。ネットワーク構成サーバーで RARP を実行すると、ネットワーククライアントモードで動作しているホストは、各自の IP アドレスと TCP/IP 構成ファイルをそのネットワーク構成サーバーから入手します。RARP サービスは、`in.rarpd` デーモンを使用して使用可能にできます。詳細は、`in.rarpd(1M)` のマニュアルページを参照してください。
- TFTP – Trivial File Transfer Protocol (TFTP) は、リモートシステム間でファイルを転送するアプリケーションです。`in.tftpd` デーモンが TFTP サービスを実施し、その結果、ネットワーク構成サーバーとそれぞれのネットワーククライアントとの間のファイル転送が可能になります。詳細は、`in.tftpd(1M)` のマニュアルページを参照してください。
- Bootparams – Bootparams プロトコルでは、ネットワークからブートされるクライアントに不可欠なブート用パラメータを提供します。このサービスを実行するのは `rpc.bootparamd` デーモンです。詳細は、`bootparamd(1M)` のマニュアルページを参照してください。

ネットワーク構成サーバーは、NFS ファイルサーバーとしても使用できます。

いずれかのホストをネットワーククライアントとして構成する場合は、さらに、ネットワーク上の少なくとも 1 つのシステムをネットワーク構成サーバーとして構成する必要があります。ネットワークをサブネット化する場合は、ネットワーククライアントを持つ各サブネットについて、ネットワーク構成サーバーが少なくとも 1 つは必要です。

ネットワーククライアントとしてのシステム

構成情報をネットワーク構成サーバーから入手するホストは、いずれもネットワーククライアントモードで動作します。ネットワーククライアントとして構成されているシステムには、TCP/IP 構成ファイルのローカルコピーは必要ありません。

「ネットワーククライアントモード」を使用すると、大規模ネットワークの管理が大幅に簡素化されます。さらに、個々のホストで行う構成作業が最小限の量で済み、ネットワーク上のすべてのシステムが同じ構成標準に従っていることが保証されます。

ネットワーククライアントモードは、あらゆるタイプのコンピュータに構成できます。たとえば、スタンドアロンシステムでネットワーククライアントモードを構成できます。

混合構成

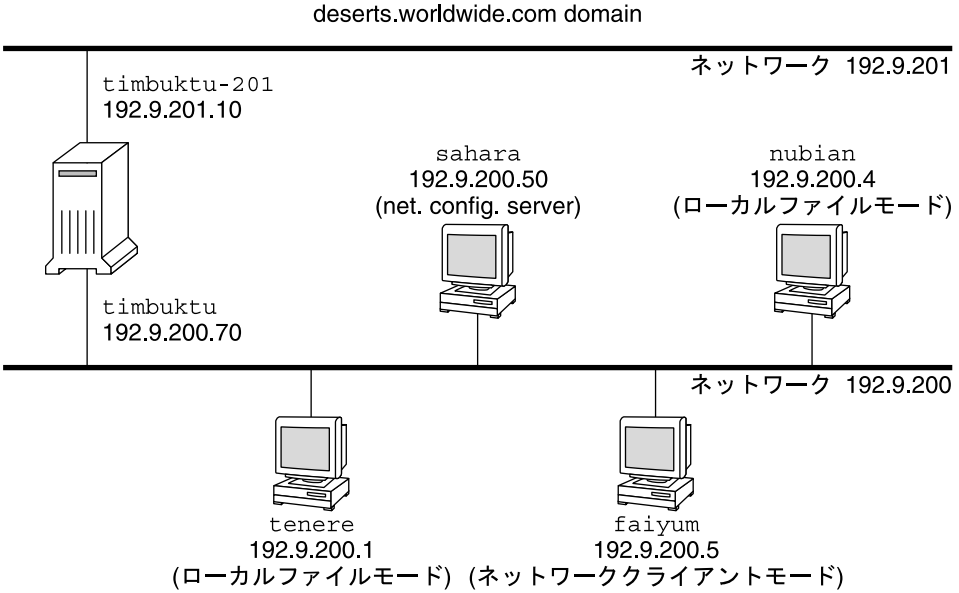
構成をローカルファイルモードだけに限ったり、ネットワーククライアントモードだけに限ったりする必要はありません。ルーターとサーバーは常にローカルモードで構成する必要があります。ホストについては、ローカルファイルモードとネットワーククライアントモードを任意に組み合わせて使用できます。

IPv4 ネットワークトポロジのシナリオ

図 5–1 は、ネットワーク番号が `192.9.200` である架空のネットワークで動作するホストを示しています。このネットワークには、`sahara` というネットワーク構成サーバーが 1 つあります。`tenere` と `nubian` の 2 つのホストはそれぞれ独自にディスクを持っており、ローカルファイルモードで動作します。ホスト `faiyum` もディスクを持っていますが、このシステムはネットワーククライアントモードで動作します。

最後に、システム `timbuktu` はルーターとして構成されています。このシステムには 2 つのネットワークインタフェースが組み込まれています。最初のインタフェースは `timbuktu` という名前です。このインタフェースはネットワーク `192.9.200` に属しています。残りの 1 つは `timbuktu-201` という名前です。このインタフェースはネットワーク `192.9.201` に属しています。どちらのネットワークも、組織ドメイン `deserts.worldwide.com` に含まれています。このドメインは、ローカルファイルをネームサービスとして使用します。

図 5–1 IPv4 ネットワークトポロジに属するホストのシナリオ



ネットワークにサブネットを追加する (作業マップ)

サブネットを使用しないネットワークを、サブネットを使用するネットワークに変更する場合は、次の作業マップに示す作業を行います。

注 – この節の情報は IPv4 サブネットだけに適用されます。IPv6 サブネットの計画については、91 ページの「IPv6 をサポートするためのネットワークトポロジの準備」と96 ページの「サブネット用の番号付けスキームの作成」を参照してください。

次の表は、サブネットを現在のネットワークに追加するための各種作業の一覧です。表では、各タスクで実行する内容の説明と、タスクの具体的な実行手順が詳しく説明されている現在のマニュアル内のセクションを示しています。

タスク	説明	参照先
1. そのネットワークトポロジでサブネットが必要かどうかを決めます。	ルーターやホストをそれぞれのサブネットのどこに置くかなど、新しいサブネットトポロジを決めます。	67 ページの「ネットワーク上でのルーターの計画」、248 ページの「サブネット化とは」、および 262 ページの「ネットワーククラス」
2. 新しいサブネット番号を持つ一連の IP アドレスをシステムに割り当ててサブネットのメンバーになります。	Oracle Solaris のインストール時かインストール後に、新しいサブネット番号を使用する IP アドレスを /etc/hostname.*interface* ファイルに構成します。	55 ページの「ネットワークの IP アドレス指定形式の決定」
3. サブネットで使用する可能性があるすべてのシステムに対してサブネットのネットワークマスクを構成します。	ネットワーククライアントを手動で構成する場合は、/etc/inet/netmasks ファイルを変更します。あるいは、Oracle Solaris インストールプログラムにネットマスクを渡します。	248 ページの「netmasks データベース」と249 ページの「IPv4 アドレス用のネットワークマスクの作成」
4. サブネットに属するすべてのシステムの新しい IP アドレスをデータベースに追加します。	新しいホストアドレスを反映させるために、すべてのホスト上で /etc/inet/hosts ファイルおよび /etc/inet/ipnodes ファイル (Solaris 10 11/06 以前のリリースの場合) を変更します。	244 ページの「hosts データベース」
5. すべてのシステムをリブートします。		

ネットワークを構成する (作業マップ)

次の表は、サブネットなしのネットワーク構成からサブネットを使用するネットワークに変更したあとに実行する追加タスクの一覧です。表では、各タスクで実行する内容の説明と、タスクの具体的な実行手順が詳しく説明されている現在のマニュアル内のセクションを示しています。

タスク	説明	参照先
ホストをローカルファイルモード用に構成します	nodename、hostname、hosts、defaultdomain、defaultrouter、および netmasks ファイルを編集します。	107 ページの「ローカルファイルモードの場合のホストの構成方法」

タスク	説明	参照先
ネットワーク構成サーバーをセットアップします	in.tftp デーモンをオンに設定し、 hosts、 ethers、および bootparams ファイルを編集します	110 ページの「ネットワーク構成サーバーの設定方法」
ホストをネットワーククライアントモード用に構成します	hostname ファイルを作成し、hosts ファイルを編集します。また、nodename ファイルと defaultdomain ファイルがある場合はこれらを削除します	111 ページの「ネットワーククライアントモードの場合のホストの構成方法」
ネットワーククライアントの経路制御戦略を指定します	ホストで静的経路制御を使用するか動的経路制御を使用するかを決めます。	137 ページの「単一インタフェースホストで静的ルーティングを有効にする方法」および 139 ページの「単一インタフェースホストで動的経路制御を有効にする方法」
既存のネットワーク構成を変更します	インストール時に設定された、またはそれ以降に構成されたホスト名、IP アドレス、およびその他のパラメータを変更します。	112 ページの「IPv4 アドレスおよびその他のネットワーク構成パラメータを変更する方法」

ローカルネットワーク上でのシステム構成

オペレーティングシステムのソフトウェアをインストールするときに、同時にネットワークのソフトウェアもインストールされます。そのときに、いくつかの IP 構成パラメータを対応するファイルに格納して、ブート時に読み取れるようにしておく必要があります。

ネットワークの構成処理では、ネットワーク構成ファイルを作成または編集する必要があります。システムのカーネルが構成情報をどのように入手するかは、設定によって異なります。これらのファイルがローカルに格納されているか (ローカルファイルモード) ネットワーク構成サーバーから入手するか (ネットワーククライアントモード) によって提供方法が変わります。

ネットワーク構成時に提供されるパラメータは次のとおりです。

- すべてのシステムの各ネットワークインタフェースの IP アドレス。
- ネットワークにある各システムのホスト名。ホスト名は、ローカルファイルまたはネームサービスデータベースに入力できます。
- システムが設置されている、NIS、LDAP、または DNS のドメイン名 (該当する場合)。

- デフォルトのルーターアドレス。この情報は、各ネットワークにルーターが 1 つしか接続していないような単純なネットワークトポロジの場合、またはルーターが RDISC (Router Discovery Protocol) や RIP (Routing Information Protocol) などのルーティングプロトコルを実行しない場合に指定します。デフォルトルーターの詳細は、117 ページの「IPv4 ネットワーク上でのパケット転送と経路制御」を参照してください。Oracle Solaris でサポートされているルーティングプロトコルのリストについては、表 5–1 を参照してください。
- サブネットマスク (サブネットを持つネットワークの場合に限り必要)。

Oracle Solaris インストールプログラムにより、複数のインタフェースがシステム上で検出された場合は、追加のインタフェースをインストール時に構成することもできます。詳しい説明は、『Oracle Solaris 10 1/13 インストールガイド: 基本インストール』を参照してください。

ここでは、ローカル構成ファイルを作成および編集する手順を説明しています。ネームサービスデータベースを使用した動作については、『Solaris のシステム管理 (ネーミングとディレクトリサービス : DNS、NIS、LDAP 編)』を参照してください。

▼ ローカルファイルモードの場合のホストの構成方法

ローカルファイルモードで動作するホスト上の TCP/IP を構成するための手順は、次のとおりです。

Solaris 10 11/06 とそれ以降のリリースのインタフェースを手動で構成する手順については、153 ページの「システムインストール後に物理インタフェースを構成する方法」を参照してください。

1. **Primary Administrator の役割を引き受けるか、スーパーユーザーになります。**

   Primary Administrator 役割には、Primary Administrator プロファイルが含まれます。役割を作成してユーザーに役割を割り当てるには、『Oracle Solaris の管理: 基本管理』の第 2 章「Solaris 管理コンソールの操作 (タスク)」を参照してください。

2. **`/etc` ディレクトリに移動します。**

3. **`/etc/nodename` ファイルに正しいホスト名が設定されていることを確認します。**

   Oracle Solaris のインストール時にシステムのホスト名を指定したときは、そのホスト名は `/etc/nodename` ファイルに入ります。そのノード名エントリがシステムの正しいホスト名であることを確認します。

4 **/etc/hostname.***interface* ファイルがシステムのネットワークインタフェースごとに存在することを確認します。

/etc/hostname.*interface* ファイルのファイル構文および基本情報については、149 ページの「物理インタフェースの管理の基礎」を参照してください。

Oracle Solaris インストールプログラムでは、インストール時に少なくとも 1 つのインタフェースを構成する必要があります。最初に構成するインタフェースが、自動的にプライマリネットワークインタフェースになります。インストールプログラムは、プライマリネットワークインタフェースと、インストール時にオプションで構成するその他のインタフェースの /etc/hostname. *interface* ファイルを作成します。

インストール時に追加のインタフェースを構成した場合は、各インタフェースに対応する /etc/hostname. *interface* ファイルがあることを確認してください。Oracle Solaris のインストール時に複数のインタフェースを構成する必要はありません。ただし、あとでインタフェースをシステムに追加する場合は、それらを手動で構成する必要があります。

Solaris 10 11/06 とそれ以降のリリースのインタフェースを手動で構成する手順については、153 ページの「システムインストール後に物理インタフェースを構成する方法」を参照してください。

5 **Solaris 10 11/06** 以前のリリースの場合は、**/etc/inet/ipnodes** ファイルのエントリが最新であることを確認します。

Solaris 10 インストールプログラムは、/etc/inet/ipnodes ファイルを作成します。このファイルには、インストール時に構成されたすべてのインタフェースのノード名と IPv4 アドレス、および IPv6 アドレス (使用される場合) が含まれます。

/etc/inet/ipnodes ファイルのエントリには、次の書式を使用します。

IP-address node-name nicknames...

nicknames は、インタフェースを識別するための追加の名前です。

6 **/etc/inet/hosts** ファイルのエントリが最新であることを確認します。

Oracle Solaris インストールプログラムは、プライマリネットワークインタフェース、ループバックアドレス、およびインストール時に構成された追加インタフェース (該当する場合) に対する各エントリを作成します。

a. **/etc/inet/hosts** ファイルにすでに存在するエントリが最新であることを確認します。

b. (オプション) インストール後にローカルホストに追加されたネットワークインタフェースの **IP** アドレスとそれに対応する名前を追加します。

c. (オプション) **/usr** ファイルシステムを **NFS** マウントする場合は、ファイルサーバーの **IP** アドレス (**1** つまたは複数) を追加します。

7 ホストの完全修飾ドメイン名を **/etc/defaultdomain** ファイルに入力します。

たとえば、ホスト tenere がドメイン deserts.worldwide.com に所属していたとします。その場合は、/etc/defaultdomain に deserts.worldwide.com を入力します。詳細は、243 ページの「/etc/defaultdomain ファイル」を参照してください。

8 ルーターの名前を **/etc/defaultrouter** ファイルに入力します。

このファイルについては、243 ページの「/etc/defaultrouter ファイル」を参照してください。

9 デフォルトのルーターの名前とその IP アドレスを **/etc/inet/hosts** ファイルに入力します。

追加の経路制御オプションについては、111 ページの「ネットワーククライアントモードの場合のホストの構成方法」を参照してください。これらのオプションは、ローカルファイルモード構成にも適用できます。

10 該当する場合は、ネットワークのネットワークマスクを追加します。

- ホストがその IP アドレスを DHCP サーバーから入手する場合は、ネットワークマスクを指定する必要はありません。
- このクライアントと同じネットワークに NIS サーバーをすでに設定している場合は、そのサーバーの適切なデータベースに netmask 情報を追加できます。
- それ以外の場合は、次のことを行います。

a. ネットワーク番号とネットマスクを **/etc/inet/netmasks** ファイルに入力します。

次の書式で入力します。

```
network-number netmask
```

たとえば、Class C ネットワーク番号 192.168.83 の場合は、次のように入力します。

```
192.168.83.0    255.255.255.0
```

CIDR アドレスの場合は、ネットワークの接頭辞をそれと同等の 10 進ドット表記に変換します。ネットワーク接頭辞とその 10 進ドット表記については、表 2–3 を参照してください。たとえば、192.168.3.0/22 という CIDR ネットワーク接頭辞を表現するには、次のような表記を使用します。

```
192.168.3.0 255.255.252.0
```

b. ローカルファイルが最初に検索されるように、**/etc/nsswitch.conf** でのネットマスクの検索順序を変更します。

```
netmasks:   files nis
```

11 システムをリブートします。

▼ ネットワーク構成サーバーの設定方法

インストールサーバーやブートサーバーの設定方法については、『Oracle Solaris 10 1/13 インストールガイド: 基本インストール』を参照してください。

1. **Primary Administrator** 役割を引き受けるか、スーパーユーザーになります。

   Primary Administrator 役割には、Primary Administrator プロファイルが含まれます。役割を作成してユーザーに役割を割り当てるには、『Oracle Solaris の管理: 基本管理』の第 2 章「Solaris 管理コンソールの操作 (タスク)」を参照してください。

2. ネットワーク構成サーバーとなるサーバーの **root (/)** ディレクトリに移ります。

3. **/tftpboot** ディレクトリを作成することによって、**in.tftpd** デーモンをオンに設定します。

   ```
   # mkdir /tftpboot
   ```

   このコマンドにより、システムは、TFTP、bootparams、RARP のサーバーに構成されます。

4. 手順 **2** で作成したディレクトリに対するシンボリックリンクを作成します。

   ```
   # ln -s /tftpboot/. /tftpboot/tftpboot
   ```

5. **/etc/inetd.conf** ファイルの **tftp** 行を有効にします。

   エントリが次のようになっているか確認します。

   ```
   tftp dgram udp6 wait root /usr/sbin/in.tftpd in.tftpd -s /tftpboot
   ```

   これによって、`in.tftpd` は、`/tftpboot` にあるファイルだけから読み取られます。

6. **hosts** データベースを編集します。

   ネットワーク上のすべてのクライアントのホスト名と IP アドレスを追加します。

7. **ethers** データベースを編集します。

   ネットワーククライアントモードで実行するネットワーク上のすべてのホストについてエントリを作成します。

8. **bootparams** データベースを編集します。

   詳細は、257 ページの「bootparams データベース」を参照してください。ワイルドカードエントリを作成するか、またはネットワーククライアントモードで実行するすべてのホストについてエントリを作成します。

9. **/etc/inetd.conf** エントリをサービス管理機能 **(SMF)** のサービスマニフェストに変換し、それによって得られるサービスを使用可能にします。

   ```
   # /usr/sbin/inetconv
   ```

10 **in.tftpd** が正しく動作しているか確認します。

```
# svcs network/tftp/udp6
```

次のような出力が表示されます。

```
STATE          STIME    FMRI
online         18:22:21 svc:/network/tftp/udp6:default
```

参考 in.tftpdデーモンの管理

in.tftpd デーモンはサービス管理機能によって管理されます。in.tftpd に対する管理アクション (有効化、無効化、再起動など) を実行するには、svcadm コマンドを使用します。このサービスを起動したり、再起動したりする責任は inetd に委譲されています。in.tftpd の構成を変更したり、構成情報を表示したりするには、inetadm コマンドを使用します。このサービスのステータスを照会するには、svcs コマンドを使用します。サービス管理機能の概要については、『Oracle Solaris の管理: 基本管理』の第 18 章「サービスの管理 (概要)」を参照してください。

ネットワーククライアントの構成

ネットワーククライアントは、その構成情報をネットワーク構成サーバーから受け取ります。したがって、あるホストをネットワーククライアントとして構成するときは、このネットワーク用として、ネットワーク構成サーバーが少なくとも 1 つは設定されていることを確認してください。

▼ ネットワーククライアントモードの場合のホストの構成方法

ネットワーククライアントモードで構成する必要のある各ホストについて、次の手順を行います。

1 **Primary Administrator** 役割を引き受けるか、スーパーユーザーになります。

Primary Administrator 役割には、Primary Administrator プロファイルが含まれます。役割を作成してユーザーに役割を割り当てるには、『Oracle Solaris の管理: 基本管理』の第 2 章「Solaris 管理コンソールの操作 (タスク)」を参照してください。

2 **/etc** ディレクトリで **nodename** ファイルを検索します。

ある場合は、このファイルを削除してください。

/etc/nodename を削除すると、システムは hostconfig プログラムを使用して、ネットワーク構成サーバーから、ホスト名、ドメイン名、ルーターアドレスを入手するようになります。106 ページの「ローカルネットワーク上でのシステム構成」を参照してください。

3 **/etc/hostname.** *interface* ファイルが存在しない場合は、このファイルを作成します。

そのファイルが空であることを確認してください。/etc/hostname. *interface* ファイルが空であれば、システムは IPv4 アドレスをネットワーク構成サーバーから入手します。

4 **/etc/inet/hosts** ファイルにループバックネットワークインタフェースの **localhost** 名と **IP** アドレスだけが含まれていることを確認します。

```
# cat /etc/inet/hosts
# Internet host table
#
127.0.0.1       localhost
```

この IPv4 ループバックインタフェースの IP アドレスは 127.0.0.1 です。

詳細は、245 ページの「ループバックアドレス」を参照してください。このファイルには、ローカルホスト (プライマリネットワークインタフェース) の IP アドレスとホスト名が含まれていてはいけません。

5 **/etc/defaultdomain** ファイルが存在するかどうかを確認します。

ある場合は、このファイルを削除してください。

hostconfig プログラムはドメイン名を自動的に設定します。hostconfig プログラムが設定したドメイン名を無効にするには、代わりのドメイン名を /etc/defaultdomain ファイルに入力します。

6 クライアントの **/etc/nsswitch.conf** ファイルに指定されている検索パスがネットワークのネームサービス要件を満たしていることを確認します。

▼ IPv4 アドレスおよびその他のネットワーク構成パラメータを変更する方法

この手順では、すでにインストールされているシステムの IPv4 アドレス、ホスト名、およびその他のネットワークパラメータを変更する方法について説明します。サーバーまたはネットワーク接続されたスタンドアロンシステムの IP アドレスを変更する場合は、この手順を使用します。この手順は、ネットワーククライアントやネットワーク機器には適用されません。この手順で作成する構成は、リブート後も保持されます。

注 – ここで説明する手順は、プライマリネットワークインタフェースの IPv4 アドレスを変更する場合にのみ適用されます。別のインタフェースをシステムに追加する場合は、153 ページの「システムインストール後に物理インタフェースを構成する方法」を参照してください。

次の手順では、IPv4 アドレスとサブネットマスクを指定するときに、ほとんどの場合は IPv4 で一般的な 10 進ドット表記を使用しています。この手順で使用されるすべてのファイルでは、CIDR 表記を使用して IPv4 アドレスを指定することもできます。CIDR 表記の概要については、56 ページの「CIDR 書式の IPv4 アドレス」を参照してください。

1 **Primary Administrator 役割を引き受けるか、スーパーユーザーになります。**

Primary Administrator 役割には、Primary Administrator プロファイルが含まれます。役割を作成してユーザーに役割を割り当てるには、『Oracle Solaris の管理: 基本管理』の第 2 章「Solaris 管理コンソールの操作 (タスク)」を参照してください。

2 **Solaris 10 11/06 以前のリリースの場合のみ、`/etc/inet/ipnodes` ファイルまたは同等の `ipnodes` データベースで、IP アドレスを変更します。**

システムに追加するすべての IP アドレスに、次の構文を使用します。

IP-address host-name, nicknames
IP-address interface-name, nicknames

最初のエントリには、プライマリネットワークインタフェースの IP アドレスとシステムのホスト名を含めるようにしてください。ホスト名のニックネームを追加することもできます。物理インタフェースをシステムに追加するときは、`/etc/inet/ipnodes` にそれらのインタフェースの IP アドレスとそれらに関連付ける名前のエントリを 作成してください。

3 **システムのホスト名を変更する必要がある場合は、`/etc/nodename` ファイルのホスト名エントリを変更します。**

4 **`/etc/inet/hosts` ファイルまたは同等の `hosts` データベースで、IP アドレスとホスト名を必要に応じて変更します。**

5 **プライマリネットワークインタフェースの `/etc/hostname.`*interface* ファイルで、IP アドレスを変更します。**

`/etc/hostname.`*interface* ファイルのプライマリネットワークインタフェースエントリには、次のいずれかを使用できます。

- 一般的な 10 進ドット形式の IPv4 アドレス

  構文は次のとおりです。

  IPv4 address subnet mask

  ネットマスクエントリはオプションです。指定しなかった場合は、デフォルトのネットマスクが指定されていると見なされます。

  次に例を示します。

  ```
  # vi hostname.eri0
  10.0.2.5 netmask 255.0.0.0
  ```

- CIDR 表記の IPv4 アドレス (ネットワーク構成で使用されている場合)。

IPv4 address/network prefix

次に例を示します。

```
# vi hostname.eri0
10.0.2.5/8
```

CIDR 接頭辞には、その IPv4 アドレスに対応するネットマスクを指定します。たとえば、前述の `/8` は `255.0.0.0` というネットマスクになります。

- ホスト名。

  `/etc/hostname.`*interface* ファイルでシステムのホスト名を使用する場合は、そのホスト名とそれに関連付けられている IPv4 アドレスが `hosts` データベースにも指定されていることを確認します。

6 サブネットマスクが変更されている場合は、次のファイルにあるサブネットエントリを変更します。

- `/etc/netmasks`
- (オプション) `/etc/hostname.`*interface*

7 サブネットアドレスが変更されている場合は、**`/etc/defaultrouter`** ファイルに指定されているデフォルトルーターの **IP** アドレスを新しいサブネットのデフォルトルーターの **IP** アドレスに変更します。

8 システムをリブートします。

```
# reboot -- -r
```

例 5–1 リブート後も保持する方法で IPv4 アドレスとその他のネットワークパラメータを変更する

この例では、別のサブネットに移動されたシステムについて、次のネットワークパラメータをどのように変更するかを示しています。

- プライマリネットワークインタフェース `eri0` の IP アドレス (`10.0.0.14` から `192.168.55.14` に変更)。
- ホスト名 (`myhost` から `mynewhostname` に変更)。
- ネットマスク (`255.0.0.0` から `255.255.255.0` に変更)。
- デフォルトルーターのアドレス (`192.168.55.200` に変更)。

システムの現在のステータスを確認します。

```
# hostname
myhost
# ifconfig -a

lo0: flags=1000849 <UP,LOOPBACK,RUNNING,MULTICAST,IPv4> mtu 8232 index 1
```

```
        inet 127.0.0.1 netmask ff000000
eri0: flags=1000843 <UP,BROADCAST,RUNNING,MULTICAST,IPv4> mtu 1500 index 2
        inet 10.0.0.14 netmask ff000000 broadcast 10.255.255.255
        ether 8:0:20:c1:8b:c3
```

次に、システムのホスト名と `eri0` の IP アドレスを各ファイルで変更します。

```
# vi /etc/nodename
mynewhostname
```

Oracle Solaris 10 11/06 とそれ以前の Oracle Solaris 10 リリースでのみ、次のことを実行します。

```
# vi /etc/inet/ipnodes
192.168.55.14   mynewhostname      #moved system to 192.168.55 net

# vi /etc/inet/hosts
#
# Internet host table
#
127.0.0.1       localhost
192.168.55.14   mynewhostname        loghost
# vi /etc/hostname.eri0
192.168.55.14   netmask  255.255.255.0
```

最後に、ネットマスクおよびデフォルトルーターの IP アドレスを変更します。

```
# vi /etc/netmasks
...
192.168.55.0    255.255.255.0

# vi /etc/defaultrouter
192.168.55.200        #moved system to 192.168.55 net
#
```

これらの変更が完了したら、システムをリブートします。

```
# reboot -- -r
```

設定したばかりの構成がリブート後も保持されていることを確認します。

```
# hostname
mynewhostname
# ifconfig -a

lo0: flags=1000849 <UP,LOOPBACK,RUNNING,MULTICAST,IPv4> mtu 8232 index 1
        inet 127.0.0.1 netmask ff000000
eri0: flags=1000843 <UP,BROADCAST,RUNNING,MULTICAST,IPv4> mtu 1500 index 2
        inet 192.168.55.14 netmask ffffff00 broadcast 10.255.255.255
        ether 8:0:20:c1:8b:c3
```

例 5–2 現在のセッションの IP アドレスとホスト名を変更する

この例では、現在のセッションだけに適用するために、ホストの名前、プライマリネットワークインタフェースの IP アドレス、およびサブネットマスクを変更する方法を示しています。システムをリブートすると、以前の IP アドレスとサブネットマスクに戻ります。プライマリネットワークインタフェース `eri0` の IP アドレスが `10.0.0.14` から `192.168.34.100` に変わります。

```
# ifconfig -a

lo0: flags=1000849 <UP,LOOPBACK,RUNNING,MULTICAST,IPv4> mtu 8232 index 1
        inet 127.0.0.1 netmask ff000000
eri0: flags=1000843 <UP,BROADCAST,RUNNING,MULTICAST,IPv4> mtu 1500 index 2
        inet 10.0.0.14 netmask ff000000 broadcast 10.255.255.255
        ether 8:0:20:c1:8b:c3
# ifconfig eri0 192.168.34.100 netmask 255.255.255.0 broadcast + up
# vi /etc/nodename
mynewhostname

# ifconfig -a
lo0: flags=1000849 <UP,LOOPBACK,RUNNING,MULTICAST,IPv4> mtu 8232 index 1
        inet 127.0.0.1 netmask ff000000
eri0: flags=1000843 <UP,BROADCAST,RUNNING,MULTICAST,IPv4> mtu 1500 index 2
        inet 192.168.34.100 netmask ffffff00 broadcast 10.255.255.255
        ether 8:0:20:c1:8b:c3
# hostname
mynewhostname
```

例 5–3 現在のセッションの IPv4 アドレスを変更する (CIDR 表記を使用)

この例では、現在のセッションだけに適用するために、CIDR 表記を使用してホスト名と IP アドレスを変更する方法を示しています。システムをリブートすると、以前の IP アドレスとサブネットマスクに戻ります。プライマリネットワークインタフェース `eri0` の IP アドレスが `10.0.0.14` から `192.168.6.25/27` に変わります。

```
# ifconfig -a

lo0: flags=1000849 <UP,LOOPBACK,RUNNING,MULTICAST,IPv4> mtu 8232 index 1
        inet 127.0.0.1 netmask ff000000
eri0: flags=1000843 <UP,BROADCAST,RUNNING,MULTICAST,IPv4> mtu 1500 index 2
        inet 10.0.0.14 netmask ff000000 broadcast 10.255.255.255
        ether 8:0:20:c1:8b:c3
# ifconfig eri0 192.168.6.25/27 broadcast + up
# vi /etc/nodename
mynewhostname
# ifconfig -a

lo0: flags=1000849 <UP,LOOPBACK,RUNNING,MULTICAST,IPv4> mtu 8232 index 1
        inet 127.0.0.1 netmask ff000000
eri0: flags=1000843 <UP,BROADCAST,RUNNING,MULTICAST,IPv4> mtu 1500 index 2
        inet 192.168.06.25 netmask ffffffe0 broadcast 10.255.255.255
        ether 8:0:20:c1:8b:c3
# hostname
mynewhostname
```

IPv4 アドレスに CIDR 表記を使用するときは、ネットマスクを指定する必要はありません。`ifconfig` は、指定されたネットワーク接頭辞を使用して、ネットマスクを決定します。たとえば、`ifconfig` は、`192.168.6.0/27` ネットワークに対して `ffffffe0` というネットマスクを設定します。より一般的な `/24` 接頭辞指定を使用した場合は、`ffffff00` というネットマスクが設定されます。新しい IP アドレスを構成するときに、`ifconfig` に `/24` 接頭辞指定を使用することとネットマスク `255.255.255.0` を指定することは同等です。

参照 プライマリネットワークインタフェース以外のインタフェースの IP アドレスを変更する場合は、『Oracle Solaris の管理: 基本管理』および153 ページの「システムインストール後に物理インタフェースを構成する方法」を参照してください。

IPv4 ネットワーク上でのパケット転送と経路制御

このセクションでは、IPv4 ネットワーク上のルーターとホストに対して転送と経路制御を構成する手順および例を示します。

「パケット転送」は、ネットワーク上のシステム間で情報を共有するための基本的な方法です。転送元インタフェースと転送先インタフェース間で、パケットが転送されます。通常、2 つのインタフェースのシステムは異なります。ローカルでないインタフェースにコマンドを発行するかメッセージを送信すると、システムによりこれらのパケットがローカルネットワーク上に転送されます。その後、パケットヘッダー内に指定された宛先 IP アドレスを持つインタフェースが、ローカルネットワークからパケットを取得します。着信先アドレスがローカルネットワーク上に存在しない場合、パケットは隣接するネットワークに転送 (ホップ) されます。デフォルトでは、パケット転送は Oracle Solaris のインストール時に自動的に設定されます。

「経路制御」は、システムがパケットの送信先を決定する処理のことです。システム上の経路制御プロトコルは、ローカルネットワーク上のほかのシステムを「検出」します。送信元システムと送信先システムが同じローカルネットワーク上にある場合、これらのシステム間でパケットが通過する経路は「直接ルート」と呼ばれます。パケットが送信元システムのあとに少なくとも 1 つのホップを通過しなければならない場合、送信元システムと送信先システムの間の経路は「間接ルート」と呼ばれます。経路制御プロトコルは、転送先インタフェースへのパスを取得して、既知の送信ルートに関するデータをシステムの「経路制御テーブル」内に保持します。

「ルーター」は、複数の物理インタフェースを備える、特別に構成されたシステムです。ルーターは、この物理インタフェースを介して複数のローカルネットワークに接続されます。このため、ルーターは、経路制御プロトコルを実行するかどうかに関係なく、ホームの LAN を越えてパケットを転送できます。ルーターによるパケット転送方法の詳細は、67 ページの「ネットワーク上でのルーターの計画」を参照してください。

「経路制御プロトコル」は、システムでの経路制御活動を処理し、経路制御情報をほかのホストと交換することで、リモートネットワークへの既知のルートを維持管理します。経路制御プロトコルはルーターとホストの両方で実行できます。ホストの経路制御プロトコルは、ほかのルーターやホストの経路制御デーモンと通信します。ホストはこれらのプロトコルを利用して、パケットの転送先を決定します。ネットワークインタフェースが使用可能な場合、システムは経路制御デーモンとの通信を自動的に行います。これらのデーモンは、ネットワーク上のルーターの状態を監視し、ローカルネットワーク上のホストにルーターのアドレスを通知します。すべてではありませんが、一部の経路制御プロトコルは統計情報も保持します。この統計を使って、経路制御パフォーマンスを計測できます。パケット転送とは異なり、Oracle Solaris システム上で経路制御を明示的に構成する必要があります。

この節では、パケット転送の管理および IPv4 ルーターとホスト上での経路制御のタスクについて説明します。IPv6 が有効なネットワークでの経路制御については、183 ページの「IPv6 ルーターの構成」を参照してください。

Oracle Solaris でサポートされている経路制御プロトコル

経路制御プロトコルは、内部ゲートウェイプロトコル (IGP)、外部ゲートウェイプロトコル (EGP)、またはそれらの組み合わせに分類されます。「内部ゲートウェイプロトコル」は、一般的な管理制御下で、ネットワーク上のルーター間で経路制御情報を交換します。図 5–3 は、ルーターがルーティング情報を交換するために IGP を実行するネットワークトポロジを示します。「外部ゲートウェイプロトコル」を使用すると、ローカルのインターネットワークを外部ネットワークに接続するルーターは、外部ネットワーク上の別のルーターと情報を交換できます。たとえば、企業ネットワークを ISP に接続するルーターは、EGP を実行して、ISP の対応するルーターと経路制御情報を交換します。BGP (Border Gateway Protocol) は、さまざまな組織や IGP 間での経路制御情報送信に使用される、一般的な EGP です。

次の表に、Oracle Solaris 経路制御プロトコルおよび各プロトコルに関連するマニュアルの場所についての情報を示します。

表 5–1 Oracle Solaris ルーティングプロトコル

プロトコル	関連するデーモン	説明	説明
経路制御情報プロトコル (RIP)	`in.routed`	IPv4 パケットの経路制御および経路制御テーブルの維持を行う IGP	124 ページの「IPv4 ルーターの構成方法」
ICMP (Internet Control Message Protocol) ルーター発見	`in.routed`	ホストがネットワーク上のルーターの存在を検索するために使用します	137 ページの「単一インタフェースホストで静的ルーティングを有効にする方法」および 139 ページの「単一インタフェースホストで動的経路制御を有効にする方法」

表 5–1 Oracle Solaris ルーティングプロトコル (続き)

プロトコル	関連するデーモン	説明	説明
RIPng (Routing Information Protocol, next generation) プロトコル	`in.ripngd`	IPv6 パケットの経路制御および経路制御テーブルの維持を行う IGP	184 ページの「IPv6 対応のルーターを構成する方法」
ND (Neighbor Discovery) プロトコル	`in.ndpd`	IPv6 ルーターの存在を通知し、ネットワーク上の IPv6 ホストの存在を検索します	177 ページの「IPv6 インタフェースの構成」

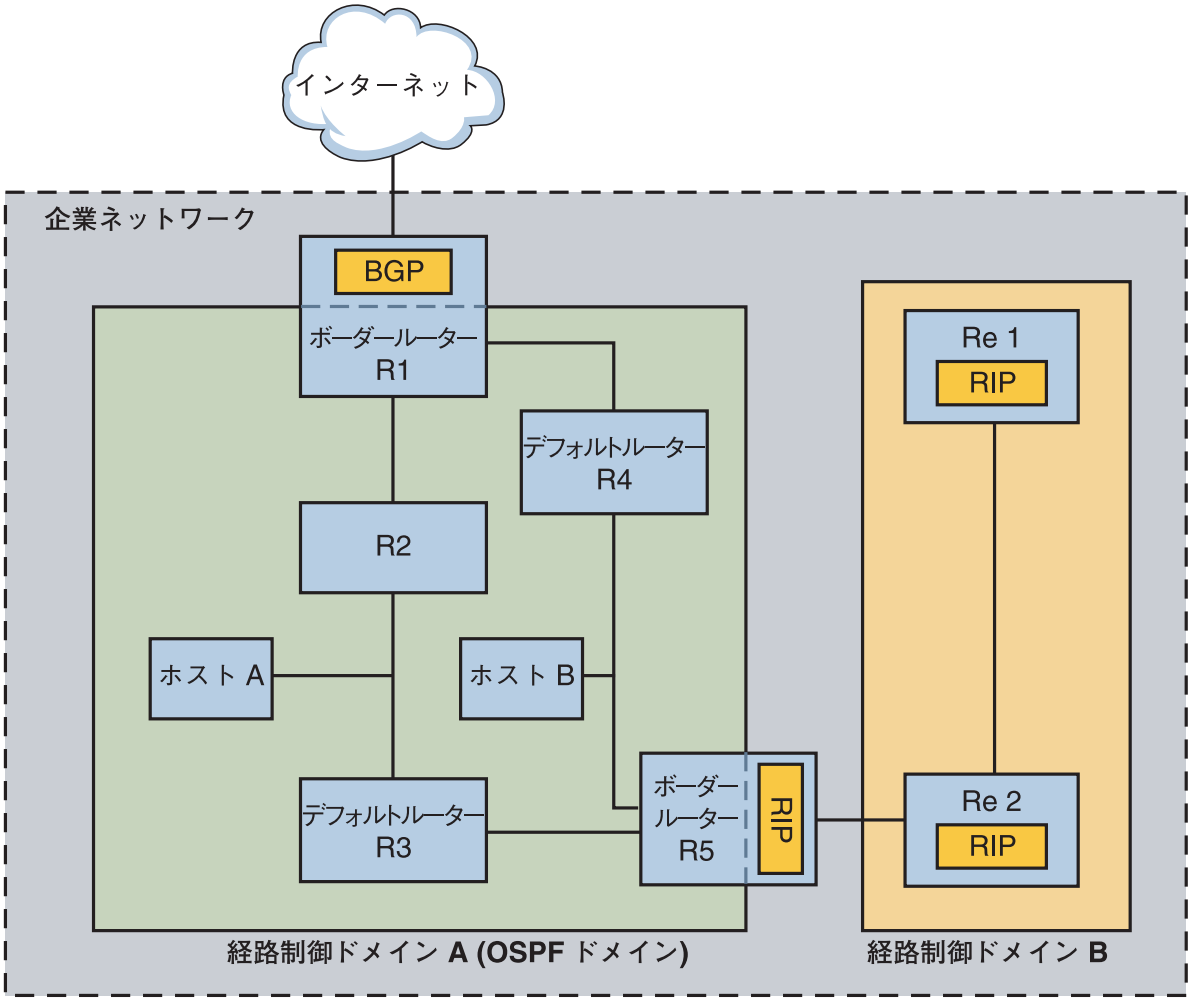
Oracle Solaris では、オープンソースの Quagga 経路制御プロトコル群もサポートされます。これらのプロトコルは、SFW 統合ディスクから入手できますが、メインの Oracle Solaris ディストリビューションには含まれていません。次の表に、Quagga プロトコルを示します。

表 5–2 オープンソースの Quagga プロトコル

プロトコル	デーモン	説明
RIP プロトコル	`ripd`	IPv4 距離ベクトル型 IGP。IPv4 パケットの経路制御および近傍への経路制御テーブルの通知を行います。
RIPng	`ripngd`	IPv6 距離ベクトル型 IGP。IPv6 パケットの経路制御および経路制御テーブルの維持を行います。
OSPF (Open Shortest Path First) プロトコル	`ospfd`	パケットの経路制御および高可用性ネットワークのための IPv4 リンク状態型 IGP。
BGP (Border Gateway Protocol)	`bgpd`	管理ドメインを越える経路制御のための IPv4 および IPv6 EGP。

次の図に、Quagga 経路制御プロトコルを使用する自律システムを示します。

図 5–2　Quagga プロトコルを実行する企業ネットワーク



この図は、2 つのルーティングドメイン A と B に分割された企業ネットワーク自律システムを示しています。ルーティングドメインは、管理目的で、またはドメインが単一のルーティングプロトコルを使用するために、統合的なルーティングポリシーを使用するインターネットワークです。この図のドメインはどちらも、Quagga プロトコル群の経路制御プロトコルを実行します。

経路制御ドメイン A は OSPF ドメインであり、単一の OSPF ドメイン ID で管理されます。このドメイン内のシステムはすべて、内部ゲートウェイプロトコルとして OSPF を実行します。内部のホストおよびルーターに加え、ドメイン A には 2 つのボーダールーターがあります。

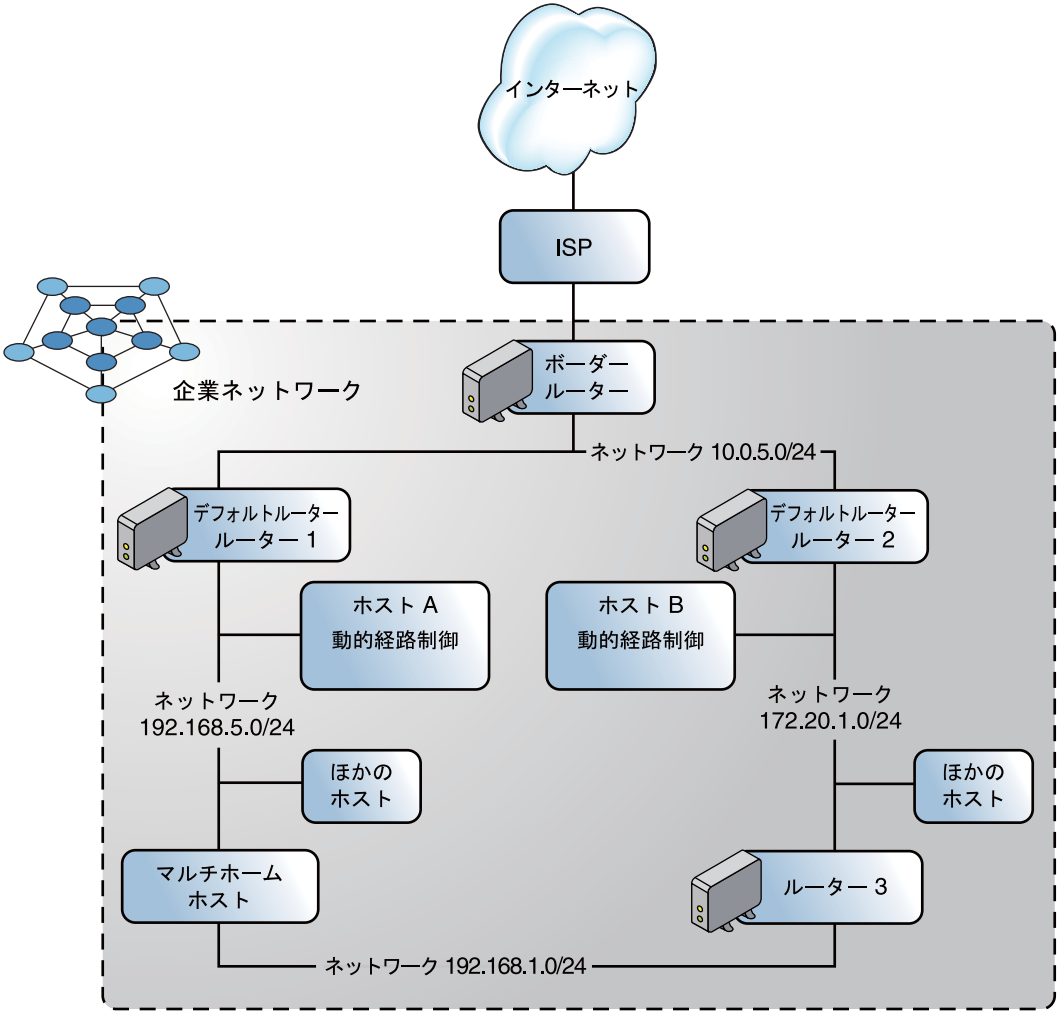
ボーダールーター R1 は、企業ネットワークを ISP に接続し、最終的にはインターネットに接続します。企業ネットワークと外部の間の通信を容易にするために、R1 はその外部に面したネットワークインタフェース上で BGP を実行します。ボーダールーター R5 は、ドメイン A とドメイン B を接続します。ドメイン B 上のシステムはすべて、内部ゲートウェイプロトコルとして RIP で管理されます。したがって、ボーダールーター R5 は、ドメイン A に面したインタフェース上では OSPF を実行し、ドメイン B に面したインタフェース上では RIP を実行する必要があります。

Quagga プロトコルの詳細については、Quagga Routing Suite の Webサイト `http://www.nongnu.org/quagga/index.html` を参照してください。

IPv4 自律システムのトポロジ

複数のルーターとネットワークを持つサイトでは、通常そのネットワークトポロジは単一のルーティングドメイン、つまり「自律システム (AS: Autonomous System)」として管理されます。次の図に、小規模な AS と見なすことのできる典型的なネットワークトポロジを示します。この節では、すべての例にこのトポロジを使用します。

図 5–3　複数の IPv4 ルーターを備えた自律システム



この図は、3 つのローカルネットワーク 10.0.5.0、172.20.1.0、および 192.168.5 に分割された AS を示しています。4 台のルーターが、パケット転送と経路制御の役目を分担しています。この AS には、次の種類のシステムが含まれています。

- 「ボーダールーター」は、インターネットなどの外部ネットワークに AS を接続します。ボーダールーターは、ローカル AS 上で実行中の IGP 外部のネットワークとの相互接続を行います。ボーダールーターは、BGP (Border Gateway Protocol) などの EGP を実行して、ISP のルーターなどの外部ルーターと情報を交換できます。図 5–3 では、ボーダールーターのインタフェースは、内部ネットワーク 10.0.5.0 およびサービスプロバイダへの高速ルーターに接続されています。

  ボーダールーターの構成については、オープンソースの Quagga のマニュアル (http://www.quagga.net/docs/docs-info.php#SEC72) で BGP に関する部分を参照してください。

  BGP を使用して AS をインターネットに接続する場合は、地域のインターネットレジストリから自律システム番号 (ASN) を取得するようにしてください。ASN の取得方法に関するガイドラインは、American Registry for Internet Numbers (ARIN) などの地域レジストリから提供されています。たとえば、 ARIN Number Resource Policy Manual (https://www.arin.net/policy/nrpm.html#five) には、米国およびカナダの自律システムの ASN を取得する方法が記載されています。または、使用している ISP が ASN を代行取得できる場合もあります。
- 「デフォルトルーター」は、ローカルネットワーク上のすべてのシステムに関する経路制御情報を維持管理します。通常、これらのルーターは、RIP などの IGP を実行します。図 5–3 では、ルーター 1 のインタフェースは内部ネットワーク 10.0.5.0 および内部ネットワーク 192.168.5 に接続されています。ルーター 1 は、192.168.5 のデフォルトルーターとしても機能します。ルーター1 は、192.168.5 上の全システムの経路制御情報を維持し、ボーダールーターなどのほかのルーターへの経路制御を行います。ルーター 2 のインタフェースは、内部ネットワーク 10.0.5.0 および内部ネットワーク 172.20.1 に接続します。

  デフォルトルーターを構成する例については、例 5–4 を参照してください。
- 「パケット転送ルーター」は、パケットを転送しますが、経路制御プロトコルは実行しません。このタイプのルーターは、ある単一のネットワークに接続されたインタフェースのひとつからパケットを受信します。その後、これらのパケットは、ルーターの別のインタフェースを経由して別のローカルネットワークに転送されます。図 5–3 では、ルーター 3 はパケット転送ルーターで、ネットワーク 172.20.1 および 192.168.5 に接続されています。
- 「マルチホームホスト」は、同一のネットワークセグメントに接続された複数のインタフェースを備えています。マルチホームホストはパケットの転送が可能です。これは、Oracle Solaris を実行するすべてのシステムでデフォルトに設定されています。図 5–3 のマルチホームホストは、両方のインタフェースがネットワーク 192.168.5 に接続されています。マルチホームホストを構成する例については、例 5–6 を参照してください。

- 「単一インタフェースホスト」は、パケット転送だけでなく、重要な構成情報の受信もローカルルーターに依存します。図 5–3 では、192.168.5 ネットワーク上のホスト A は動的ルーティングを実装しており、172.20.1 ネットワーク上のホスト B は静的ルーティングを実装しています。動的経路制御を実行するホストを構成する方法については、139 ページの「単一インタフェースホストで動的経路制御を有効にする方法」を参照してください。静的経路制御を実行するホストを構成する方法については、137 ページの「単一インタフェースホストで静的ルーティングを有効にする方法」を参照してください。

IPv4 ルーターの構成

このセクションでは、IPv4 ルーターを構成する手順と例を説明します。IPv6 対応のルーターを構成する方法については、184 ページの「IPv6 対応のルーターを構成する方法」を参照してください。

ルーターは複数のネットワーク間のインタフェースを提供するため、ルーターの物理ネットワークインタフェースごとに一意の名前および IP アドレスを割り当てる必要があります。これで、各ルーターは、そのプライマリネットワークインタフェースのホスト名と IP アドレスに加えて、増設した各ネットワークインタフェースについて少なくとも 1 つずつ、一意な名前と IP アドレスを持つことになります。

次の手順を使えば、物理インタフェースが 1 つだけのシステム (デフォルトではホスト) をルーターとして構成することもできます。システムを PPP リンクの 1 つのエンドポイントとして使用するような場合、単一インタフェースのシステムをルーターとして構成する場合があります (『System Administration Guide: Network Services』の「Planning a Dial-up PPP Link」を参照)。

注 – Oracle Solaris システムのインストール時にルーターのすべてのインタフェースを構成できます。手順については、『Oracle Solaris 10 1/13 インストールガイド: 基本インストール』を参照してください。

▼ IPv4 ルーターの構成方法

次の手順では、システムのインストール後にルーターのインタフェースを構成していることを想定しています。

始める前に　ルーターをネットワークに物理的に設置してから、ローカルファイルモードで動作するようにルーターを構成します。詳細は、107 ページの「ローカルファイルモードの場合のホストの構成方法」を参照してください。これで、ネットワーク構成サーバーがダウンしても、ルーターが確実にブートされるようになります。

1 ルーターとして構成されるシステムで、**Primary Administrator** の役割を引き受けるか、スーパーユーザーになります。

Primary Administrator 役割には、Primary Administrator プロファイルが含まれます。役割を作成してユーザーに役割を割り当てるには、『Oracle Solaris の管理: 基本管理』の第 2 章「Solaris 管理コンソールの操作 (タスク)」を参照してください。

2 **Solaris 10 1/06** リリース以降では、**dladm show-link** コマンドを使って、ルーターに物理的に取り付けられているインタフェースを判定します。

```
# dladm show-link
```

次の `dladm show-link` コマンドの出力例は、4 つのインタフェースを持つ `qfe` NIC と 2 つの `bge` インタフェースがシステム上で物理的に使用可能であることを示しています。

```
qfe0            type: legacy    mtu: 1500       device: qfe0
qfe1            type: legacy    mtu: 1500       device: qfe1
qfe2            type: legacy    mtu: 1500       device: qfe0
qfe3            type: legacy    mtu: 1500       device: qfe1
bge0            type: non-vlan  mtu: 1500       device: bge0
bge1            type: non-vlan  mtu: 1500       device: bge1
```

3 インストール時に、ルーター上のどのインタフェースが構成され、**plumb** されたのかを確認します。

```
# ifconfig -a
```

次の `ifconfig -a` コマンドの出力例は、インストール時にインタフェース `qfe0` が構成されたことを示しています。このインタフェースは `172.16.0.0` ネットワークにあります。`qfe` NIC の残りのインタフェースである `qfe1` - `qfe3`、および `bge` インタフェースは構成されていません。

```
lo0: flags=1000849 <UP,LOOPBACK,RUNNING,MULTICAST,IPv4> mtu 8232 index 1
        inet 127.0.0.1 netmask ff000000
qfe0: flags=1000843 <UP,BROADCAST,RUNNING,MULTICAST,IPv4> mtu 1500 index 2
        inet 172.16.26.232 netmask ffff0000 broadcast 172.16.26.255
        ether 0:3:ba:11:b1:15
```

4 別のインタフェースを構成して **plumb** します。

```
# ifconfig interface plumb
```

たとえば、`qfe1` の場合は、次のように入力します。

```
# ifconfig qfe1 plumb
```

注 – `ifconfig` コマンドを使用して明示的に構成されたインタフェースは、リブート後には保持されません。

---

5 インタフェースに **IPv4** アドレスとネットマスクを割り当てます。

注意 – IPv4 ルーターがその IP アドレスを DHCP 経由で受け取るように構成することもできますが、これは十分に経験を積んだ DHCP システム管理者だけが行うようにしてください。

```
# ifconfig interface IPv4-address netmask netmask
```

たとえば、IP アドレス 192.168.84.3 を qfe1 に割り当てるには、次のいずれかを実行します。

- 従来の IPv4 の表記法を使用して、次のように入力します。

  ```
  # ifconfig qfe1 192.168.84.3 netmask 255.255.255.0
  ```

- CIDR の表記法を使用して、次のように入力します。

  ```
  # ifconfig qfe1 192.168.84.3/24
  ```

  接頭辞 /24 により、自動的にネットマスク 255.255.255.0 が qfe1 に割り当てられます。CIDR の接頭辞と、それと同等の 10 進ドット表記のネットマスクの表については、図 2–2 を参照してください。

6 (オプション) リブート後もインタフェースの構成が保持されるようにするには、追加の物理インタフェースごとに **/etc/hostname.***interface* ファイルを作成します。

たとえば、/etc/hostname.qfe1 ファイルと /etc/hostname.qfe2 ファイルを作成します。次に、ホスト名 timbuktu を /etc/hostname.qfe1 ファイルに入力し、ホスト名 timbuktu-201 を /etc/hostname.qfe1 ファイルに入力します。個々のインタフェースを構成する方法の詳細については、153 ページの「システムインストール後に物理インタフェースを構成する方法」を参照してください。

このファイルを作成したあとは、必ず構成リブートを実行してください。

```
# reboot -- -r
```

7 各インタフェースのホスト名と IP アドレスを **/etc/inet/hosts** ファイルに追加します。

次に例を示します。

```
172.16.26.232      deadsea        #interface for network 172.16.0.0
192.168.200.20     timbuktu       #interface for network 192.168.200
192.168.201.20     timbuktu-201   #interface for network 192.168.201
192.168.200.9      gobi
192.168.200.10     mojave
192.168.200.110    saltlake
192.168.200.12     chilean
```

インタフェース timbuktu と timbuktu-201 は、同じシステムにあります。timbuktu-201 のネットワークアドレスが、timbuktu のネットワークインタフェースとは異なる点に注意してください。この違いは、ネットワーク 192.168.201

の物理ネットワークメディアが timbuktu-201 ネットワークインタフェースに接続されているのに対し、ネットワーク 192.168.200 のメディアは timbuktu インタフェースに接続されているためです。

8 **Solaris 10 11/06 以前の Solaris 10 リリースの場合のみ、新しいインタフェースのそれぞれの IP アドレスとホスト名を、/etc/inet/ipnodes ファイルまたは同等の ipnodes データベースに追加します。**

次に例を示します。

```
vi /etc/inet/ipnodes
172.16.26.232      deadsea        #interface for network 172.16.0.0
192.168.200.20     timbuktu       #interface for network 192.168.200
192.168.201.20     timbuktu-201   #interface for network 192.168.201
```

9 このルーターがサブネット化されたいずれかのネットワークに接続されている場合は、ネットワーク番号とネットマスクを /etc/inet/netmasks ファイルに追加します。

- 192.168.83.0 など、従来の IPv4 アドレス表記法の場合は、次のように入力します。

  ```
  192.168.83.0    255.255.255.0
  ```

- CIDR アドレスの場合は、/etc/inet/netmask ファイルのエントリに接頭辞の 10 進ドット表記のバージョンを使用します。ネットワークの接頭辞とそれと同等の 10 進ドット表記については、図 2–2 を参照してください。たとえば、192.168.3.0/22 という CIDR ネットワーク接頭辞を表現するには、/etc/netmasks ファイルで次のエントリを使用します。

  ```
  192.168.3.0 255.255.252.0
  ```

10 ルーターで **IPv4** パケット転送を使用可能にします。

次のいずれかのコマンドを使用して、パケット転送を有効にします。

- routeadm コマンドを次のように使用します。

  ```
  # routeadm -e ipv4-forwarding -u
  ```

- 次のサービス管理機能 (SMF) コマンドを使用します。

  ```
  # svcadm enable ipv4-forwarding
  ```

これで、ルーターは、ローカルネットワークの外にもパケットを転送できます。さらにルーターは、ルートを手動で経路制御テーブルに追加できる機能である、「静的経路制御」もサポートします。このシステムで静的経路制御を使用する場合は、ルーターの構成はこれで終わりです。ただし、システム経路制御テーブルで、ルートを維持管理する必要があります。ルートの追加方法については、131 ページの「ルートの構成」および route(1M) のマニュアルページを参照してください。

11 (任意) 経路制御プロトコルを起動する。

経路制御デーモン `/usr/sbin/in.routed` は自動的に経路制御テーブルを更新します。このプロセスのことを「動的経路制御」と呼びます。次のいずれかの方法で、デフォルトの IPv4 経路制御プロトコルをオンに設定します。

- `routeadm` コマンドを次のように使用します。

  ```
  # routeadm -e ipv4-routing -u
  ```

- 次の SMF コマンドを使用して、RIP などの経路制御プロトコルを起動します。

  ```
  # svcadm enable route:default
  ```

  `in.routed` デーモンに関連付けられている SMF FMRI は `svc:/network/routing/route` です。

`routeadm` コマンドの詳細については、`routeadm(1M)` のマニュアルページを参照してください。

例 5–4 ネットワークのデフォルトルーターを構成する

この例では、複数のインタフェースを持つシステムを、デフォルトルーターにアップグレードする方法を示します。目標は、図 5–3 に示されたルーター 2 を、ネットワーク `172.20.1.0` のデフォルトルーターにすることです。ルーター 2 には有線ネットワーク接続が 2 つあり、1 つはネットワーク `172.20.1.0`、もう 1 つはネットワーク `10.0.5.0` に接続されています。この例では、ルーターはローカルファイルモードで動作するものとします。このモードについては、107 ページの「ローカルファイルモードの場合のホストの構成方法」を参照してください。

スーパーユーザーになるか、同等の役割になったあと、システムのインタフェースのステータスを調べます。Solaris 10 1/06 以降では、`dladm` コマンドを次のように使用できます。

```
# dladm show-link
ce0             type: legacy    mtu: 1500       device: ce0
bge0            type: non-vlan  mtu: 1500       device: bge0
bge1            type: non-vlan  mtu: 1500       device: bge1

# ifconfig -a
lo0: flags=1000849 <UP,LOOPBACK,RUNNING,MULTICAST,IPv4> mtu 8232 index 1
        inet 127.0.0.1 netmask ff000000
ce0: flags=1000843 <UP,BROADCAST,RUNNING,MULTICAST,IPv4> mtu 1500 index 2
        inet 172.20.1.10 netmask ffff0000 broadcast 172.20.10.100
        ether 8:0:20:c1:1b:c6
```

`dladm show-link` の出力は、システムで 3 つのリンクが使用可能であることを示しています。`ce0` インタフェースのみ、IP アドレスを伴う構成となっています。デフォルトルーターの構成を始めるために、まず `bge0` インタフェースを `10.0.5.0` ネットワークに物理的に接続します。次に、このインタフェースを plumb し、リブート後も保持されるようにします。

```
# ifconfig bge0 plumb
# ifconfig bge0 10.0.5.10/8 up
# ifconfig -a
lo0: flags=1000849 <UP,LOOPBACK,RUNNING,MULTICAST,IPv4> mtu 8232 index 1
        inet 127.0.0.1 netmask ff000000
ce0: flags=1000843 <UP,BROADCAST,RUNNING,MULTICAST,IPv4> mtu 1500 index 2
        inet 172.20.1.10 netmask ffff0000 broadcast 172.255.255.255
        ether 8:0:20:c1:1b:c6
bge0: flags=1000843 <UP,BROADCAST,RUNNING,MULTICAST,IPv4> mtu 1500 index 2
        inet 10.0.5.10 netmask ff000000 broadcast 10.255.255.255
        ether 8:0:20:e5:95:c4
 # vi /etc/hostname.bge0
10.0.5.10
255.0.0.0
```

再構成用ブートコマンドを使用して、システムをリブートします。

```
# reboot -- -r
```

続いて、新たに plumb したこのインタフェースの情報とその接続先ネットワークの情報を使用して、次のネットワークデータベースを構成します。

```
# vi /etc/inet/hosts
127.0.0.1       localhost
172.20.1.10        router2        #interface for network 172.20.1
10.0.5.10          router2-out    #interface for network 10.0.5
# vi /etc/inet/netmasks
172.20.1.0    255.255.0.0
10.0.5.0      255.0.0.0
```

最後に、SMF を使用してパケット転送を有効にしてから、`in.routed` 経路制御デーモンを有効にします。

```
# svcadm enable ipv4-forwarding
# svcadm enable route:default
```

これで、IPv4 パケット転送と RIP による動的ルーティングがルーター 2 で有効になりました。ただし、ネットワーク `172.20.1.0` のデフォルトルーターの構成はまだ完了していません。次の作業を行う必要があります。

- `172.10.1.10` の各ホストを変更して、それぞれの経路制御情報をこの新しいデフォルトルーターから取得するようにします。詳細については、137 ページの「単一インタフェースホストで静的ルーティングを有効にする方法」を参照してください。
- ルーター 2 のルーティングテーブルで、ボーダールーターへの静的ルートを定義します。詳細については、130 ページの「ルーティングテーブルとルーティングの種類」を参照してください。

ルーティングテーブルとルーティングの種類

ルーターとホストの両方で「経路制御テーブル」が管理されます。各システムの経路制御デーモンは、すべての既知のルートを使用してテーブルを更新します。システムのカーネルは、パケットをローカルネットワークに転送する前に、経路制御テーブルを読み取ります。経路制御テーブルには、システムのデフォルトのローカルネットワークも含め、システムで知られているネットワークの IP アドレスがリストされています。このテーブルには、既知の各ネットワークに対するゲートウェイシステムの IP アドレスもリストされています。「ゲートウェイ」とは、発信されるパケットを受け取り、ローカルネットワークより 1 つ外側のホップに転送するシステムのことです。次に、IPv4 専用ネットワーク上にあるシステムの単純な経路制御テーブルを示します。

```
Routing Table: IPv4
  Destination           Gateway           Flags  Ref   Use   Interface
-------------------- -------------------- ----- ----- ------ ---------
default              172.20.1.10          UG        1    532   ce0
224.0.0.0            10.0.5.100           U         1      0   bge0
10.0.0.0             10.0.5.100           U         1      0   bge0
127.0.0.1            127.0.0.1            UH        1     57   lo0
```

Oracle Solaris システムでは、2 種類の経路制御を構成できます。静的なものと動的なものです。1 つのシステムに、これらの経路制御のどちらか一方を構成することも、両方を構成することもできます。「動的経路制御」を実装するシステムは、IPv4 ネットワークの場合は RIP、IPv6 ネットワークの場合は RIPng などの経路制御プロトコルを利用して、経路制御テーブルを管理します。「静的経路制御」だけを実行するシステムは、経路制御情報の取得や経路制御テーブルの更新を経路制御プロトコルに依存しません。その代わり、システムの既知のルートを、`route` コマンドを使って手動で管理する必要があります。詳細は、`route(1M)` のマニュアルページを参照してください。

ローカルネットワークまたは自律システムの経路制御を構成するときは、特定のルーターやホストでどの種類の経路制御をサポートするかを検討してください。

次の表に、ルーティングの種類と、それぞれの種類のルーティングを適用するのに最も適したネットワークの条件を示します。

ルーティングの種類	最適な使用対象
静的	小規模なネットワーク、デフォルトルーターから経路を取得するホスト、および、隣接する数ホップの範囲にある 1 つか 2 つのルーターに関する情報のみを必要とするデフォルトルーター。
動的	より規模の大きいインターネットワーク、多数のホストを含むローカルネットワーク上のルーター、および、大規模な自律システム上のホスト。動的ルーティングは、ほとんどのネットワークのシステムに最適です。

ルーティングの種類	最適な使用対象
静的経路制御と動的経路制御の組み合わせ	静的に経路制御されるネットワークと動的に経路制御されるネットワークを接続するルーター、および、内部の自律システムと外部のネットワークを接続するボーダールーター。1 つのシステムで静的ルーティングと動的ルーティングの両方を組み合わせることは、一般的に行われています。

図 5–3 に示された AS は、静的ルーティングと動的ルーティングの両方を組み合わせて使用しています。

ルートの構成

IPv4 ネットワークに動的経路制御を実装するには、`routeadm` コマンドまたは `svcadm` コマンドを使用して `in.routed` 経路制御デーモンを起動します。手順については、124 ページの「IPv4 ルーターの構成方法」を参照してください。動的経路制御は、ほとんどのネットワークおよび自律システムに最適な方法です。ただし、使用しているネットワークトポロジやネットワーク上の特定のシステムで、静的経路制御が必要になる場合もあります。その場合は、システムの経路制御テーブルを手動で編集して、ゲートウェイへの既知のルートを反映させる必要があります。次の手順は、静的ルートを追加する方法を示しています。

注 – 同じ宛先へのルートが 2 つあっても、システムで負荷分散やフェイルオーバーが自動的に行われるわけではありません。これらの機能が必要な場合は、第 27 章「IPMP の紹介 (概要)」に説明されている IPMP を使用します。

▼ ルーティングテーブルに静的ルートを追加する方法

1 経路制御テーブルの現在の状態を表示します。

通常のユーザーアカウントを使用して、次の形式の `netstat` コマンドを実行します。

```
% netstat -rn
```

次のような出力が表示されます。

```
Routing Table: IPv4
  Destination           Gateway           Flags  Ref   Use   Interface
-------------------- -------------------- ----- ----- ------ ---------
192.168.5.125        192.168.5.10          U      1   5879   ipge0
224.0.0.0            198.168.5.10          U      1  0       ipge0
default              192.168.5.10          UG     1  91908
127.0.0.1            127.0.0.1             UH     1  811302   lo0
```

2 **Primary Administrator** の役割を引き受けるか、スーパーユーザーになります。

Primary Administrator 役割には、Primary Administrator プロファイルが含まれます。役割を作成してユーザーに役割を割り当てるには、『Oracle Solaris の管理: 基本管理』の第 2 章「Solaris 管理コンソールの操作 (タスク)」を参照してください。

3 (オプション) 経路制御テーブル内の既存のエントリを消去します。

```
# route flush
```

4 リブート後も保持されるルートを追加します。

```
# route -p add -net network-address -gateway gateway-address
```

`-p`	リブート後も保持される必要のあるルートを作成します。現在のセッションだけに有効なルートを作成する場合は、`-p` オプションを使用しないでください。
`add`	そのあとに指定されているルートを追加することを示します。
`-net` *network-address*	*network-address* で指定されたアドレスを持つネットワークへのルートであることを示します。
`-gateway` *gateway-address*	指定されたルートのゲートウェイシステムの IP アドレスが *gateway-address* であることを示します。

例 5–5 ルーティングテーブルに静的ルートを追加する

次の例は、システムに静的ルートを追加する方法を示しています。このシステムは、図 5–3 で示されている、172.20.1.0 ネットワークのデフォルトルーターであるルーター 2 です。例 5–4 では、ルーター 2 は動的ルーティング用に構成されています。ネットワーク 172.20.1.0 上のホストのデフォルトルーターとしてルーター 2 の機能を向上させるには、AS のボーダールーター 10.0.5.150 への静的ルートを追加する必要があります。

ルーター 2 のルーティングテーブルを表示するために、次の手順を実行します。

```
# netstat -rn
Routing Table: IPv4
  Destination           Gateway           Flags  Ref   Use   Interface
-------------------- -------------------- ----- ----- ------ ---------
default              172.20.1.10          UG        1    249 ce0
224.0.0.0            172.20.1.10          U         1      0 ce0
10.0.5.0             10.0.5.20            U         1     78 bge0
127.0.0.1            127.0.0.1            UH        1     57 lo0
```

このルーティングテーブルは、ルーター 2 に既知のルートが 2 つあることを示しています。デフォルトのルートは、ルーター 2 の 172.20.1.10 インタフェースをゲートウェイとして使用します。2 番目のルート 10.0.5.0 は、ルーター 2 で実行中の in.routed デーモンによって検出されました。このルートのゲートウェイはルーター 1 で、その IP アドレスは 10.0.5.20 です。

ネットワーク 10.0.5.0 にはボーダールーターとして機能するゲートウェイがあります。このネットワークへのルートをもう 1 つ追加するには、次の手順を実行します。

```
# route -p add -net 10.0.5.0/24 -gateway 10.0.5.150
add net 10.0.5.0: gateway 10.0.5.150
```

これで、IP アドレス 10.0.5.150/24 を持つボーダールーターへのルートが、ルーティングテーブルに追加されました。

```
# netstat -rn
Routing Table: IPv4
  Destination           Gateway           Flags  Ref   Use   Interface
-------------------- -------------------- ----- ----- ------ ---------
default              172.20.1.10          UG        1    249 ce0
224.0.0.0            172.20.1.10          U         1      0 ce0
10.0.5.0             10.0.5.20            U         1     78 bge0
10.0.5.0             10.0.5.150           U         1    375 bge0
127.0.0.1            127.0.0.1            UH        1     57 lo0
```

マルチホームホストの構成

Oracle Solaris では、複数のインタフェースを持つシステムはマルチホームホストであると見なされます。マルチホームホストのインタフェースは、異なる物理ネットワーク上または同じ物理ネットワーク上のさまざまなサブネットに接続します。

複数のインタフェースが同じサブネットに接続しているシステムでは、最初にそれらのインタフェースを IPMP グループ内に構成する必要があります。そうしない場合、システムはマルチホームホストになることができません。IPMP の詳細については、パート V「IPMP」を参照してください。

マルチホームホストは IP パケットを転送しませんが、ルーティングプロトコルを実行するように構成できます。一般に、次のような種類のシステムをマルチホームホストとして構成します。

一般に、次のような種類のシステムをマルチホームホストとして構成します。

- NFS サーバー、特に大規模なデータセンターとして機能する NFS サーバーを複数のネットワークに接続することによって、多数のユーザー間でファイルを共有できるようになります。この種のサーバーはルーティングテーブルを備えている必要はありません。
- データベースサーバーは、NFS サーバーと同様に、多数のユーザーにリソースを提供する目的で複数のネットワークインタフェースを持つことができます。
- ファイアウォールゲートウェイは、企業のネットワークとインターネットなどの公共ネットワークとの間の接続を提供するシステムです。管理者は、セキュリティーの手段としてファイアウォールを設定します。ファイアウォールとして構成されたホストは、ホストのインタフェースに接続されたネットワーク間でのパケット交換を行いません。ただしこの場合でも、承認ユーザーに対する ssh など、ホストは標準的な TCP/IP サービスを提供します。

注 – いずれかのインタフェース上に異なる種類のファイアウォールがあるマルチホームホストの場合は、ホストのパケットが意図せずに中断されることがないよう注意してください。この問題は、特にステートフルなファイアウォールで発生します。解決策の 1 つは、ステートレスなファイアウォールを構成することです。ファイアウォールの詳細は、『Solaris のシステム管理: セキュリティーサービス』の「ファイアウォールシステム」またはサードパーティーのファイアウォールのマニュアルを参照してください。

▼ マルチホームホストの作成方法

1 予定するマルチホームホストで、**Primary Administrator** の役割を引き受けるか、スーパーユーザーになります。

Primary Administrator 役割には、Primary Administrator プロファイルが含まれます。役割を作成してユーザーに役割を割り当てるには、『Oracle Solaris の管理: 基本管理』の第 2 章「Solaris 管理コンソールの操作 (タスク)」を参照してください。

2 **Oracle Solaris** インストールの一部として構成されなかった追加の各ネットワークインタフェースを、構成して **plumb** します。

詳細は、153 ページの「システムインストール後に物理インタフェースを構成する方法」を参照してください。

3 マルチホームホストで **IP** 転送が使用可能になっていないことを確認します。

```
# routeadm
```

オプションを指定せずに `routeadm` コマンドを実行すると、経路制御デーモンの状態が報告されます。次の `routeadm` の出力は、IPv4 転送が有効になっていることを示しています。

```
   Configuration   Current             Current
                     Option   Configuration        System State
---------------------------------------------------------------
               IPv4 routing   disabled             disabled
               IPv6 routing   disabled             disabled
           IPv4 forwarding    enabled              disabled
           IPv6 forwarding    disabled             disabled

           Routing services   "route:default ripng:default"
```

4 システムでパケット転送が有効になっている場合は、パケット転送をオフに設定します。

次のコマンドのいずれかを使用します。

- `routeadm` コマンドの場合は、次のように入力します。

```
# routeadm -d ipv4-forwarding -u
```

- SMF を使用する場合は、次のように入力します。

```
# svcadm disable ipv4-forwarding
```

5 (オプション) マルチホームホストの動的経路制御をオンに設定します。
次のいずれかのコマンドを使用して、in.routed デーモンを有効にします。

- routeadm コマンドの場合は、次のように入力します。

```
# routeadm -e ipv4-routing -u
```

- SMF を使用する場合は、次のように入力します。

```
# svcadm enable route:default
```

例 5–6 マルチホームホストの構成

次の例は、図 5–3 に示されているマルチホームホストを構成する方法を示しています。この例で、システムのホスト名は hostc です。このホストには 2 つのインタフェースがあり、両方ともネットワーク 192.168.5.0 に接続されています。

まず、システムのインタフェースのステータスを表示します。

```
# dladm show-link
hme0            type: legacy    mtu: 1500       device: hme0
qfe0            type: legacy    mtu: 1500       device: qfe0
qfe1            type: legacy    mtu: 1500       device: qfe1
qfe2            type: legacy    mtu: 1500       device: qfe2
qfe3            type: legacy    mtu: 1500       device: qfe3
# ifconfig -a
lo0: flags=1000849 <UP,LOOPBACK,RUNNING,MULTICAST,IPv4> mtu 8232 index 1
        inet 127.0.0.1 netmask ff000000
hme0: flags=1000843 <UP,BROADCAST,RUNNING,MULTICAST,IPv4> mtu 1500 index 2
      inet 192.168.5.82 netmask ff000000 broadcast 192.255.255.255
      ether 8:0:20:c1:1b:c6
```

dladm show-link コマンドの報告は、hostc にはインタフェースが 2 つあり、可能なリンクが合計 5 つあることを示しています。ただし、hme0 だけが plumb されています。hostc をマルチホームホストとして構成するには、qfe NIC の qfe0 または別のリンクを追加する必要があります。まず、qfe0 インタフェースを 192.168.5.0 ネットワークに物理的に接続します。次に、qfe0 インタフェースを plumb し、リブート後も保持されるようにします。

```
# ifconfig qf0 plumb
# ifconfig qfe0 192.168.5.85/8 up
# ifconfig -a
lo0: flags=1000849 <UP,LOOPBACK,RUNNING,MULTICAST,IPv4> mtu 8232 index 1
        inet 127.0.0.1 netmask ff000000
hme0: flags=1000843 <UP,BROADCAST,RUNNING,MULTICAST,IPv4> mtu 1500 index 2
        inet 192.168.5.82 netmask ff0000 broadcast 192.255.255.255
```

```
        ether 8:0:20:c1:1b:c6
qfe0: flags=1000843 <UP,BROADCAST,RUNNING,MULTICAST,IPv4> mtu 1500 index 2
        inet 192.168.5.85 netmask ff000000 broadcast 192.255.255.255
        ether 8:0:20:e1:3b:c4
 # vi /etc/hostname.qfe0
192.168.5.85
255.0.0.0
```

再構成用ブートコマンドを使用して、システムをリブートします。

```
# reboot -- -r
```

次に、`qfe0` インタフェースを `hosts` データベースに追加します。

```
# vi /etc/inet/hosts
127.0.0.1           localhost
192.168.5.82        host3    #primary network interface for host3
192.168.5.85        host3-2  #second interface
```

その後、`host3` でのパケット転送とルーティングの状態を確認します。

```
# routeadm
              Configuration   Current             Current
                     Option   Configuration       System State
---------------------------------------------------------------
               IPv4 routing   enabled             enabled
               IPv6 routing   disabled            disabled
             IPv4 forwarding   enabled             enabled
             IPv6 forwarding   disabled            disabled

           Routing services   "route:default ripng:default"
```

`routeadm` コマンドの報告は、`in.routed` デーモンによる動的ルーティングとパケット転送が現在有効になっていることを示しています。ただし、パケット転送を無効にする必要があります。

```
# svcadm disable ipv4-forwarding
```

134 ページの「マルチホームホストの作成方法」で説明されているように、`routeadm` コマンドを使用してパケット転送をオフに設定することもできます。パケット転送を無効にすると、`host3` はマルチホームホストになります。

単一インタフェースシステムの経路制御の構成

単一インタフェースホストは、何らかの形式の経路制御を実装する必要があります。ホストがそのルートを 1 つ以上のローカルデフォルトルーターから取得する場合は、静的経路制御を使用するようにホストを構成する必要があります。それ以外の場合は、ホストで動的経路制御を使用することをお勧めします。次の手順では、両方の種類のルーティングを有効にする方法を示します。

▼ 単一インタフェースホストで静的ルーティングを有効にする方法

この手順は、単一インタフェースホストで静的経路制御を有効にします。静的経路制御を使用するホストは、RIP などの動的経路制御プロトコルをいっさい実行しません。代わりに、ホストはデフォルトルーターのサービスを利用して経路制御情報を取得する必要があります。121 ページの「IPv4 自律システムのトポロジ」の図には、いくつかのデフォルトルーターとそのクライアントホストが示されています。特定のホストをインストールするときにデフォルトルーターの名前を指定した場合、そのホストはすでに、静的経路制御を使用するように構成されています。

注 – 次の手順を使用して、マルチホームホストで静的経路制御を構成することもできます。

`/etc/defaultrouter` ファイルの詳細については、243 ページの「/etc/defaultrouter ファイル」を参照してください。静的経路制御と経路制御テーブルの詳細については、130 ページの「ルーティングテーブルとルーティングの種類」を参照してください。

1. 単一インタフェースのホスト上で、**Primary Administrator** の役割を引き受けるか、スーパーユーザーになります。

   Primary Administrator 役割には、Primary Administrator プロファイルが含まれます。役割を作成してユーザーに役割を割り当てるには、『Oracle Solaris の管理: 基本管理』の第 2 章「Solaris 管理コンソールの操作 (タスク)」を参照してください。

2. ホストに **/etc/defaultrouter** ファイルが存在するかどうかを確認します。

   ```
   # cd /etc
   # ls | grep defaultrouter
   ```

3. テキストエディタを開き、**/etc/defaultrouter** ファイルを作成または変更します。

4. デフォルトルーターのエントリを追加します。

   ```
   # vi  /etc/defaultrouter
   router-IP
   ```

   ここで、*router-IP* は、ホストが使用するデフォルトルーターの IP アドレスを示しています。

5. 経路制御とパケット転送がホストで実行されていないことを確認します。

   ```
   # routeadm
      Configuration   Current             Current
                        Option   Configuration        System State
   ---------------------------------------------------------------
                  IPv4 routing   disabled             disabled
                  IPv6 routing   disabled             disabled
   ```

```
              IPv4 forwarding   disabled            disabled
              IPv6 forwarding   disabled             disabled

            Routing services   "route:default ripng:default"
```

6 デフォルトルーターのエントリをローカルの **/etc/inet/hosts** ファイルに追加します。

/etc/inet/hosts の構成については、112 ページの「IPv4 アドレスおよびその他のネットワーク構成パラメータを変更する方法」を参照してください。

例 5–7 単一インタフェースホストのデフォルトルーターと静的経路制御を構成する

次の例は、単一インタフェースホスト hostb に静的ルーティングを設定する方法を示しています。このホストは、図 5–3 に示されているネットワーク 172.20.1.0 上にあります。hostb は、そのデフォルトルーターとしてルーター 2 を使用する必要があります。

まず、スーパーユーザーまたは同等の役割として hostb にログインします。次に、ホストに /etc/defaultrouter ファイルが存在するかどうかを調べます。

```
# cd /etc
# ls | grep defaultrouter
```

grep で検出されなかったので、/etc/defaultrouter ファイルを作成する必要があります。

```
# vi /etc/defaultrouter
172.20.1.10
```

/etc/defaultrouter ファイルのエントリは、172.20.1.0 ネットワークに接続されているルーター 2 のインタフェースの IP アドレスです。次に、ホストでパケット転送や経路制御が現在有効になっているかどうかを確認します。

```
# routeadm
   Configuration   Current             Current
                     Option   Configuration        System State
---------------------------------------------------------------
                IPv4 routing   disabled             disabled
                IPv6 routing   disabled             disabled
              IPv4 forwarding   enabled              enabled
              IPv6 forwarding   disabled             disabled

            Routing services   "route:default ripng:default"
```

この特定のホストでは、パケット転送が有効になっています。次のように、これをオフに設定します。

```
# svcadm disable ipv4-forwarding
```

最後に、ホストの /etc/inet/hosts ファイルにこの新しいデフォルトルーターのエントリが存在することを確認します。

```
# vi /etc/inet/hosts
127.0.0.1           localhost
172.20.1.18         host2    #primary network interface for host2
172.20.1.10         router2  #default router for host2
```

▼ 単一インタフェースホストで動的経路制御を有効にする方法

動的経路制御は、ホストで経路制御を管理するためのもっとも簡単な方法です。動的経路制御を使用するホストは、IPv4 の場合は in.routed デーモンで提供される経路制御プロトコルを実行し、IPv6 の場合は in.ripngd デーモンで提供される経路制御プロトコルを実行します。単一インタフェースホストで IPv4 動的経路制御を有効にするには、次の手順を使用します。動的経路制御の詳細は、117 ページの「IPv4 ネットワーク上でのパケット転送と経路制御」を参照してください。

1. ホスト上で、**Primary Administrator** の役割を引き受けるか、スーパーユーザーになります。

   Primary Administrator 役割には、Primary Administrator プロファイルが含まれます。役割を作成してユーザーに役割を割り当てるには、『Oracle Solaris の管理: 基本管理』の第 2 章「Solaris 管理コンソールの操作 (タスク)」を参照してください。

2. **/etc/defaultrouter** ファイルがあることを確認します。

   ```
   # cd /etc
   # ls | grep defaultrouter
   ```

3. **/etc/defaultrouter** が存在する場合は、その中のエントリをすべて削除します。

   /etc/defaultrouter ファイルが空の場合、ホストは自動的に動的経路制御を使用します。

4. ホストでパケット転送と経路制御が有効になっているかどうかを確認します。

   ```
   # routeadm
      Configuration   Current             Current
                        Option   Configuration        System State
   ---------------------------------------------------------------
                  IPv4 routing   disabled             disabled
                  IPv6 routing   disabled             disabled
              IPv4 forwarding    enabled              enabled
              IPv6 forwarding    disabled             disabled

             Routing services   "route:default ripng:default"
   ```

5. パケット転送が有効になっている場合は、パケット転送をオフに設定します。

次のコマンドのいずれかを使用します。

- routeadm コマンドの場合は、次のように入力します。

```
# routeadm -d ipv4-forwarding -u
```

- SMF を使用する場合は、次のように入力します。

```
# svcadm disable ipv4-forwarding
```

6 ホストで経路制御プロトコルを有効にします。

次のコマンドのいずれかを使用します。

- routeadm コマンドの場合は、次のように入力します。

```
# routeadm -e ipv4-routing -u
```

- SMF を使用する場合は、次のように入力します。

```
# svcadm enable route:default
```

これで、IPv4 動的経路制御が有効になりました。ホストの経路制御テーブルは、in.routed デーモンによって動的に管理されます。

例 5–8 単一インタフェースホストで動的経路制御を実行する

次の例は、単一インタフェースホスト hosta に動的ルーティングを設定する方法を示しています。このホストは、図 5–3 に示されているネットワーク 192.168.5.0 上にあります。hosta は、そのデフォルトルーターとしてルーター 1 を現在使用しています。ただし、これからは hosta で動的経路制御を実行する必要があります。

まず、スーパーユーザーまたは同等の役割として hosta にログインします。次に、ホストに /etc/defaultrouter ファイルが存在するかどうかを調べます。

```
# cd /etc
# ls | grep defaultrouter
defaultrouter
```

grep で検出されたので、hosta には /etc/defaultrouter ファイルが存在します。

```
# vi /etc/defaultrouter
192.168.5.10
```

このファイルには 192.168.5.10 というエントリがあります。これはルーター 1 の IP アドレスです。静的経路制御を有効にするには、このエントリを削除します。次に、ホストでパケット転送と経路制御がすでに有効になっているかどうかを確認する必要があります。

```
# routeadm   Configuration   Current             Current
                     Option   Configuration       System State
---------------------------------------------------------------
               IPv4 routing   disabled            disabled
```

```
   IPv6 routing   disabled             disabled
IPv4 forwarding   disabled             disabled
IPv6 forwarding   disabled             disabled

Routing services   "route:default ripng:default"
```

hosta では、経路制御とパケット転送の両方がオフに設定されています。次のように経路制御をオンに構成して、hosta の動的経路制御の構成を完了します。

```
# svcadm enable route:default
```

トランスポート層サービスの監視と変更

トランスポート層プロトコル TCP、SCTP、および UDP は、Oracle Solaris の標準パッケージの一部です。通常、これらのプロトコルは、ユーザーの介入なしで正常に動作します。ただし、サイトの条件によっては、トランスポート層プロトコルの上で動作するサービスをログまたは変更しなければならない場合があります。次に、サービス管理機能 (SMF) を使ってこれらのサービスのプロファイルを変更する必要があります。SMF については、『Oracle Solaris の管理: 基本管理』の第 18 章「サービスの管理 (概要)」を参照してください。

inetd デーモンは、システムがブートされると、標準的なインターネットサービスを起動します。これらのサービスは、TCP や SCTP、UDP をそのトランスポート層プロトコルとして使用するアプリケーションなどです。SMF コマンドを使えば、既存のインターネットサービスの組み合わせを変更したり、新しいサービスを追加したりできます。inetd についての詳細は、252 ページの「inetd インターネットサービスデーモン」を参照してください。

トランスポート層プロトコルが関係する操作には、次の操作があります。

- すべての着信 TCP 接続を記録する
- トランスポート層プロトコル (たとえば、SCTP) の上で動作するサービスを追加する
- アクセス制御のために TCP ラッパー機能を構成する

inetd デーモンの詳細は、inetd(1M) のマニュアルページを参照してください。

▼ すべての着信 TCP 接続の IP アドレスを記録する方法

1 ローカルシステムで「ネットワーク管理者」役割になるか、スーパーユーザーになります。

役割には、認証と特権コマンドが含まれます。役割の詳細については、『Solaris のシステム管理: セキュリティーサービス』の「RBAC の構成 (タスクマップ)」を参照してください。

2 **inetd** で管理されるすべてのサービスに対して **TCP** トレースを使用可能にします。

```
# inetadm -M tcp_trace=TRUE
```

▼ SCTP プロトコルを使用するサービスを追加する方法

SCTP トランスポートプロトコルは、TCP に類似した方法でアプリケーション層プロトコルにサービスを提供します。ただし、SCTP では 2 つのシステム間での通信が可能です。これらのシステムは、片方または両方がマルチホームであってもかまいません。SCTP 接続は「アソシエーション」と呼ばれます。アソシエーションでは、アプリケーションがデータを分割し、1 つまたは複数のメッセージストリームとして伝送します (マルチストリーム化)。SCTP 接続は、複数の IP アドレスを持つエンドポイントに到達できます。これは、テレフォニーアプリケーションにとって特に重要です。IP Filter や IPsec を使用する場合、SCTP のマルチホーム機能はセキュリティーの点で考慮を要します。考慮点については、`sctp(7P)` のマニュアルページを参照してください。

デフォルトで SCTP は Oracle Solaris に組み込まれています。したがって、構成を別に行う必要はありません。ただし、SCTP を使用するためには、一定のアプリケーション層サービスを明示的に構成しなければならない場合があります。このようなアプリケーションの例としては、`echo` や `discard` があります。次の手順は、ワンツーワンスタイルの SCTP ソケットを使用する echo サービスの追加方法を示しています。

注 – さらに、次の手順を使えば、TCP や UDP のトランスポート層プロトコル用のサービスを追加できます。

次のタスクでは、`inetd` デーモンによって管理される SCTP `inet` サービスを SMF リポジトリに追加します。さらに、タスクの後半では、サービス管理機能 (SMF) コマンドを使ってこのサービスを追加します。

- SMF コマンドについては、『Oracle Solaris の管理: 基本管理』の「SMF コマンド行管理ユーティリティー」を参照してください。
- 構文については、SMF コマンドのマニュアルページを参照してください (手順を参照)。
- SMF の詳細は、`smf(5)` のマニュアルページを参照してください。

始める前に　次の手順を実行する前に、サービスのマニフェストファイルを作成してください。この手順では、例として、`echo` サービス用のマニフェスト `echo.sctp.xml` を使用します。

1. システムファイルに対する書き込みアクセス権を持つユーザーアカウントでローカルシステムにログインします。

2. **`/etc/services`** ファイルを編集し、新しいサービスの定義を追加します。

   サービスを定義する構文は次のとおりです。

   service-name |*port/protocol* | *aliases*

3. 新しいサービスを追加します。

   サービスマニフェストが格納されているディレクトリに移り、次のように入力します。

   ```
   # cd dir-name
   # svccfg import service-manifest-name
   ```

   `svccfg` の詳しい構文については、`svccfg(1M)` のマニュアルページを参照してください。

   現在 `service.dir` ディレクトリにあるマニフェスト `echo.sctp.xml` を使用して、SCTP の新しい `echo` サービスを追加するとします。その場合、次のように入力します。

   ```
   # cd service.dir
   # svccfg import echo.sctp.xml
   ```

4. サービスマニフェストが追加されているか確認します。

   ```
   # svcs FMRI
   ```

   FMRI 引数には、サービスマニフェストの Fault Managed Resource Identifier (FMRI) を使用します。たとえば、SCTP の `echo` サービスの場合は、次のコマンドを使用します。

   ```
   # svcs svc:/network/echo:sctp_stream
   ```

   次のような出力が表示されます。

   ```
       STATE          STIME    FMRI
   disabled       16:17:00 svc:/network/echo:sctp_stream
   ```

svcs コマンドの詳細は、svcs(1) のマニュアルページを参照してください。

出力は、新しいサービスマニフェストが使用不可になっていることを示しています。

5 サービスの属性をリストして、変更を加える必要があるかどうかを決めます。

```
# inetadm -l FMRI
```

inetadm コマンドの詳細は、inetadm(1M) のマニュアルページを参照してください。

たとえば、SCTP echo サービスの場合は、次のように入力します。

```
# inetadm -l svc:/network/echo:sctp_stream
SCOPE    NAME=VALUE
             name="echo"
             endpoint_type="stream"
             proto="sctp"
             isrpc=FALSE
             wait=FALSE
             exec="/usr/lib/inet/in.echod -s"
         .
         .
         default  tcp_trace=FALSE
           default  tcp_wrappers=FALSE
```

6 新しいサービスを使用可能にします。

```
# inetadm -e FMRI
```

7 サービスが使用可能になっていることを確認します。

たとえば、新しい echo サービスの場合、次のように入力します。

```
# inetadm | grep sctp_stream
.
.
    enabled   online         svc:/network/echo:sctp_stream
```

例 5–9 SCTP トランスポートプロトコルを使用するサービスの追加

次の例では、使用するコマンドと、echo サービスで SCTP トランスポート層プロトコルを使用するために必要なファイルエントリを示します。

```
$ cat /etc/services
.
.
echo            7/tcp
echo            7/udp
echo            7/sctp

# cd service.dir

    # svccfg import echo.sctp.xml
```

```
# svcs network/echo*
    STATE          STIME    FMRI
    disabled       15:46:44 svc:/network/echo:dgram
    disabled       15:46:44 svc:/network/echo:stream
    disabled       16:17:00 svc:/network/echo:sctp_stream

# inetadm -l svc:/network/echo:sctp_stream
    SCOPE    NAME=VALUE
             name="echo"
             endpoint_type="stream"
             proto="sctp"
             isrpc=FALSE
             wait=FALSE
             exec="/usr/lib/inet/in.echod -s"
             user="root"
    default  bind_addr=""
    default  bind_fail_max=-1
    default  bind_fail_interval=-1
    default  max_con_rate=-1
    default  max_copies=-1
    default  con_rate_offline=-1
    default  failrate_cnt=40
    default  failrate_interval=60
    default  inherit_env=TRUE
    default  tcp_trace=FALSE
    default  tcp_wrappers=FALSE

# inetadm -e svc:/network/echo:sctp_stream

# inetadm | grep echo
    disabled  disabled       svc:/network/echo:stream
    disabled  disabled       svc:/network/echo:dgram
    enabled   online         svc:/network/echo:sctp_stream
```

▼ TCP ラッパーを使って TCP サービスのアクセスを制御する方法

「TCP ラッパー」は *tcpd* プログラムによって実装されます。TCP ラッパーは、送られてくるサービス要求とサービスデーモンの間で動作することによって、`ftpd` などのサービスデーモンにセキュリティー対策を追加します。TCP ラッパーは、正常および異常な接続の試みを記録します。さらに、TCP ラッパーはアクセス制御の機能を備えています。したがって、要求の発行元がどこかによって接続を許可することも拒否することもできます。TCP ラッパーを使えば、SSH、Telnet、FTP などのデーモンを保護できます。さらに、`sendmail` アプリケーションでも TCP ラッパーを使用できます。詳細は、『System Administration Guide: Network Services』の「Support for TCP Wrappers From Version 8.12 of sendmail」を参照してください。

1. ローカルシステムで、**Primary Administrator** の役割を引き受けるか、スーパーユーザーになります。

   Primary Administrator 役割には、Primary Administrator プロファイルが含まれます。役割を作成してユーザーに役割を割り当てるには、『Oracle Solaris の管理: 基本管理』の第 2 章「Solaris 管理コンソールの操作 (タスク)」を参照してください。

2. **TCP** ラッパーを使用可能にします。

   ```
   # inetadm -M tcp_wrappers=TRUE
   ```

3. **TCP** ラッパーのアクセス制御ポリシーを構成します (**hosts_access(3)** のマニュアルページを参照)。

   このマニュアルページは、Oracle Solaris CD-ROM と一緒にパッケージ化されている SFW CD-ROM の `/usr/sfw/man` ディレクトリにあります。

◆ ◆ ◆ 第 6 章

ネットワークインタフェースの管理 (作業)

この章には、ネットワークインタフェースに関する作業と関連情報を記載します。

ネットワークインタフェースの管理の新機能

この章では、Solaris 10 1/06 リリース以降のインタフェースの構成について説明します。Oracle Solaris の新機能すべての一覧や Oracle Solaris リリースの説明については、『Oracle Solaris 10 1/13 の新機能』を参照してください。

Solaris 10 1/06 では、次の新機能が導入されました。

- インタフェースのステータスを表示するための新しい `dladm` コマンドについては、153 ページの「システムインストール後に物理インタフェースを構成する方法」で概要を説明しています。
- VLAN サポートが GLDv3 インタフェースに拡張されています。158 ページの「仮想ローカルエリアネットワークの管理」の説明を参照してください。
- リンク集約のサポートの概要は、165 ページの「リンクアグリゲーションの概要」で説明しています。

Solaris 10 7/07 では、`/etc/inet/ipnodes` ファイルは廃止されました。個々の手順で説明されているとおり、`/etc/inet/ipnodes` は以前の Solaris 10 リリースにのみ使用してください。

インタフェースの管理 (タスクマップ)

次の表に、ネットワークインタフェースを構成するための各種作業の一覧を示します。VLAN やリンク集約などの特殊な構成のための作業も含まれます。表では、各タスクで実行する内容の説明と、タスクの具体的な実行手順が詳しく説明されている現在のマニュアル内のセクションを示しています。

タスク	説明	参照先
システムのインタフェースのステータスを確認します。	システムのすべてのインタフェースを一覧表示し、すでに plumb されているインタフェースを確認します。	151 ページの「インタフェースのステータスを取得する方法」
システムをインストールしたあとで、1 つのインタフェースを追加します。	別のインタフェースを構成することによってシステムをマルチホームホストまたはルーターに変更します。	153 ページの「システムインストール後に物理インタフェースを構成する方法」
SPARC: インタフェースの MAC アドレスが一意であることを確認します。	インタフェースが、システムの MAC アドレスではなく、出荷時に設定された MAC アドレスを使用して構成されていることを確認します (SPARC のみ)。	156 ページの「SPARC: インタフェースの MAC アドレスが一意であることを確認する方法」
仮想ローカルエリアネットワーク (VLAN) を計画します。	VLAN を作成する前に必要な計画タスクを実行します。	162 ページの「VLAN 構成を計画する方法」
VLAN を構成します。	ネットワーク上で VLAN を作成および変更します。	163 ページの「VLAN を構成する方法」
集約を計画します。	集約を構成する前に、集約を設計し、必要な計画タスクを実行します。	165 ページの「リンクアグリゲーションの概要」
集約を構成します。	リンク集約に関連するさまざまなタスクを実行します。	169 ページの「リンクアグリゲーションを作成する方法」
IPMP グループを計画および構成します。	IPMP グループのメンバーになっているインタフェースのフェイルオーバーおよびフェイルバックを構成します。	739 ページの「IPMP グループの計画を立てる方法」 741 ページの「複数のインタフェースを持つ IPMP グループを構成する方法」

物理インタフェースの管理の基礎

ネットワークインタフェースは、システムとネットワークの間の接続を提供します。Oracle Solaris ベースのシステムには、物理インタフェースと論理インタフェースという 2 つのタイプのインタフェースがあります。「物理インタフェース」はソフトウェアドライバとコネクタから成り、コネクタには、Ethernet ケーブルなどのネットワークメディアを接続します。物理インタフェースは、管理効率と可用性を高めるためにグループ化できます。「論理インタフェース」は、既存の物理インタフェースの上に論理的に構成され、通常、物理インタフェース上のアドレスの追加やトンネルの終端の作成に使用されます。

注 – 論理ネットワークインタフェースについては、論理ネットワークインタフェースを使用する作業で説明します。 たとえば、IPv6、IPMP、DHCP などに関する作業に含まれています。

ほとんどのコンピュータシステムでは、製造元によってメインシステムボードに少なくとも 1 つの物理インタフェースが組み込まれています。複数のインタフェースが組み込まれたシステムもあります。

システムには、組み込みインタフェースのほかに別個に購入したインタフェースを追加できます。別個に購入したインタフェースは、「ネットワークインタフェースカード」(NIC) と呼ばれます。NIC は製造元の指示に従って物理的に取り付けてください。

注 – NIC は「ネットワークアダプタ」とも呼ばれます。

システムのインストール中に、Oracle Solaris インストールプログラムは、物理的に取り付けられたすべてのインタフェースを検出し、各インタフェースの名前を表示します。インタフェースのリストから少なくとも 1 つのインタフェースを構成する必要があります。インストール中に最初に構成されるインタフェースが、「プライマリネットワークインタフェース」になります。プライマリネットワークインタフェースの IP アドレスは、`/etc/nodename` ファイルに保存されているシステムの構成済みのホスト名に関連付けられます。ただし、インストール中またはインストール後に、追加のインタフェースを構成できます。

ネットワークインタフェース名

各物理インタフェースは、一意のデバイス名によって識別されます。デバイス名の構文は、次のとおりです。

`<driver-name><instance-number>`

Oracle Solaris システム上のドライバ名には、ce、 hme、bge、 e1000g などの数多くのドライバ名があります。*instance-number* 変数には、システムにインストールされているドライバタイプのインタフェースの数に応じて、0 から *n* までの値を指定できます。

たとえば、ホストシステムとサーバーシステムの両方でプライマリネットワークインタフェースとして使用されることが多い 100BASE-TX Fast Ethernet インタフェースについて考えてみます。このインタフェースの一般的なドライバ名として、eri、qfe、hme などがあります。Fast Ethernet インタフェースをプライマリネットワークインタフェースとして使用する場合、eri0 や qfe0 などのデバイス名が使用されます。

eri や hme などの NIC にはインタフェースが 1 つしかありません。ただし、多くの種類の NIC には複数のインタフェースがあります。たとえば、Quad Fast Ethernet (qfe) カードには qfe0 から qfe3 まで 4 つのインタフェースがあります。

インタフェースを plumb する

インタフェースがシステムとネットワーク間のトラフィックを受け渡しできるようにするには、その前にインタフェースを「*plumb* する」必要があります。plumb するときには、インタフェースにデバイス名を関連付けます。次に、IP プロトコルでインタフェースを使用できるようにストリームをセットアップします。物理インタフェースと論理インタフェースの両方を plumb する必要があります。インタフェースは、ブート手順の一部として plumb されるか、ifconfig コマンドの適切な構文を使用して明示的に plumb されます。

インストール中にインタフェースを構成する場合、インタフェースは自動的に plumb されます。インストール中にシステムで追加のインタフェースを構成しない場合、それらのインタフェースは plumb されません。

Oracle Solaris インタフェースタイプ

Solaris 10 1/06 リリース以降、Oracle Solaris では次の 2 つのタイプのインタフェースがサポートされています。

- 旧式インタフェース – これらのインタフェースは、DLPI インタフェースと GLDv2 インタフェースです。旧式インタフェースのタイプには、eri、qfe、ce などがあります。dladm show-link コマンドを使用してインタフェースのステータスを確認すると、これらのインタフェースは、「legacy」(旧式) として報告されます。
- 非 **VLAN** インタフェース – これらのインタフェースは GLDv3 インタフェースです。

注 – 現在、GLDv3 は、次のインタフェースタイプでサポートされています。`bge`、`xge`、および `e1000g` です。

個々のネットワークインタフェースの管理

Oracle Solaris のインストール後に、次の目的でシステム上のインタフェースを構成、管理する場合があります。

- システムをアップグレードしてマルチホームホストにする。詳細については、133 ページの「マルチホームホストの構成」を参照してください。
- ホストをルーターに変更する。ルーターの構成については、124 ページの「IPv4 ルーターの構成」を参照してください。
- インタフェースを VLAN の一部として構成する。詳細については、158 ページの「仮想ローカルエリアネットワークの管理」を参照してください。
- インタフェースを集約のメンバーとして構成する。詳細については、165 ページの「リンクアグリゲーションの概要」を参照してください。
- IPMP グループにインタフェースを追加する。IPMP グループを構成する手順については、739 ページの「高可用性のための IPMP グループの使用」を参照してください

このセクションでは、Solaris 10 1/06 リリース以降での個々のネットワークインタフェースの構成について説明します。次のいずれかのグループに含めるインタフェースを構成する方法については、次の節を参照してください。

- VLAN に含めるインタフェースを構成する方法については、158 ページの「仮想ローカルエリアネットワークの管理」を参照してください。
- 集約に含めるインタフェースを構成する方法については、165 ページの「リンクアグリゲーションの概要」を参照してください。
- IPMP グループのメンバーとしてインタフェースを構成する方法については、739 ページの「高可用性のための IPMP グループの使用」を参照してください。

▼ インタフェースのステータスを取得する方法

Solaris 10 1/06 以降、この手順では、システム上で現在使用可能なインタフェースとそれらのステータスを調べる方法について説明します。この手順では、現在 plumb されているインタフェースを表示する方法も示します。以前の Solaris 10 3/05 リリースを使用している場合は、211 ページの「特定のインタフェースに関する情報を入手する方法」を参照してください。

1 インタフェースがあるシステムで、**Primary Administrator** の役割を引き受けるか、スーパーユーザーになります。

Primary Administrator 役割には、Primary Administrator プロファイルが含まれます。役割を作成してユーザーに役割を割り当てるには、『Oracle Solaris の管理: 基本管理』の第 2 章「Solaris 管理コンソールの操作 (タスク)」を参照してください。

2 現在システムにインストールされているインタフェースを調べます。

```
# dladm show-link
```

この手順で使用する `dladm` コマンドの詳細は、`dladm(1M)` のマニュアルページを参照してください。このコマンドは、それらのインタフェースが現在構成されているかどうかに関係なく、検出したすべてのインタフェースドライバについて報告します。

3 システム上で現在 **plumb** されているインタフェースを調べます。

```
# ifconfig -a
```

`ifconfig` コマンドには、インタフェースを plumb する機能など数多くの追加の機能があります。詳細については、`ifconfig(1M)` のマニュアルページを参照してください。

例 6–1 dladm コマンドを使用したインタフェースのステータスの取得

次の例は、`dladm` コマンドによるステータスの表示を示しています。

```
# dladm show-link
ce0             type: legacy    mtu: 1500       device: ce0
ce1             type: legacy    mtu: 1500       device: ce1
bge0            type: non-vlan  mtu: 1500       device: bge0
bge1            type: non-vlan  mtu: 1500       device: bge1
bge2            type: non-vlan  mtu: 1500       device: bge2
```

`dladm show-link` の出力は、ローカルホストで 4 つのインタフェースドライバが使用可能であることを示しています。`ce` インタフェースと `bge` インタフェースは、どちらも VLAN 用に構成できます。ただし、リンク集約で使用できるのは、非 `VLAN` タイプの GLDV3 インタフェースのみです。

次の例は、`ifconfig -a` コマンドによるステータスの表示を示しています。

```
# ifconfig -a
 lo0: flags=2001000849 <UP,LOOPBACK,RUNNING,MULTICAST,IPv4,VIRTUAL> mtu
8232 index 1
        inet 127.0.0.1 netmask ff000000
ce0: flags=1000843 <UP,BROADCAST,RUNNING,MULTICAST,IPv4>mtu 1500 index 3
         inet 192.168.84.253 netmask ffffff00 broadcast 192.168.84.255
        ether 0:3:ba:7:84:5e
bge0: flags=1004843 <UP,BROADCAST,RUNNING,MULTICAST,DHCP,IPv4>mtu 1500 index 2
         inet 10.8.57.39 netmask ffffff00 broadcast 10.8.57.255
        ether 0:3:ba:29:fc:cc
```

`ifconfig -a` コマンドの出力には、`ce0` と `bge0` という 2 つのインタフェースの統計のみが表示されます。この出力は、`ce0` と `bge0` のみが plumb され、ネットワークトラフィックで使用可能なことを示しています。これらのインタフェースは VLAN で使用できます。`bge0` は plumb されているので、このインタフェースを集約で使用することはできません。

▼ システムインストール後に物理インタフェースを構成する方法

始める前に
- 追加のインタフェースで使用する IPv4 アドレスを決定します。
- 構成する物理インタフェースが、システムに物理的に取り付けられていることを確認します。別個に購入した NIC ハードウェアのインストールについては、NIC に付属している製造元のマニュアルを参照してください。
- インタフェースをインストールした直後の場合は、次のタスクに進む前に、再ブートを実行してください。

1. インタフェースがあるシステムで、**Primary Administrator** の役割を引き受けるか、スーパーユーザーになります。

   Primary Administrator 役割には、Primary Administrator プロファイルが含まれます。役割を作成してユーザーに役割を割り当てるには、『Oracle Solaris の管理: 基本管理』の第 2 章「Solaris 管理コンソールの操作 (タスク)」を参照してください。

2. 現在システムにインストールされているインタフェースを調べます。

   ```
   # dladm show-link
   ```

3. 別のインタフェースを構成して **plumb** します。

   ```
   # ifconfig interface plumb up
   ```

   たとえば、`qfe0` の場合は、次のように入力します。

   ```
   # ifconfig qfe0 plumb up
   ```

   注 – `ifconfig` コマンドを使用して明示的に構成されたインタフェースは、リブート後には保持されません。

   ---

4. インタフェースに **IPv4** アドレスとネットマスクを割り当てます。

   ```
   # ifconfig interface IPv4-address netmask+netmask
   ```

   たとえば、`qfe0` の場合は、次のように入力します。

   ```
   # ifconfig
   qfe0 192.168.84.3 netmask + 255.255.255.0
   ```

注 – IPv4 アドレスは、一般的な IPv4 表記または CIDR 表記で指定できます。

5 新しく構成したインタフェースが **plumb** されて構成済みの状態 (「**UP**」) であることを確認します。

```
# ifconfig
-a
```

表示される各インタフェースのステータス行を確認します。たとえば次のように出力のステータス行に UP フラグが含まれていることを確認します。

```
qfe0: flags=1000843 <UP,BROADCAST,RUNNING,MULTICAST,IPv4>
mtu 1500 index 2
```

6 (オプション) リブート後もインタフェースの構成が保持されるようにするには、次の手順を実行します。

a. 構成するインタフェースごとに、**/etc/hostname.***interface* ファイルを作成します。

たとえば、qfe0 インタフェースを追加する場合には、次のファイルを作成します。

```
# vi /etc/hostname.qfe0
```

注 – 同じインタフェース用に別のホスト名ファイルを作成する場合、そのホスト名ファイルの名前も hostname.[0–9]* の形式 (hostname.qfe0.a123など) にする必要があります。hostname.qfe0.bak や hostname.qfe0.old のような名前は無効であり、システムのブート中にスクリプトに無視されます。

また、特定のインタフェースに対応するホスト名ファイルは 1 つだけにする必要があります。/etc/hostname.qfe と /etc/hostname.qfe.a123 のように、インタフェースの代替ホスト名ファイルを有効なファイル名で作成した場合、ブートスクリプトは両方のホスト名ファイルの内容を参照することによって構成を試みるため、エラーが発生します。これらのエラーを回避するには、特定の構成で使用したくないホスト名ファイルに無効なファイル名を付けます。

b. **/etc/hostname.***interface* ファイルを編集します。

少なくとも、インタフェースの IPv4 アドレスをファイルに追加します。インタフェースの IP アドレスを指定するときには、一般的な IPv4 表記または CIDR 表記を使用できます。ネットマスクやほかの構成情報もファイルに追加できます。

注 – インタフェースに IPv6 アドレスを追加するには、188 ページの「ホストとサーバーの IPv6 インタフェース構成の変更」を参照してください。

c. **Solaris 10 11/06** 以前の **Solaris 10** リリースの場合は、新しいインタフェース用のエントリを **/etc/inet/ipnodes** ファイルに追加します。

d. 新しいインタフェース用のエントリを **/etc/inet/hosts** ファイルに追加します。

e. 再構成用ブートを実行します。

```
# reboot -- -r
```

f. **/etc/hostname.***interface* ファイルで作成したインタフェースが構成されていることを確認します。

```
# ifconfig -a
```

例については、例 6–2 を参照してください。

例 6–2 持続的なインタフェース構成の追加

この例は、qfe0 インタフェースと qfe1 インタフェースをホストに対して構成する方法を示しています。これらのインタフェースは、リブート後も保持されます。

```
# dladm show-link
eri0    type: legacy    mtu: 1500       device: eri0
qfe0    type: legacy    mtu: 1500       device: qfe0
qfe1    type: legacy    mtu: 1500       device: qfe1
qfe2    type: legacy    mtu: 1500       device: qfe2
qfe3    type: legacy    mtu: 1500       device: qfe3
bge0    type: non-vlan  mtu: 1500       device: bge0
# vi /etc/hostname.qfe0
192.168.84.3 netmask 255.255.255.0
# vi /etc/hostname.qfe1
192.168.84.72 netmask 255.255.255.0
# vi /etc/inet/hosts
# Internet host table
#
127.0.0.1       localhost
10.0.0.14       myhost
192.168.84.3       interface-2
192.168.84.72       interface-3
For Solaris 10 11/06 and earlier releases:# vi /etc/inet/ipnodes
10.0.0.14 myhost
192.168.84.3       interface-2
192.168.84.72       interface-3
```

この時点でシステムをリブートします。

```
# reboot -- -r
```

システムがブートされたら、インタフェースの構成を確認します。

```
ifconfig -a
# ifconfig -a lo0: flags=1000849 <UP,LOOPBACK,RUNNING,MULTICAST,IPv4> mtu
8232 index 1
         inet 127.0.0.1 netmask ff000000
eri0: flags=1000843<UP,BROADCAST,RUNNING,MULTICAST,IPv4> mtu 1500 index 2
         inet 10.0.0.14netmask ff000000 broadcast 10.255.255.255
         ether 8:0:20:c1:8b:c3
qfe0:flags=1000843 <UP,BROADCAST,RUNNING,MULTICAST,IPv4> mtu 1500 index 3
```

```
        inet 192.168.84.3 netmask ffffff00 broadcast 192.255.255.255
        ether 8:0:20:c8:f4:1d
qfe1: flags=1000843 <UP,BROADCAST,RUNNING,MULTICAST,IPv4>mtu 1500 index 4
          inet 192.168.84.72 netmask ffffff00 broadcast 10.255.255.255
         ether 8:0:20:c8:f4:1e
```

参照
- IPv6 アドレスをインタフェースに構成する場合は、179 ページの「現在のセッションの IPv6 インタフェースを有効にする方法」を参照してください。
- IP ネットワークマルチパス (IP Network Multipathing、IPMP) を使用するインタフェースに対してフェイルオーバー検出とフェイルバックを設定する場合は、第 28 章「IPMP の管理 (タスク)」を参照してください。

▼ 物理インタフェースを削除する方法

1. 削除するインタフェースがあるシステムで、**Primary Administrator** の役割を引き受けるか、スーパーユーザーになります。

   Primary Administrator 役割には、Primary Administrator プロファイルが含まれます。役割を作成してユーザーに役割を割り当てるには、『Oracle Solaris の管理: 基本管理』の第 2 章「Solaris 管理コンソールの操作 (タスク)」を参照してください。

2. 物理インタフェースを削除します。

   # ifconfig *interface* **down unplumb**

   たとえば、`qfe1` インタフェースを削除するには、次のように入力します。

   ```
   # ifconfig qfe1 down unplumb
   ```

▼ SPARC: インタフェースの MAC アドレスが一意であることを確認する方法

MAC アドレスを構成するには、次の手順に従います。

アプリケーションによっては、ホスト上のすべてのインタフェースでそれぞれ一意の MAC アドレスが使用されている必要があります。ただし、すべての SPARC ベースの システムは、システム共通 MAC アドレスを持っており、デフォルトではすべてのインタフェースがこのアドレスを使用します。次の 2 つの状況では、SPARC システム上のインタフェースに出荷時に設定された MAC アドレスを構成する場合があります。

- リンク集約の場合、集約構成では出荷時に設定されたインタフェースの MAC アドレスを使用してください。

- IPMP グループの場合、グループ内の各インタフェースで一意の MAC アドレスを使用する必要があります。これらのインタフェースでは出荷時に設定された MAC アドレスを使用する必要があります。

EEPROM パラメータ `local-mac-address?` によって、SPARC システム上のすべてのインタフェースがシステム共通 MAC アドレスまたは一意の MAC アドレスのどちらを使用しているかを判別します。次の手順では、`eeprom` コマンドを使用して、`local-mac-address?` の現在値をチェックし、必要に応じて変更する方法を示します。

1 インタフェースがあるシステムで、**Primary Administrator** の役割を引き受けるか、スーパーユーザーになります。

Primary Administrator 役割には、Primary Administrator プロファイルが含まれます。役割を作成してユーザーに役割を割り当てるには、『Oracle Solaris の管理: 基本管理』の第 2 章「Solaris 管理コンソールの操作 (タスク)」を参照してください。

2 システム上のすべてのインタフェースがシステム共通 **MAC** アドレスを現在使用しているかどうかを判断します。

```
# eeprom local-mac-address?
local-mac-address?=false
```

この例では、`eeprom` コマンドの応答の `local-mac-address?=false` によって、すべてのインタフェースがシステム共通 MAC アドレスを使用していることが示されています。`local-mac-address?=false` の値は、インタフェースを IPMP グループのメンバーにする前に、`local-mac-address?=true` に変更する必要があります。集約の場合にも、`local-mac-address?=false` を `local-mac-address?=true` に変更してください。

3 必要に応じて、**`local-mac-address?`** の値を次のように変更します。

```
# eeprom local-mac-address?=true
```

システムをリブートすると、出荷時に設定された MAC アドレスを持つインタフェースは、システム共通 MAC アドレスの代わりに、その出荷時の設定を使用します。出荷時に設定された MAC アドレスを持たないインタフェースは、システム共通 MAC アドレスを引き続き使用します。

4 システム上のすべてのインタフェースの **MAC** アドレスをチェックします。

複数のインタフェースが 同じ MAC アドレスを持つ場合がないかどうかを調べてください。この例では、すべてのインタフェースがシステム共通 MAC アドレス `8:0:20:0:0:1` を使用しています。

```
ifconfig -a
lo0: flags=1000849 <UP,LOOPBACK,RUNNING,MULTICAST,IPv4> mtu 8232 index 1
      inet 127.0.0.1 netmask ff000000
hme0: flags=1004843 <UP,BROADCAST,RUNNING,MULTICAST,IPv4> mtu 1500 index 2
      inet 10.0.0.112 netmask ffffff80 broadcast 10.0.0.127
      ether 8:0:20:0:0:1
ce0: flags=1004843 <UP,BROADCAST,RUNNING,MULTICAST,IPv4> mtu 1500 index 2
      inet 10.0.0.114 netmask ffffff80 broadcast 10.0.0.127
```

```
      ether 8:0:20:0:0:1
ce1: flags=1004843 <UP,BROADCAST,RUNNING,MULTICAST,IPv4> mtu 1500 index 2
      inet 10.0.0.118 netmask ffffff80 broadcast 10.0.0.127
      ether 8:0:20:0:0:1
```

注 – 同じ MAC アドレスを持つ複数のネットワークインタフェースがまだ残っている場合だけ、次の手順を続けます。それ以外の場合は、最後の手順に進んでください。

---

5 すべてのインタフェースが一意の **MAC** アドレスを持つように、必要に応じて、残りのインタフェースを手動で構成します。

`/etc/hostname.`*interface* ファイル内で、特定のインタフェースに対して一意の MAC アドレスを指定します。

前の手順の例では、`ce0` と `ce1` をローカルで管理されている MAC アドレスで構成する必要がありました。たとえば、ローカルで管理されている MAC アドレス `06:05:04:03:02` で `ce1` を再構成するには、次の行を `/etc/hostname.ce1` に追加します。

```
ether 06:05:04:03:02
```

注 – 手動で構成した MAC アドレスがネットワークのほかの MAC アドレスと衝突する危険を避けるために、「ローカルで管理される」MAC アドレスは、必ず IEEE 802.3 標準の定義に従って構成してください。

`ifconfig ether` コマンドを使用して、現在のセッションに対してインタフェースの MAC アドレスを構成することもできます。ただし、リブート後は `ifconfig` で直接行われた変更の内容は失われます。詳細は `ifconfig(1M)` のマニュアルページを参照してください。

6 システムをリブートします。

仮想ローカルエリアネットワークの管理

「仮想ローカルエリアネットワーク *(VLAN)*」は、ローカルエリアネットワークを TCP/IP プロトコルスタックのデータリンク層で分割したものです。スイッチテクノロジを使用するローカルエリアネットワークの VLAN を作成できます。ユーザーのグループを VLAN に割り当てることで、ローカルネットワーク全体のネットワーク管理とセキュリティーを改善できます。さらに、同じシステム上のインタフェースを異なる VLAN に割り当てることもできます。

次の作業を行う必要がある場合は、ローカルネットワークを VLAN に分割することを検討してください。

- 作業グループの論理的な分割を作成する。

  たとえば、ある建物の 1 つの階に置かれたすべてのホストが 1 つのスイッチベースのローカルネットワークに接続されているとします。この階の各作業グループ用に個別の VLAN を作成できます。
- 作業グループに異なるセキュリティーポリシーを適用する。

  たとえば、財務部門と情報技術部門のセキュリティーニーズはまったく異なります。両方の部門のシステムが同じローカルネットワークを共有している場合、各部門用の個別の VLAN を作成できます。そのあとで、適切なセキュリティーポリシーを VLAN ごとに適用できます。
- 作業グループを管理可能なブロードキャストドメインに分割する。

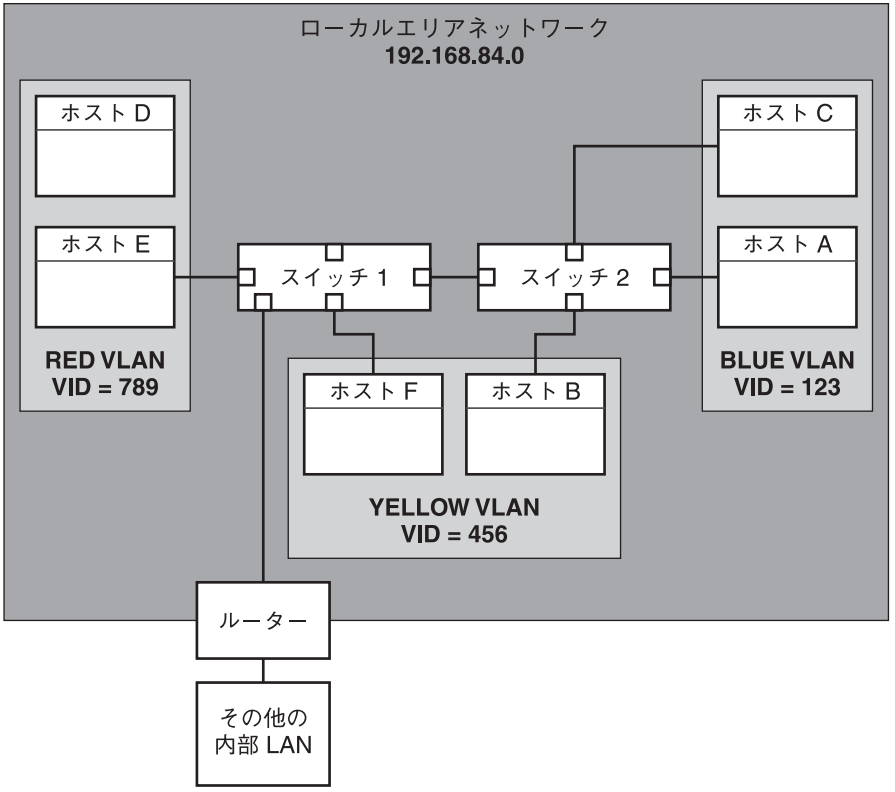
  VLAN を使用すると、ブロードキャストドメインのサイズが小さくなり、ネットワークの効率が向上します。

VLAN トポロジの概要

スイッチ LAN テクノロジを使用すると、ローカルネットワーク上のシステムを VLAN に編成できます。ローカルネットワークを VLAN に分割する前に、VLAN テクノロジをサポートするスイッチを入手する必要があります。VLAN トポロジの設計に応じて、スイッチ上のすべてのポートで単一の VLAN を処理するか、複数の VLAN を処理するように構成できます。スイッチのポートを構成する手順はスイッチの製造元によって異なります。

次の図は、サブネットアドレス `192.168.84.0` を使用するローカルエリアネットワークを示しています。この LAN は、RED、YELLOW、および BLUE という 3 つの VLAN に分割されています。

図 6–1　3 つの VLAN を含むローカルエリアネットワーク



LAN 192.168.84.0 の接続は、スイッチ 1 とスイッチ 2 によって処理されます。財務作業グループのシステムは RED VLAN に含まれています。人事作業グループのシステムは YELLOW VLAN 上にあります。情報技術作業グループのシステムは BLUE VLAN に割り当てられています。

VLAN タグと物理接続点

ローカルエリアネットワーク内の各 VLAN は、VLAN タグ、つまり「*VID (VLAN ID)*」によって識別されます。VID は、VLAN の構成時に割り当てられます。VID は、1 から 4094 までの 12 ビットの識別子で、各 VLAN を一意に識別します。図 6–1 では、BLUE VLAN が VID 123、YELLOW VLAN が VID 456、RED VLAN が VID 789 をそれぞれ持ちます。

VLAN をサポートするスイッチを構成する場合は、各ポートに VID を割り当てる必要があります。次の図に示すように、ポートに割り当てる VID は、ポートに接続されるインタフェースに割り当てる VID と同じにする必要があります。

図 6–2　VLAN を使用するネットワークのスイッチの構成

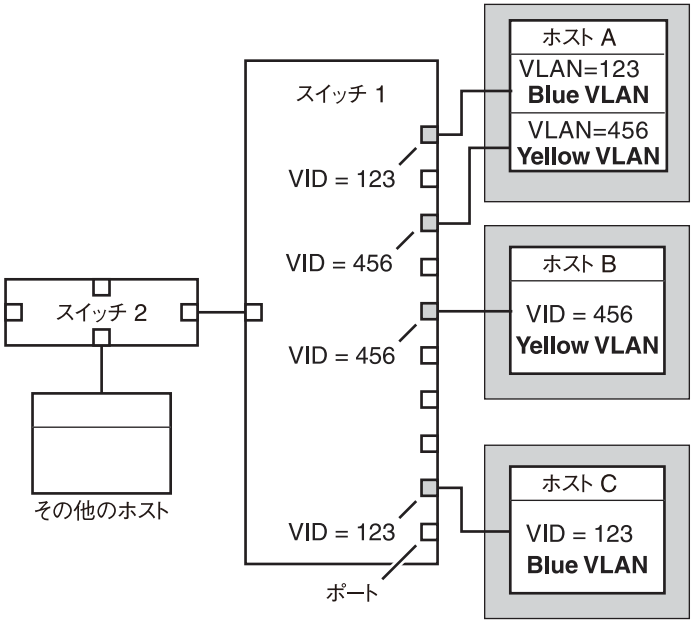


図 6–2 は、異なる VLAN に接続された複数のホストを示しています。同じ VLAN に 2 つのホストが属しています。この図では、3 つのホストのプライマリネットワークインタフェースがスイッチ 1 に接続されています。ホスト A は BLUE VLAN のメンバーです。そのため、ホスト A のインタフェースは VID 123 で構成されています。このインタフェースはスイッチ 1 のポート 1 に接続され、このポートは VID 123 で構成されています。ホストB は YELLOW VLAN のメンバーで、VID 456 で構成されています。ホストB のインタフェースはスイッチ 5 のポート 1 に接続され、このポートは VID 456 で構成されています。最後に、ホスト C のインタフェースはスイッチ 1 のポート 9 に接続されています。BLUE VLAN は VID 123 で構成されています。

また、この図で示されているとおり、1 つのホストが複数の VLAN に属すこともできます。たとえば、ホスト A にはホストのインタフェースを通じて 2 つの VLAN が構成されています。2 番目の VLAN は VID 456 で構成され、ポート 3 に接続されています。このポートも VID 456 で構成されています。したがって、ホスト A は Blue VLAN と Yellow VLAN の両方のメンバーになっています。

VLAN の構成時に、VLAN の「物理接続点 *(PPA、physical point of attachment)*」を指定する必要があります。PPA 値を取得するには、次の数式を使用します。

driver-name `+ VID * 1000 +` *device-instance*

device-instance の数値は 1000 未満である必要があります。

たとえば、次のような VLAN 456 の一部として構成される ce1 インタフェースの PPA を作成します。

```
ce + 456 * 1000 + 1= ce456001
```

ネットワーク上の VLAN の計画

ネットワーク上の VLAN を計画するには、次の手順に従います。

▼ VLAN 構成を計画する方法

1. ローカルネットワークのトポロジを調べて、**VLAN** への分割が適切かどうかを判断します。
   このようなトポロジの基本的な例については、図 6–1 を参照してください。

2. **VID** の番号指定スキームを作成し、各 **VLAN** に **VID** を割り当てます。

   注 – VLAN の番号指定スキームは、ネットワーク上にすでに存在している場合があります。その場合は、既存の VLAN 番号指定スキームに従って VID を作成する必要があります。

   ---

3. 各システム上で、特定の **VLAN** のメンバーにするインタフェースを決定します。

   a. システム上で構成されているインタフェースを調べます。

      ```
      # dladm show-link
      ```

   b. システム上の各データリンクに関連付ける **VID** を判別します。

   c. **VLAN** で構成する各インタフェースの **PPA** を作成します。

   システムのすべてのインタフェースを同じ VLAN 上で構成する必要はありません。

4. インタフェースとネットワークのスイッチの接続を確認します。
   各インタフェースと各インタフェースが接続されているスイッチポートの VID を書き留めます。

5. スイッチの各ポートの **VID** をポートが接続されるインタフェースと同じ **VID** に構成します。
   構成手順については、スイッチの製造元のドキュメントを参照してください。

VLAN の構成

Oracle Solaris は現在、次のタイプのインタフェース上で VLAN をサポートします。

- ce
- bge
- xge
- e1000g

旧式インタフェースタイプのうち、ce インタフェースのみが VLAN のメンバーになることができます。同じ VLAN 内に異なるタイプのインタフェースを構成できます。

注 – 1 つの IPMP グループ内に複数の VLAN を構成できます。IPMP グループについての詳細は、727 ページの「IPMP インタフェースの構成」を参照してください。

▼ VLAN を構成する方法

1 **Primary Administrator 役割を引き受けるか、スーパーユーザーになります。**

Primary Administrator 役割には、Primary Administrator プロファイルが含まれます。役割を作成してユーザーに役割を割り当てるには、『Oracle Solaris の管理: 基本管理』の第 2 章「Solaris 管理コンソールの操作 (タスク)」を参照してください。

2 システムでどのようなタイプのインタフェースが使用されているか判別します。

```
# dladm show-link
```

使用可能なインタフェースのタイプが出力されます。

```
ce0             type: legacy    mtu: 1500       device: ce0
 ce1             type: legacy    mtu: 1500       device: ce1
 bge0            type: non-vlan  mtu: 1500       device: bge0
 bge1            type: non-vlan  mtu: 1500       device: bge1
 bge2            type: non-vlan  mtu: 1500       device: bge2
```

3 **VLAN の一部としてインタフェースを構成します。**

```
# ifconfig interface-PPA plumb IP-address up
```

たとえば次のコマンドを使用して、新しい IP アドレス 10.0.0.2 のインタフェース ce1 を、VID 123 の VLAN 内に構成します。

```
# ifconfig ce123001 plumb 10.0.0.2
up
```

注 – ほかのインタフェースの場合と同様に、VLAN には IPv4 アドレスと IPv6 アドレスを割り当てることができます。

4 (オプション)**VLAN** の設定がリブート後も保持されるようにするには、**VLAN** の一部として構成される各インタフェース用に **hostname.***interface-PPA* ファイルを作成します。

```
# cat hostname.interface-PPA
IPv4-address
```

5 スイッチで、**VLAN** のタグ付けと **VLAN** ポートを、システムに設定した **VLAN** と一致するように設定します。

例 6–3 VLAN の構成

この例は、デバイス bge1 と bge2 を VID 123 で VLAN 内に構成する方法を示しています。

```
# dladm show-link
ce0             type: legacy    mtu: 1500       device: ce0
ce1             type: legacy    mtu: 1500       device: ce1
bge0            type: non-vlan  mtu: 1500       device: bge0
bge1            type: non-vlan  mtu: 1500       device: bge1
bge2            type: non-vlan  mtu: 1500       device: bge2
# ifconfig bge123001 plumb 10.0.0.1 up
# ifconfig bge123002 plumb 10.0.0.2 up
# cat hostname.bge123001   10.0.0.1
# cat hostname.bge123002   10.0.0.2
# ifconfig -a
 lo0: flags=2001000849 <UP,LOOPBACK,RUNNING,MULTICAST,IPv4,VIRTUAL> mtu 8232 index 1
         inet 127.0.0.1 netmask ff000000
 bge123001: flags=201000803<UP,BROADCAST,MULTICAST,IPv4,CoS> mtu 1500 index 2
         inet 10.0.0.1 netmask ff000000 broadcast 10.255.255.255
         ether 0:3:ba:7:84:5e
bge123002:flags=201000803 <UP,BROADCAST,MULTICAST,IPv4,CoS> mtu 1500 index 3
         inet 10.0.0.2 netmask ff000000 broadcast 10.255.255.255
         ether 0:3:ba:7:84:5e
ce0: flags=1000843 <UP,BROADCAST,RUNNING,MULTICAST,IPv4>mtu 1500 index 4
         inet 192.168.84.253 netmask ffffff00 broadcast 192.168.84.255
         ether 0:3:ba:7:84:5e
# dladm show-link
ce0             type: legacy    mtu: 1500       device: ce0
ce1             type: legacy    mtu: 1500       device: ce1
bge0            type: non-vlan  mtu: 1500       device: bge0
bge1            type: non-vlan  mtu: 1500       device: bge1
bge2            type: non-vlan  mtu: 1500       device: bge2
bge123001       type: vlan 123  mtu: 1500       device: bge1
bge123002       type: vlan 123  mtu: 1500       device: bge2
```

リンクアグリゲーションの概要

注 – オリジナルの Solaris 10 リリースと以前のバージョンの Solaris OS はリンクアグリゲーションをサポートしていません。これらの以前の Solaris リリースでリンクアグリゲーションを作成するには、『Sun Trunking 1.3 Installation and Users Guide』で説明されているように、Sun Trunking を使用します。

Oracle Solaris では、リンク集約へのネットワークインタフェースの編成をサポートしています。「リンク集約」は、単一の論理的なユニットとして構成されるシステム上の複数のインタフェースで構成されます。リンク集約は「*Trunking*」とも呼ばれ、IEEE 802.3ad Link Aggregation Standard (`http://www.ieee802.org/3/index.html`) で定義されています。

IEEE 802.3ad Link Aggregation Standard には、複数の全二重 Ethernet リンクの伝送容量を単一の論理リンクに統合する方法が記載されています。このリンク集約グループは、事実上単一のリンクであるかのように扱われます。

次にリンク集約の機能を示します。

- 帯域幅の増加 – 複数のリンクの伝送容量が 1 つの論理的なリンクに統合されます。
- 自動フェイルオーバーまたは自動フェイルバック – 障害が発生したリンクのトラフィックが集約内の正常なリンクにフェイルオーバーされます。
- 負荷分散 – 受信と送信の両方のトラフィックが、送信元と送信先の MAC アドレスまたは IP アドレスなどのユーザーが選択した負荷分散ポリシーに従って分散されます。
- 冗長性のサポート – 並列集約を使用して 2 つのシステムを構成できます。
- 管理効率の向上 – すべてのインタフェースが単一のユニットとして管理されます。
- ネットワークアドレスプールのアドレスの節約 – 集約全体に 1 つの IP アドレスを割り当てることができます。

リンク集約の基本

基本的なリンク集約のトポロジには、一連の物理インタフェースで構成された単一の集約が含まれます。基本的なリンク集約は、次のような状況で使用します。

- 分散された多くのトラフィックを処理するアプリケーションを実行するシステムの場合、集約をそのアプリケーションのトラフィック専用で使用できます。
- IP アドレス空間が制限されていながら大容量の帯域幅が必要なサイトの場合、大規模なインタフェースの集約でも 1 つの IP アドレスのみで済みます。

- 内部インタフェースの存在を隠す必要があるサイトの場合、集約の IP アドレスによって、内部インタフェースを外部アプリケーションから隠します。

図 6–3 に、有名な Web サイトをホストするサーバーの集約を示します。このサイトでは、インターネット顧客とサイトのデータベースサーバーの間の照会トラフィックのために帯域幅を増やす必要があります。セキュリティー上の理由で、サーバー上の各インタフェースの存在を外部アプリケーションから隠す必要があります。解決策として、 IP アドレス `192.168.50.32` で集約 `aggr1` を使用します。この集約は、`bge0` - `bge2` の 3 つのインタフェースで構成されます。これらのインタフェースは、顧客の照会に答えるためのトラフィックの送信専用で使用されます。すべてのインタフェースからのパケットトラフィック上の送信アドレスは、`aggr1` の IP アドレスである `192.168.50.32` です。

図 6–3 基本的なリンク集約トポロジ

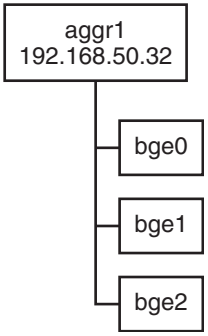
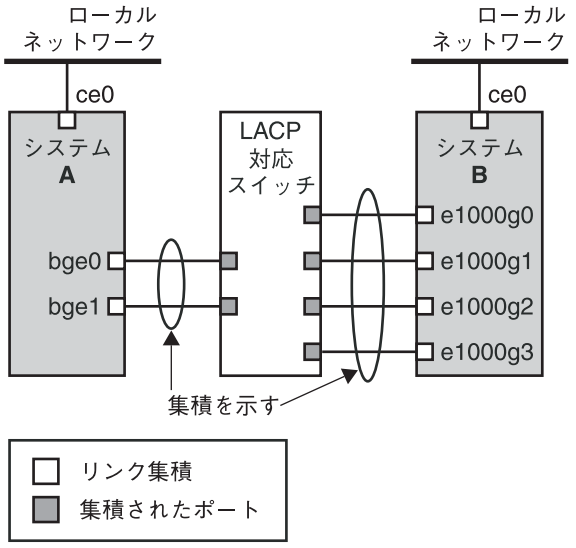


図 6–4 は、それぞれに集約が構成された 2 つのシステムを含むローカルネットワークを示しています。2 つのシステムはスイッチによって接続されています。スイッチ経由で集約を実行する必要がある場合は、そのスイッチが集約テクノロジをサポートしている必要があります。このタイプの構成は、高可用性と冗長性を持つシステムを実現するために特に有効です。

図では、システム A が `bge0` と `bge1` という 2 つのインタフェースで構成される集約を使用しています。これらのインタフェースは、集約に入れられたポートを介してスイッチに接続されています。システム B は、`e1000g0` - `e1000g3` という 4 つのインタフェースの集約を使用しています。これらのインタフェースもスイッチの集約に入れられたポートに接続されています。

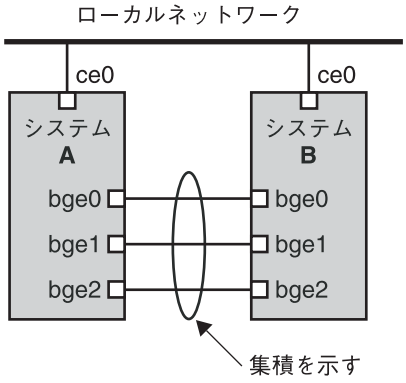
図 6–4 スイッチを使用したリンク集約のトポロジ



バックツーバックリンク集約

バックツーバックリンク集約のトポロジには、次の図に示すように、相互に直接ケーブル接続された 2 台の別個のシステムが含まれます。これらのシステムでは並列集約が実行されます。

図 6–5 基本的なバックツーバック集約のトポロジ



この図では、システム A 上のデバイス `bge0` が、システム B 上の `bge0` に直接リンクされ、ほかのデバイスも同様にリンクされています。この方法では、システム A とシステム B は、冗長性と高可用性を提供し、さらに両方のシステム間での高速通信を

サポートできます。各システムではさらに、ローカルネットワーク内のトラフィックフロー用の `ce0` インタフェースも構成されています。

バックツーバックリンク集約のもっとも一般的なアプリケーションはミラー化されたデータベースサーバーです。両方のサーバーを同時に更新する必要があるため、大きな帯域幅、高速のトラフィックフロー、および信頼性が必要になります。バックツーバックリンク集約のもっとも一般的な使用場所としてデータセンターがあります。

ポリシーと負荷分散

リンク集約を使用する予定の場合は、送信トラフィック用のポリシーを定義することを検討してください。このポリシーでは、使用可能な集約のリンク全体にパケットを分散する方法を指定し、負荷分散を確立することができます。次に、使用可能な層指定子と集約ポリシーに対するそれらの意味について説明します。

- **L2** – 各パケットの MAC (L2) ヘッダーをハッシュすることで送信リンクを決定します
- **L3** – 各パケットの IP (L3) ヘッダーをハッシュすることで送信リンクを決定します
- **L4** – 各パケットの TCP、UDP、またはほかの ULP (L4) ヘッダーをハッシュすることで送信リンクを決定します

これらのポリシーを任意に組み合わせて使用することもできます。デフォルトのポリシーは L4 です。詳細は、`dladm(1M)` のマニュアルページを参照してください。

集約モードとスイッチ

集約トポロジにスイッチ経由の接続が含まれている場合は、スイッチが「*LACP (Link Aggregation Control Protocol)*」をサポートするかどうかに注意する必要があります。スイッチが LACP をサポートしている場合は、スイッチと集約の LACP を構成する必要があります。ただし、次のいずれかの LACP の動作「モード」を定義できます。

- オフモード – 集約のデフォルトのモード。「*LACPDU*」と呼ばれる LACP パケットは生成されません。
- アクティブモード – ユーザーが指定可能な間隔でシステムによって LACPDU が定期的に生成されます。
- 受動モード – システムは、スイッチから LACPDU を受け取った場合のみ LACPDU を生成します。集約とスイッチの両方が受動モードで構成されている場合、それらの間で LACPDU を交換することはできません。

構文については、`dladm(1M)` のマニュアルページとスイッチの製造元のマニュアルを参照してください。

リンク集約の要件

リンク集約の構成には次のような要件があります。

- `dladm` コマンドを使用して集約を構成する必要があります。
- すでに plumb されているインタフェースを集約のメンバーにすることはできません。
- インタフェースは、次の GLDv3 タイプである必要があります。 `xge`、`e1000g`、および `bge` です。
- 集約内のすべてのインタフェースは、同じ速度で全二重モードで実行されている必要があります。
- EEPROM パラメータ `local-mac-address?` 内で、MAC アドレスの値を「true」に設定する必要があります。手順の詳細は、156 ページの「SPARC: インタフェースの MAC アドレスが一意であることを確認する方法」を参照してください。

▼ リンクアグリゲーションを作成する方法

始める前に

注 – リンクアグリゲーションは、同一の速度で稼働する全二重のポイントツーポイントリンク上でのみ機能します。アグリゲーション内のインタフェースがこの要件を満たしていることを確認してください。

集約トポロジ内でスイッチを使用している場合は、スイッチ上で次の操作を行なったことを確認してください。

- アグリゲーションとして使用されるようにポートを構成します。
- スイッチが LACP をサポートしている場合は、LACP をアクティブモードまたは受動モードで構成します。

1. **Primary Administrator** 役割を引き受けるか、スーパーユーザーになります。

   Primary Administrator 役割には、Primary Administrator プロファイルが含まれます。役割を作成してユーザーに役割を割り当てるには、『Oracle Solaris の管理: 基本管理』の第 2 章「Solaris 管理コンソールの操作 (タスク)」を参照してください。

2. 現在システムにインストールされているインタフェースを調べます。

   ```
   # dladm show-link
   ```

3. **plumb** されているインタフェースを判別します。

   ```
   # ifconfig -a
   ```

4. 集約を作成します。

   ```
   # dladm create-aggr -d interface -d interface [...]key
   ```

interface 集約の一部になるインタフェースのデバイス名を表します。

key 集約を識別する番号です。最小のキー番号は 1 です。0 はキーには使用できません。

次に例を示します。

```
# dladm create-aggr -d bge0 -d bge1 1
```

5 新しく作成した集約を構成して **plumb** します。

```
# ifconfig aggrkey plumb IP-address up
```

次に例を示します。

```
# ifconfig aggr1  plumb 192.168.84.14 up
```

6 作成した集約のステータスを確認します。

```
# dladm show-aggr
```

画面に次のような出力が表示されます。

```
key: 1 (0x0001) policy: L4      address: 0:3:ba:7:84:5e (auto)
device   address           speed         duplex  link    state
bge0     0:3:ba:7:b5:a7    1000  Mbps    full    up      attached
bge1     0:3:ba:8:22:3b    0     Mbps    unknown down    standby
```

この出力は、キーが 1 でポリシーが L4 の集約が作成されたことを示しています。

7 (オプション) リンク集約の **IP** 構成は、リブート後も保持されるようにします。

a. **IPv4** アドレスベースのリンク集約の場合は、**/etc/hostname.aggr.***key* ファイルを作成します。**IPv6** ベースのリンクアグリゲーションの場合は、**/etc/hostname6.aggr.***key* ファイルを作成します。

b. リンク集約の **IPv4** または **IPv6** アドレスをファイルに入力します。

たとえば、この処理で作成される集約の場合、次のファイルを作成します。

```
# vi /etc/hostname.aggr1
192.168.84.14
```

c. 再構成用ブートを実行します。

```
# reboot -- -r
```

d. **/etc/hostname.aggr***key* ファイルに入力したリンク集約構成が構成済みであることを確認します。

```
# ifconfig -a
.
.
aggr1: flags=1000843 <UP,BROADCAST,RUNNING,MULTICAST,IPv4> mtu 1500 index 3
```

```
        inet 192.168.84.14 netmask ff000000 broadcast 192.255.255.
```

例 6–4 リンク集約の作成

この例は、bge0 と bge1 という 2 つのデバイスを含むリンク集約を作成するために使用するコマンドと、その出力結果を示しています。

```
# dladm show-link
ce0             type: legacy    mtu: 1500       device: ce0
ce1             type: legacy    mtu: 1500       device: ce1
bge0            type: non-vlan  mtu: 1500       device: bge0
bge1            type: non-vlan  mtu: 1500       device: bge1
bge2            type: non-vlan  mtu: 1500       device: bge2
# ifconfig -a
lo0: flags=2001000849 <UP,LOOPBACK,RUNNING,MULTICAST,IPv4,VIRTUAL> mtu 8232 index 1
        inet 127.0.0.1 netmask ff000000
ce0: flags=1000843 <UP,BROADCAST,RUNNING,MULTICAST,IPv4> mtu 1500 index 2
        inet 192.168.84.253 netmask ffffff00 broadcast 192.168.84.255
        ether 0:3:ba:7:84:5e
# dladm create-aggr -d bge0 -d bge1 1
# ifconfig aggr1 plumb 192.168.84.14 up
# dladm show-aggr
key: 1 (0x0001) policy: L4      address: 0:3:ba:7:84:5e (auto)
device   address           speed         duplex  link    state
bge0     0:3:ba:7:b5:a7    1000  Mbps    full    up      attached
bge1     0:3:ba:8:22:3b    0     Mbps    unknown down    standby

# ifconfig -a
lo0: flags=2001000849 <UP,LOOPBACK,RUNNING,MULTICAST,IPv4,VIRTUAL> mtu 8232 index 1
        inet 127.0.0.1 netmask ff000000
ce0: flags=1000843 <UP,BROADCAST,RUNNING,MULTICAST,IPv4> mtu 1500 index 2
        inet 192.168.84.253 netmask ffffff00 broadcast 192.168.84.255
        ether 0:3:ba:7:84:5e
aggr1: flags=1000843 <UP,BROADCAST,RUNNING,MULTICAST,IPv4> mtu 1500 index 3
        inet 192.168.84.14 netmask ff000000 broadcast 192.255.255.255
        ether 0:3:ba:7:84:5e
```

集約に使用された 2 つのインタフェースは、 ifconfig によって事前に plumb されていません。

▼ 集約を変更する方法

この手順では、集約の定義に次の変更を加える方法を示します。

- 集約のポリシーの変更
- 集約のモードの変更

1 **Primary Administrator** 役割を引き受けるか、スーパーユーザーになります。

Primary Administrator 役割には、Primary Administrator プロファイルが含まれます。役割を作成してユーザーに役割を割り当てるには、『Oracle Solaris の管理: 基本管理』の第 2 章「Solaris 管理コンソールの操作 (タスク)」を参照してください。

2 集約のポリシーを変更します。

```
# dladm modify-aggr -Ppolicy key
```

policy 168 ページの「ポリシーと負荷分散」で説明されているように 1 つ以上のポリシー L2、L3、および L4 を表します。

key 集約を識別する番号です。最小のキー番号は 1 です。0 はキーには使用できません。

3 集約内のデバイスが接続されているスイッチ上で **LACP** が実行されている場合は、**LACP** をサポートするように集約を変更します。

スイッチ上で受動モードで LACP が実行されている場合は、集約用にアクティブモードに構成したことを確認してください。

```
# dladm modify-aggr -l LACP mode -t timer-value key
```

`-l` *LACP mode* 集約が実行される LACP モードを示します。値は、`active`、`passive`、および `off` です。

`-t` *timer-value* LACP タイマー値を示します。値は、`short` または `long` です。

key 集約を識別する番号です。最小のキー番号は 1 です。0 はキーには使用できません。

例 6–5 リンク集約の変更

この例は、アグリゲーション `aggr1` のポリシーを L2 に変更し、LACP モードをアクティブにする方法を示しています。

```
# dladm modify-aggr -P L2 1
# dladm modify-aggr -l active -t short 1
# dladm show-aggr
key: 1 (0x0001) policy: L2      address: 0:3:ba:7:84:5e (auto)
device   address           speed         duplex  link    state
bge0     0:3:ba:7:b5:a7    1000  Mbps    full    up      attached
bge1     0:3:ba:8:22:3b    0     Mbps    unknown down    standby
```

▼ 集約からインタフェースを削除する方法

1 **Primary Administrator** 役割を引き受けるか、スーパーユーザーになります。

Primary Administrator 役割には、Primary Administrator プロファイルが含まれます。役割を作成してユーザーに役割を割り当てるには、『Oracle Solaris の管理: 基本管理』の第 2 章「Solaris 管理コンソールの操作 (タスク)」を参照してください。

2 集約からインタフェースを削除します。

```
# dladm remove-aggr -d interface
```

例 6–6 集約からのインタフェースの削除

この例は、アグリゲーション aggr1 からインタフェースを削除する方法を示しています。

```
# dladm show-aggr
key: 1 (0x0001) policy: L2      address: 0:3:ba:7:84:5e (auto)
device   address           speed         duplex  link    state
bge0     0:3:ba:7:b5:a7    1000  Mbps    full    up      attached
bge1     0:3:ba:8:22:3b    0     Mbps    unknown down    standby
# dladm remove-aggr -d bge1 1
# dladm show-aggr
key: 1 (0x0001) policy: L2      address: 0:3:ba:7:84:5e (auto)
device   address           speed         duplex  link    state
bge0     0:3:ba:7:b5:a7    1000  Mbps    full    up      attached
```

▼ 集約を削除する方法

1 **Primary Administrator** 役割を引き受けるか、スーパーユーザーになります。

Primary Administrator 役割には、Primary Administrator プロファイルが含まれます。役割を作成してユーザーに役割を割り当てるには、『Oracle Solaris の管理: 基本管理』の第 2 章「Solaris 管理コンソールの操作 (タスク)」を参照してください。

2 アグリゲーションを削除します。

```
# dladm delete-aggr key
```

key 集約を識別する番号です。最小のキー番号は 1 です。0 はキーには使用できません。

例 6–7 集約を削除する方法

この例は、集約 aggr1 を削除する方法を示しています。

```
# dladm show-aggr
key: 1 (0x0001) policy: L2      address: 0:3:ba:7:84:5e (auto)
     device   address           speed     duplex  link    state
# dladm delete-aggr -d 1
```

▼ リンクアグリゲーション上に**VLAN**を構成する方法

インタフェース上に VLAN を構成する場合と同じ方法で、リンクアグリゲーション上に VLAN を作成することもできます。VLAN については、158 ページの「仮想ローカルエリアネットワークの管理」を参照してください。このセクションでは、VLAN とリンクアグリゲーションの構成について説明します。

始める前に リンクアグリゲーションを作成します。集約の key の値を書き留めます。この値は、集約上に VLAN を作成する際に必要になります。リンクアグリゲーションを作成する場合は、169 ページの「リンクアグリゲーションを作成する方法」を参照してください。

1. リンク集約がすでに作成されている場合は、その集約の鍵を取得します。

   ```
   # dladm show-aggr
   ```

2. リンク集約上に**VLAN**を作成します。

   ```
   # ifconfig aggrVIDkey plumb
   ```

   次に、各引数について説明します。

   VID VLAN の ID

   key VLAN を作成するリンク集約の鍵。鍵は 3 桁の形式でなければいけません。たとえば、集約の鍵が 1 の場合、VLAN の名前に含まれる鍵の番号は `001` となります。

3. 手順 **2** を繰り返して、集約上にほかの**VLAN**も作成します。

4. 有効な**IP**アドレスを使用して**VLAN**を構成します。

5. 持続的な**VLAN**構成を作成するには、対応する **/etc/hostname.***VLAN* 構成ファイルに**IP**アドレス情報を追加します。

例 **6–8** リンク集約上に複数の VLAN を構成する

この例では、リンク集約上に 2 つの VLAN を構成します。`dladm show-aggr` コマンドの出力で、リンク集約の鍵は 1 であることが示されています。VLAN には VID 193 と 194 がそれぞれ割り当てられます。

```
# dladm show-aggr
key: 1 (0x0001) policy: L4      address: 0:3:ba:7:84:5e (auto)
device   address           speed         duplex  link    state
bge0     0:3:ba:7:b5:a7    1000  Mbps    full    up      attached
bge1     0:3:ba:8:22:3b    0     Mbps    unknown down    standby

# ifconfig aggr193001 plumb
# ifconfig aggr193001 192.168.10.0/24 up

# ifconfig aggr194001 plumb
# ifconfig aggr194001 192.168.20.0/24 up

# vi /etc/hostname.aggr193001
192.168.10.0/24

# vi /etc/hostname.aggr194001
192.168.20.0/24
```

◆ ◆ ◆ 第 7 章

IPv6 ネットワークの構成 (手順)

この章では、IPv6 をネットワークに構成する作業について説明します。この章で説明する内容は次のとおりです。

- 177 ページの「IPv6 インタフェースの構成」
- 178 ページの「IPv6 をインタフェース上で有効にする方法 (タスクマップ)」
- 183 ページの「IPv6 ルーターの構成」
- 188 ページの「ホストとサーバーの IPv6 インタフェース構成の変更」
- 188 ページの「IPv6 インタフェース構成の変更 (タスクマップ)」
- 196 ページの「IPv6 サポート用のトンネルの構成」
- 195 ページの「IPv6 サポート用にトンネルを構成するためのタスク (タスクマップ)」
- 205 ページの「ネームサービスの IPv6 サポート用の構成」

IPv6 に関するさまざまな種類の情報については、次のリソースを参照してください。

- IPv6 の概念の概要: 第 3 章「IPv6 の紹介 (概要)」
- IPv6 の計画タスク: 第 4 章「IPv6 ネットワークの計画 (手順)」
- IP トンネルを使用する準備: 94 ページの「ネットワークトポロジにおけるトンネルの計画」
- 参照情報: Chapter 11, IPv6 の詳細 (リファレンス)

IPv6 インタフェースの構成

ネットワーク上で IPv6 を使用するための最初の手順として、システムの IP インタフェースで IPv6 を構成します。

Oracle Solaris インストール時に、1 つまたは複数のインタフェース上で IPv6 を有効にすることができます。インストール時に IPv6 サポートを有効にした場合、インストール完了後に次の IPv6 関連のファイルやテーブルが存在しています。

- IPv6 を有効にしたインタフェースごとに、`/etc/hostname6.`*interface* ファイル (たとえば、`hostname6.dmfe0`) が関連付けられます。
- Solaris 10 11/06 以前のリリースの場合は、`/etc/inet/ipnodes` ファイルが作成されています。インストール後、このファイルには通常、IPv6 ループバックアドレスおよび IPv4 ループバックアドレスだけが含まれます。
- `/etc/nsswitch.conf` ファイルは、IPv6 アドレスを使用して検索できるように変更されます。
- IPv6 アドレスを使用した検索が行えるように、`name-service/switch` SMF サービスが変更されています。
- IPv6 アドレス選択ポリシーテーブルが作成されます。このテーブルは、IPv6 が有効なインタフェース経由の転送で使用される IP アドレス書式に優先順位を付けます。

このセクションでは、Oracle Solaris のインストール完了後にインタフェース上で IPv6 を有効にする方法について説明します。

IPv6 をインタフェース上で有効にする方法 (タスクマップ)

次の表に、IPv6 インタフェースを構成するための各種タスクの一覧を示します。表では、各タスクで実行する内容の説明と、タスクの具体的な実行手順が詳しく説明されている現在のマニュアル内のセクションを示しています。

タスク	説明	参照先
Oracle Solaris でインストールされているシステムのインタフェース上で IPv6 を有効にします。	このタスクでは、Oracle Solaris のインストール後に、インタフェースの IPv6 を有効にします。	179 ページの「現在のセッションの IPv6 インタフェースを有効にする方法」
IPv6 が有効なインタフェースがリブート後も保持されるようにします。	このタスクでは、インタフェースの IPv6 アドレスを持続する設定にします。	180 ページの「持続する IPv6 インタフェースを有効にする方法」
IPv6 アドレスの自動構成を無効にします。	このタスクは、IPv6 アドレスのインタフェース ID 部分を手動で構成する必要がある場合に使用します。	183 ページの「IPv6 アドレスの自動構成を無効にする方法」

▼ 現在のセッションの IPv6 インタフェースを有効にする方法

IPv6 を構成する手順は、IPv6 ノードになるすべてのシステムインタフェースで IPv6 を有効にすることから始まります。それらのインタフェースは最初に、自動構成プロセスによって IPv6 アドレスを取得します (84 ページの「IPv6 アドレスの自動構成」を参照)。それらのノードの構成は、IPv6 ネットワーク上の機能 (ホスト、サーバー、またはルーター) に基づいて調整できます。

注 – インタフェースと同じリンク上に IPv6 接頭辞を現在通知しているルーターが存在する場合、そのインタフェースは自動構成アドレスの一部としてそのサイトの接頭辞を取得します。詳細については、184 ページの「IPv6 対応のルーターを構成する方法」を参照してください。

次の手順では、Oracle Solaris のインストール後に追加されたインタフェースで IPv6 を有効にする方法について説明します。

始める前に IPv6 ネットワークの計画作業を完了します。たとえば、ハードウェアとソフトウェアのアップグレードや、アドレス指定計画の準備などです。詳細は、87 ページの「IPv6 の計画 (タスクマップ)」を参照してください。

1. **IPv6 ノードになるノードに Primary Administrator またはスーパーユーザーとしてログインします。**

   Primary Administrator 役割には、Primary Administrator プロファイルが含まれます。役割を作成してユーザーに役割を割り当てるには、『Oracle Solaris の管理: 基本管理』の第 2 章「Solaris 管理コンソールの操作 (タスク)」を参照してください。

2. **インタフェースの IPv6 を有効にします。**

   ```
   # ifconfig interface inet6 plumb up
   ```

3. **IPv6 デーモン `in.ndpd` を起動します。**

   ```
   # /usr/lib/inet/in.ndpd
   ```

注 – 特定のノード上で、IPv6 が有効なインタフェースのステータスは、`ifconfig -a6` コマンドを使用して表示できます。

例 7–1 インストール後に IPv6 インタフェースを有効にする方法

この例では、`qfe0` インタフェースの IPv6 を有効にする方法を示します。作業を始める前に、システムに構成されているすべてのインタフェースのステータスを確認します。

```
# ifconfig -a
lo0: flags=1000849 <UP,LOOPBACK,RUNNING,MULTICAST,IPv4> mtu 8232 index 1
        inet 127.0.0.1 netmask ff000000
qfe0: flags=1000863 <UP,BROADCAST,NOTRAILERS,RUNNING,MULTICAST,IPv4> mtu 1500
           index 2
        inet 172.16.27.74 netmask ffffff00 broadcast 172.16.27.255
        ether 0:3:ba:13:14:e1
```

このシステムに現在構成されているインタフェースは、`qfe0` だけです。このインタフェースの IPv6 を次のように有効にします。

```
# ifconfig qfe0 inet6 plumb up
# /usr/lib/inet/in.ndpd
# ifconfig -a6
lo0: flags=2000849 <UP,LOOPBACK,RUNNING,MULTICAST,IPv6> mtu 8252 index 1
        inet6 ::1/128
qfe0: flags=2000841 <UP,RUNNING,MULTICAST,IPv6> mtu 1500 index 2
        ether 0:3:ba:13:14:e1
        inet6 fe80::203:baff:fe13:14e1/10
```

この例では、`qfe0` が、IPv6 が有効になる前後のシステムインタフェースのステータスを表示しています。`ifconfig` に `-a6` オプションを指定すると、`qfe0` とループバックインタフェースの IPv6 情報だけが表示されます。出力は、リンクのローカルアドレス (`fe80::203:baff:fe13:14e1/10`) だけが `qfe0` に構成されたことを示しています。このアドレスは、ノードのローカルリンク上のルーターが現時点ではまだサイト接頭辞を通知していないことを示しています。

IPv6 を有効にしたあとに `ifconfig -a` コマンドを使用すると、システム上のすべてのインタフェースの IPv4 アドレスと IPv6 アドレスを表示できます。

次の手順

- IPv6 ノードをルーターとして構成する方法については、183 ページの「IPv6 ルーターの構成」を参照してください。
- リブート後も IPv6 インタフェース構成を保持する方法については、180 ページの「持続する IPv6 インタフェースを有効にする方法」を参照してください。
- ノード上でのアドレスの自動構成を無効にする方法については、183 ページの「IPv6 アドレスの自動構成を無効にする方法」を参照してください。
- ノードをサーバーとして調整する方法については、194 ページの「サーバー上での IPv6 が有効なインタフェースの管理」を参照してください。

▼ 持続する IPv6 インタフェースを有効にする方法

この手順では、IPv6 インタフェースを有効にするときに自動的に構成した IPv6 アドレスが、リブート後も保持されるように 設定する方法について説明します。

注 – インタフェースと同じリンク上に IPv6 接頭辞を現在通知しているルーターが存在する場合、そのインタフェースは自動構成アドレスの一部としてそのサイトの接頭辞を取得します。詳細については、184 ページの「IPv6 対応のルーターを構成する方法」を参照してください。

1 **IPv6 ノードに Primary Administrator またはスーパーユーザーとしてログインします。**

Primary Administrator 役割には、Primary Administrator プロファイルが含まれます。役割を作成してユーザーに役割を割り当てるには、『Oracle Solaris の管理: 基本管理』の第 2 章「Solaris 管理コンソールの操作 (タスク)」を参照してください。

2 **インストール後に追加されたインタフェースの IPv6 アドレスを作成します。**

a. **構成ファイルを作成します。**

```
# touch /etc/hostname6.interface
```

b. **アドレスを構成ファイルに追加します。**

```
ipv6-address up
...
```

3 **静的 IPv6 デフォルトルートを作成します。**

```
# /usr/sbin/route -p add -inet6 default ipv6-address
```

4 **(オプション) ノード上でインタフェース変数のパラメータを定義する /etc/inet/ndpd.conf ファイルを作成します。**

ホストのインタフェースに一時アドレスを作成する必要がある場合は、188 ページの「インタフェースに対する一時アドレスの使用」を参照してください。/etc/inet/ndpd.conf の詳細については、ndpd.conf(4) のマニュアルページおよび272 ページの「ndpd.conf 構成ファイル」を参照してください。

5 **ノードをリブートします。**

```
# reboot -- -r
```

リブートすると、ルーター発見パケットが送信されます。ルーターがサイト接頭辞を返す場合は、ノードに対応する /etc/hostname6.*interface* ファイルに、グローバル IPv6 アドレスを持つ任意のインタフェースを構成できます。それ以外の場合、IPv6 が有効なインタフェースにリンクのローカルアドレスだけが構成されます。リブートすると、in.ndpd およびその他のネットワークデーモンも IPv6 モードで再起動します。

例 7–2 リブート後も IPv6 インタフェースが持続するように設定する

この例では、リブート後も qfe0 インタフェースの IPv6 構成が持続するように設定する方法を示します。この例では、ローカルリンク上のルーターから、サイト接頭辞とサブネット ID 2001:db8:3c4d:15/64 が通知されます。

最初に、システムのインタフェースのステータスを確認します。

```
# ifconfig -a
lo0: flags=1000849 <UP,LOOPBACK,RUNNING,MULTICAST,IPv4> mtu 8232 index 1
        inet 127.0.0.1 netmask ff000000
qfe0: flags=1000863 <UP,BROADCAST,NOTRAILERS,RUNNING,MULTICAST,IPv4> mtu 1500
           index 2
        inet 172.16.27.74 netmask ffffff00 broadcast 172.16.27.255
        ether 0:3:ba:13:14:e1

# touch /etc/hostname6.qfe0
# vi /etc/hostname6.qfe0
inet6 fe80::203:baff:fe13:1431/10 up
addif 2001:db8:3c4d:15:203:baff:fe13:14e1/64 up

# route -p add -inet6 default fe80::203:baff:fe13:1431
# reboot -- -r
```

構成した IPv6 アドレスがまだ qfe0 インタフェースに適用されていることを確認します。

```
# ifconfig -a6
qfe0: flags=2000841 <UP,RUNNING,MULTICAST,IPv6> mtu 1500 index 2
       ether 0:3:ba:13:14:e1
       inet6 fe80::203:baff:fe13:14e1/10
 qfe0:1: flags=2180841 <UP,RUNNING,MULTICAST,ADDRCONF,IPv6> mtu 1500
          index 2
        inet6 2001:db8:3c4d:15:203:baff:fe13:14e1/64
```

ifconfig -a6 の出力には、qfe0 の 2 つのエントリが表示されています。標準の qfe0 エントリには、MAC アドレスとリンクのローカルアドレスが含まれています。2 番目のエントリ qfe0:1 は、qfe0 インタフェースに追加された IPv6 アドレスに、擬似インタフェースが作成されたことを示しています。新しいグローバル IPv6 アドレス 2001:db8:3c4d:15:203:baff:fe13:14e1/64 には、ローカルルーターから通知されたサイト接頭辞とサブネット ID が含まれています。

次の手順

- 新しい IPv6 ノードをルーターとして構成する方法については、183 ページの「IPv6 ルーターの構成」を参照してください。
- ノード上でのアドレスの自動構成を無効にする方法については、183 ページの「IPv6 アドレスの自動構成を無効にする方法」を参照してください。
- 新しいノードをサーバーとして調整する方法については、194 ページの「サーバー上での IPv6 が有効なインタフェースの管理」を参照してください。

▼ IPv6 アドレスの自動構成を無効にする方法

ホストやサーバーのインタフェースに IPv6 アドレスを生成するときには、通常はアドレスの自動構成を使用するようにしてください。ただし、アドレスの自動構成を無効にしなければならない場合があります。特に、191 ページの「IPv6 トークンの構成」 で説明するようにトークンを手動で構成する場合には、この操作が必要になります。

1. **IPv6 ノードに Primary Administrator またはスーパーユーザーとしてログインします。**

   Primary Administrator 役割には、Primary Administrator プロファイルが含まれます。役割を作成してユーザーに役割を割り当てるには、『Oracle Solaris の管理: 基本管理』の第 2 章「Solaris 管理コンソールの操作 (タスク)」を参照してください。

2. **このノードの `/etc/inet/ndpd.conf` ファイルを作成します。**

   `/etc/inet/ndpd.conf` は、特定のノードのインタフェース変数を定義するファイルです。サーバーのすべてのインタフェースに対してアドレスの自動構成を無効にするためには、このファイルの内容が次のとおりである必要があります。

   if-variable-name `StatelessAddrConf false`

   `/etc/inet/ndpd.conf` の詳細については、`ndpd.conf(4)` のマニュアルページおよび 272 ページの「`ndpd.conf` 構成ファイル」 を参照してください。

3. **変更に合わせて、IPv6 デーモンを更新します。**

   ```
   # pkill -HUP in.ndpd
   ```

IPv6 ルーターの構成

ネットワークで IPv6 を構成するための最初の手順は、ルーターで IPv6 を構成することです。このセクションでは、ルーターを構成するために必要な複数のタスクについて説明します。これらのタスクは必ずしもすべてを実行する必要はなく、サイトの要件によって異なります。

IPv6 ルーターの構成 (タスクマップ)

次の表に示された順序で後続の作業を実行し、IPv6 ネットワークを構成します。表では、各タスクで実行する内容の説明と、タスクの具体的な実行手順が詳しく説明されている現在のマニュアル内のセクションを示しています。

タスク	説明	参照先
1. IPv6 の構成を始める前に、必要な前提条件をすべて満たしていることを確認します。	IPv6 が有効なルーターを構成する前に、計画タスクを完了し、IPv6 が有効なインタフェースを持つ Oracle Solaris をインストールしておく必要があります。	第 4 章「IPv6 ネットワークの計画 (手順)」 および 177 ページの「IPv6 インタフェースの構成」。
2. ルーターを構成します。	ネットワークのサイト接頭辞を定義します。	184 ページの「IPv6 対応のルーターを構成する方法」
3. ルーター上でトンネルインタフェースを構成します。	ルーター上で手動トンネルまたは 6to4 トンネルインタフェースを設定します。ローカルの IPv6 ネットワークがほかの隔離された IPv6 ネットワークと通信するためには、トンネルが必要になります。	■ 199 ページの「6to4 トンネルを構成する方法」 ■ 197 ページの「IPv6 over IPv4 トンネルを手動で構成する方法」 ■ 198 ページの「IPv6 over IPv6 トンネルを手動で構成する方法」 ■ 199 ページの「IPv4 over IPv6 トンネルを構成する方法」
4. ネットワーク上のスイッチを構成します。	ネットワーク構成にスイッチが含まれる場合、この時点で IPv6 用に構成します。	詳細については、スイッチに付属するマニュアルを参照してください。
5. ネットワーク上のハブを構成します。	ネットワーク構成にハブが含まれる場合、この時点で IPv6 用に構成します。	詳細については、ハブに付属するマニュアルを参照してください。
6. ネットワークネームサービスを IPv6 用に構成します。	ルーターを IPv6 用に構成したあと、IPv6 アドレスを認識するようにプライマリネームサービス (DNS、NIS、または LDAP) を構成します。	205 ページの「DNS に対する IPv6 アドレスを追加する方法」
7. (オプション) ホストおよびサーバー上で、IPv6 が有効なインタフェースのアドレスを変更します。	IPv6 ルーターを構成してから、IPv6 が有効なホストおよびサーバーに変更を加えます。	188 ページの「ホストとサーバーの IPv6 インタフェース構成の変更」
IPv6 をサポートするようにアプリケーションを構成します。	IPv6 をサポートする方法は、アプリケーションによって異なります。	アプリケーションに付属するマニュアルを参照してください。

▼ IPv6 対応のルーターを構成する方法

この手順では、Oracle Solaris のインストール時にルーターのすべてのインタフェースが IPv6 用に構成されているものとします。

1 **IPv6** ルーターになるシステムにおいて、**Primary Administrator** の役割を引き受けるか、スーパーユーザーになります。

Primary Administrator 役割には、Primary Administrator プロファイルが含まれます。役割を作成してユーザーに役割を割り当てるには、『Oracle Solaris の管理: 基本管理』の第 2 章「Solaris 管理コンソールの操作 (タスク)」を参照してください。

2 ルーター上で、インストール中に **IPv6** 用に構成したインタフェースを調査します。

```
# ifconfig -a
```

この出力を調べて、IPv6 用に構成したいインタフェースがリンクローカルアドレスで plumb されていることを確認します。次の `ifconfig -a` コマンドの出力例に、ルーターのインタフェースに構成されている IPv4 アドレスと IPv6 アドレスを示します。

```
lo0: flags=1000849 <UP,LOOPBACK,RUNNING,MULTICAST,IPv4> mtu 8232 index 1
        inet 127.0.0.1 netmask ff000000
dmfe0: flags=1000843 <UP,BROADCAST,RUNNING,MULTICAST,IPv4> mtu 1500 index 2
        inet 172.16.26.232 netmask ffffff00 broadcast 172.16.26.255
        ether 0:3:ba:11:b1:15
dmfe1: flags=1000843 <UP,BROADCAST,RUNNING,MULTICAST,IPv4 mtu 1500 index 3
        inet 172.16.26.220 netmask ffffff00 broadcast 172.16.26.255
        ether 0:3:ba:11:b1:16
lo0: flags=2000849 <UP,LOOPBACK,RUNNING,MULTICAST,IPv6> mtu 8252 index 1
        inet6 ::1/128
dmfe0: flags=2000841 <UP,RUNNING,MULTICAST,IPv6> mtu 1500 index 2
        ether 0:3:ba:11:b1:15
        inet6 fe80::203:baff:fe11:b115/10
dmfe1: flags=2000841 <UP,RUNNING,MULTICAST,IPv6> mtu 1500 index 3
        ether 0:3:ba:11:b1:16
        inet6 fe80::203:baff:fe11:b116/10
```

この出力を見ると、インストール中、プライマリネットワークインタフェース `dmfe0` と追加のインタフェース `dmfe1` が IPv6 リンクローカルアドレス `fe80::203:baff:fe11:b115/10` と `fe80::203:baff:fe11:b116/10` で構成されていることがわかります。

3 ルーターのすべてのインタフェース上で、**IPv6** パケット転送を構成します。

Solaris 10 11/03 以前のリリースの場合は、次のコマンドを使用します。

```
# routeadm -e ipv6-forwarding -u
```

次のいずれかを使用して、パケット転送を有効にします。

- `routeadm` コマンドを次のように使用します。

  ```
  # routeadm -e ipv6-forwarding -u
  ```

- サービス管理機能 (SMF) コマンドを次のように使用します。

  ```
  # svcadm enable ipv6-forwarding
  ```

4 ルーティングデーモンを起動します。

`in.ripngd` デーモンは IPv6 ルーティングを処理します。次のいずれかの方法で、IPv6 ルーティングをオンに設定します。

Solaris 10 11/06 以前のリリースの場合は、次のコマンドを入力して `in.ripngd` を起動します。

```
# routeadm -e ipv6-routing
# routeadm -u
```

- `routeadm` コマンドを次のように使用します。

  ```
  # routeadm -e ipv6-routing -u
  ```

- 適切な SMF コマンドを次のように使用します。

  ```
  # svcadm enable ripng:default
  ```

`routeadm` コマンドの構文については、`routeadm(1M)` のマニュアルページを参照してください。

5 **`/etc/inet/ndpd.conf`** ファイルを作成します。

`/etc/inet/ndpd.conf` には、ルーターが通知するサイト接頭辞などの構成情報を指定します。このファイルを `in.ndpd` デーモンが読み取って、IPv6 近傍検察プロトコルを実装します。

変数と指定できる値のリストについては、272 ページの「`ndpd.conf` 構成ファイル」と `ndpd.conf(4)` のマニュアルページを参照してください。

6 次のテキストを **`/etc/inet/ndpd.conf`** ファイルに入力します。

```
ifdefault AdvSendAdvertisements true
prefixdefault AdvOnLinkFlag on AdvAutonomousFlag on
```

このテキストは、ルーターの IPv6 用に構成されたすべてのインタフェース経由で、ルーター広告を送信することを `in.ndpd` デーモンに指示します。

7 ルーターのさまざまなインタフェースでサイト接頭辞を構成するには、**`/etc/inet/ndpd.conf`** ファイルに別のテキストを追加します。

このテキストの書式は次のとおりである必要があります。

`prefix` *global-routing-prefix*:*subnet ID*`/64` *interface*

次の `/etc/inet/ndpd.conf` ファイルの例は、サイト接頭辞 `2001:0db8:3c4d::/48` をインタフェース `dmfe0` および `dmfe1` 経由で通知するようにルーターを構成します。

```
ifdefault AdvSendAdvertisements true
prefixdefault AdvOnLinkFlag on AdvAutonomousFlag on

if dmfe0 AdvSendAdvertisements 1
prefix 2001:0db8:3c4d:15::0/64 dmfe0

if dmfe1 AdvSendAdvertisements 1
prefix 2001:0db8:3c4d:16::0/64 dmfe1
```

8 システムをリブートします。

IPv6 ルーターは、`ndpd.conf` ファイルにあるサイト接頭辞をローカルリンクに通知し始めます。

例 7–3 IPv6 インタフェースを示す `ifconfig` コマンドの出力

次の例に、183 ページの「IPv6 ルーターの構成」の手順を行なったあとに受信するような `ifconfig -a` コマンドの出力を示します。

```
lo0: flags=1000849 <UP LOOPBACK,RUNNING,MULTICAST,IPv4> mtu 8232 index 1
        inet 127.0.0.1 netmask ff000000
dmfe0: flags=1000843 <UP,BROADCAST,RUNNING,MULTICAST,IPv4> mtu 1500 index 2
        inet 172.16.15.232 netmask ffffff00 broadcast 172.16.26.255
        ether 0:3:ba:11:b1:15
dmfe1: flags=1000843 <UP,BROADCAST,RUNNING,MULTICAST,IPv4 mtu 1500 index 3
        inet 172.16.16.220 netmask ffffff00 broadcast 172.16.26.255
        ether 0:3:ba:11:b1:16
lo0: flags=2000849 <UP,LOOPBACK,RUNNING,MULTICAST,IPv6> mtu 8252 index 1
        inet6 ::1/128
dmfe0: flags=2100841 <UP,RUNNING,MULTICAST,ROUTER,IPv6> mtu 1500 index 2
        ether 0:3:ba:11:b1:15
        inet6 fe80::203:baff:fe11:b115/10
dmfe0:1: flags=2180841 <UP,RUNNING,MULTICAST,ADDRCONF,ROUTER,IPv6> mtu 1500
          index 2
        inet6 2001:db8:3c4d:15:203:baff:fe11:b115/64
dmfe1: flags=2100841 <UP,RUNNING,MULTICAST,ROUTER,IPv6> mtu 1500 index 3
        ether 0:3:ba:11:b1:16
        inet6 fe80::203:baff:fe11:b116/10
dmfe1:1: flags=2180841 <UP,RUNNING,MULTICAST,ADDRCONF,ROUTER,IPv6> mtu 1500
           index 3
        inet6 2001:db8:3c4d:16:203:baff:fe11:b116/64
```

この例では、IPv6 用に構成されている各インタフェースは、この時点で 2 つのアドレスを持っています。`dmfe0` のようなインタフェース名を持つエントリは、そのインタフェースのリンクローカルアドレスを示します。`dmfe0:1` のような *interface:n* 形式のエントリは、グローバル IPv6 アドレスを示します。このアドレスには、インタフェース ID に加えて、`/etc/ndpd.conf` ファイルに構成されているサイト接頭辞が含まれます。

参照
- IPv6 ネットワークトポロジで識別したルーターからのトンネルを構成する方法については、196 ページの「IPv6 サポート用のトンネルの構成」を参照してください。
- ネットワーク上のスイッチやハブを構成する方法については、スイッチまたはハブに付属するドキュメントを参照してください。
- IPv6 ホストを構成する方法については、188 ページの「ホストとサーバーの IPv6 インタフェース構成の変更」を参照してください。
- サーバーの IPv6 サポートを向上させる方法については、194 ページの「サーバー上での IPv6 が有効なインタフェースの管理」を参照してください。

- IPv6 のコマンド、ファイル、およびデーモンの詳細については、272 ページの「Oracle Solaris の IPv6 の実装」を参照してください。

ホストとサーバーの IPv6 インタフェース構成の変更

このセクションでは、ノードがホストまたはサーバーのときに、そのノードで IPv6 が有効なインタフェースの構成を変更する方法について説明します。ほとんどの場合、IPv6 が有効なインタフェースでは、84 ページの「ステートレス自動構成の概要」で説明しているようにアドレスの自動構成を使用するようにしてください。ただし、インタフェースの IPv6 アドレスの変更が必要な場合は、このセクションのタスクの説明に従って変更できます。

IPv6 インタフェース構成の変更 (タスクマップ)

次の表に、既存の IPv6 ネットワークに変更を加えるための各種作業の一覧を示します。表では、各タスクで実行する内容の説明と、タスクの具体的な実行手順が詳しく説明されている現在のマニュアル内のセクションを示しています。

タスク	説明	参照先
IPv6 アドレスの自動構成を無効にします。	このタスクは、IPv6 アドレスのインタフェース ID 部分を手動で構成する必要がある場合に使用します。	183 ページの「IPv6 アドレスの自動構成を無効にする方法」
ホストの一時アドレスを作成します。	ランダムに作成される一時アドレスを構成し、それをアドレスの下位 64 ビットとして使用することで、ホストのインタフェース ID を隠します。	189 ページの「一時アドレスを構成する方法」
システムのインタフェース ID のトークンを構成します。	IPv6 アドレスのインタフェース ID として使用される 64 ビットのトークンを作成します。	192 ページの「ユーザー指定の IPv6 トークンを構成する方法」

インタフェースに対する一時アドレスの使用

IPv6 「一時アドレス」には、インタフェースの MAC アドレスの代わりに、インタフェース ID としてランダムに生成された 64 ビットの数字が含まれます。匿名にしておきたい IPv6 ノード上の任意のインタフェースに対しては、一時アドレスを使用できます。たとえば、公開 Web サーバーにアクセスする必要があるホストのインタフェースに対しては、一時アドレスを使用したい場合もあります。一時アドレスには、IPv6 プライバシー拡張が実装されます。これらの拡張機能については、RFC

3041"Privacy Extensions for Stateless Address Autoconfiguration in IPv6" (http://www.ietf.org/rfc/rfc3041.txt?number=3041) を参照してください。

1 つまたは複数のインタフェースに対して一時アドレスを有効にする必要がある場合は、/etc/inet/ndpd.conf ファイルを使用します。しかし、標準の自動構成された IPv6 アドレスとは異なり、一時アドレスは、64 ビットのサブネット接頭辞とランダムに生成された 64 ビット数から構成されます。このランダムな数は、IPv6 アドレスのインタフェース ID 部分になります。リンクローカルアドレスでは、一時アドレスはインタフェース ID としては生成されません。

一時アドレスの *preferred lifetime* のデフォルトは、1 日です。一時アドレスの生成を有効にした場合、/etc/inet/ndpd.conf ファイルでは次の変数も構成できます。

valid lifetime TmpValidLifetime	一時アドレスが存在できる寿命。この寿命を過ぎると、そのアドレスはホストから削除されます。
preferred lifetime TmpPreferredLifetime	一時アドレスが無効にされるまでの時間。この時間は、valid lifetime よりも短くします。
address regeneration	preferred lifetime が満了するまでの時間。この時間内に、ホストは新しい一時アドレスを生成します。

一時アドレスの時間を表現するには、次の書式を使用します。

n	*n* 秒数 (デフォルト)
n h	*n* 時間数 (h)
n d	*n* 日数 (d)

▼ 一時アドレスを構成する方法

1. **IPv6 ホストに Primary Administrator またはスーパーユーザーとしてログインします。**

   Primary Administrator 役割には、Primary Administrator プロファイルが含まれます。役割を作成してユーザーに役割を割り当てるには、『Oracle Solaris の管理: 基本管理』の第 2 章「Solaris 管理コンソールの操作 (タスク)」を参照してください。

2. **必要に応じて、ホストのインタフェースの IPv6 を有効にします。**

   179 ページの「現在のセッションの IPv6 インタフェースを有効にする方法」を参照してください。

3. **/etc/inet/ndpd.conf ファイルを編集して、一時アドレスの生成を有効にします。**

   - ホストのすべてのインタフェースに対して一時アドレスを構成するには、次の行を /etc/inet/ndpd.conf ファイルに追加します。

     ```
     ifdefault TmpAddrsEnabled true
     ```

- 特定のインタフェースに対して一時アドレスを構成するには、次の行を `/etc/inet/ndpd.conf` ファイルに追加します。

  `if` *interface* **`TmpAddrsEnabled true`**

4 (オプション) 一時アドレスの **valid lifetime** を指定します。

`ifdefault TmpValidLifetime` *duration*

この構文は、ホストのすべてのインタフェースに対して valid lifetime を指定します。*duration* の値は、秒、時間、または日です。valid lifetime のデフォルトは 7 日です。`TmpValidLifetime` に `if` *interface* キーワードを使用すると、特定のインタフェースに対して一時アドレスの valid lifetime を指定できます。

5 (オプション) 一時アドレスの **preferred lifetime** を指定します。この寿命を過ぎると、一時アドレスは無効になります。

`if` *interface* **`TmpPreferredLifetime`** *duration*

この構文は、特定のインタフェースに対して一時アドレスの preferred lifetime を指定します。preferred lifetime のデフォルトは 1 日です。`TmpPreferredLifetime` に `ifdefault` キーワードを使用すると、ホストのすべてのインタフェースに対して preferred lifetime を指定できます。

注 – デフォルトアドレス選択では、無効にされた IPv6 アドレスには低い優先順位が与えられます。IPv6 一時アドレスが無効にされると、デフォルトアドレス選択によって、パケットのソースアドレスとして無効でないアドレスが選択されます。無効でないアドレスは、自動的に生成された IPv6 アドレス、またはインタフェースの IPv4 アドレス (使用できる場合) になります。デフォルトアドレス選択の詳細については、232 ページの「デフォルトアドレス選択の管理」を参照してください。

---

6 (オプション) アドレスを無効にするまでの時間を指定します。この間に、ホストは新しい一時アドレスを生成する必要があります。

`ifdefault TmpRegenAdvance duration`

この構文は、ホストのすべてのインタフェースに対して、一時アドレスを無効にするまでの時間を指定します。デフォルトは 5 秒です。

7 **`in.ndpd`** デーモンの構成を変更します。

```
# pkill -HUP in.ndpd
# /usr/lib/inet/in.ndpd
```

8 例 7–5 に示すように、**`ifconfig -a6`** コマンドを実行することによって、一時アドレスが作成されたことを確認します。

`ifconfig` コマンドの出力において、インタフェース定義と同じ行に `TEMPORARY` というキーワードが存在します。

例 7–4 /etc/inet/ndpd.conf ファイルの一時アドレス変数

次に、プライマリネットワークインタフェースに対して一時アドレスを有効にした /etc/inet/ndpd.conf ファイルの例 (一部) を示します。

```
ifdefault TmpAddrsEnabled true

ifdefault TmpValidLifetime 14d

ifdefault TmpPreferredLifetime 7d

ifdefault TmpRegenAdvance 6s
```

例 7–5 一時アドレスを有効にした ifconfig -a6 コマンドの出力

次に、一時アドレスを作成したあとにおける ifconfig コマンドの出力の例を示します。

```
# ifconfig -a6
lo0: flags=2000849 <UP,LOOPBACK,RUNNING,MULTICAST,IPv6> mtu 8252 index 1
     inet6 ::1/128
hme0: flags=2000841 <UP,RUNNING,MULTICAST,IPv6> mtu 1500 index 2
     ether 8:0:20:b9:4c:54
     inet6 fe80::a00:20ff:feb9:4c54/10
hme0:1: flags=2080841 <UP,RUNNING,MULTICAST,ADDRCONF,IPv6> mtu 1500 index 2
     inet6 2001:db8:3c4d:15:a00:20ff:feb9:4c54/64
hme0:2: flags=802080841<UP,RUNNING,MULTICAST,ADDRCONF,IPv6,TEMPORARY> mtu 1500 index 2
      inet6 2001:db8:3c4d:15:7c37:e7d1:fc9c:d2cb/64
```

インタフェース hme0:2 に続く行に TEMPORARY というキーワードが含まれていることに注目してください。この行は、2001:db8:3c4d:15:7c37:e7d1:fc9c:d2cb/64 が一時インタフェース ID を持っていることを示します。

参照
- ネームサービスが IPv6 アドレスをサポートするように設定する方法については、205 ページの「ネームサービスの IPv6 サポート用の構成」を参照してください。
- サーバー上で IPv6 アドレスを構成する方法については、192 ページの「ユーザー指定の IPv6 トークンを構成する方法」を参照してください。
- IPv6 ノード上での活動を監視する方法については、第 8 章「TCP/IP ネットワークの管理 (手順)」を参照してください。

IPv6 トークンの構成

IPv6 アドレスの 64 ビットインタフェース ID は、*IPv6 Addressing Overview*で説明したように、「76 ページの「IPv6 アドレス指定の概要」」とも呼ばれます。トークンは、アドレスが自動構成されるときに、インタフェースの MAC アドレスに関連付け

られます。ほとんどの場合、ルーティングを行わないノード (IPv6 ホストと IPv6 サーバー) では、自動構成されたトークンを使用するようにしてください。

ただし、システムが保守されるときにインタフェースが定期的に交換されるサーバーでは、自動構成されたトークンを使用すると問題が発生することがあります。インタフェースカードが変更されると、MAC アドレスも変更されます。その結果、IP アドレスが変わらないことを前提とするサーバーでは、問題が発生することがあります。ネットワークインフラストラクチャーの各ノード (DNS、NIS など) に、サーバーのインタフェースに固有の IPv6 アドレスが保存されている場合があります。

アドレスが変わることで発生する問題を回避するために、IPv6 アドレスのインタフェース ID として使用されるトークンを手動で構成できます。トークンを作成するには、IPv6 アドレスのインタフェース ID 部分に相当する 64 ビット以下の 16 進数を指定します。それ以降は、アドレスが自動構成されるときに近傍検索によって作成されるインタフェース ID は、インタフェースの MAC アドレスからは作成されません。代わりに、手動で作成したトークンがインタフェースID になります。このトークンは、カードを交換しても、インタフェースに割り当てられたままになります。

注 – ユーザー指定のトークンと一時アドレスとの違いは、一時アドレスがランダムに生成されるのに対し、ユーザー指定のトークンはユーザーが明示的に作成する点です。

▼ ユーザー指定の IPv6 トークンを構成する方法

次の手順は、インタフェースが定期的に置き換えられるサーバーで特に役立ちます。また、任意の IPv6 ノード上でユーザー指定のトークンを構成する場合にも有効です。

1 **トークンを構成するインタフェースが plumb されていることを確認します。**

IPv6 アドレスのトークンを構成するときは、そのインタフェースが plumb されている必要があります。

```
# ifconfig a6
qfe0: flags=2000841 <UP,RUNNING,MULTICAST,IPv6> mtu 1500 index 2
        ether 0:3:ba:13:14:e1
        inet6 fe80::203:baff:fe13:14e1/10
```

この出力から、ネットワークインタフェース `qfe0` が plumb されていて、リンクのローカルアドレス `fe80::203:baff:fe13:14e1/10` が割り当てられていることがわかります。このアドレスは、インストール中に自動的に構成されています。

2. ノードのインタフェースのトークンとして使用する、1 つまたは複数の 64 ビットの 16 進数を作成します。トークンの例については、81 ページの「リンクローカルユニキャストアドレス」を参照してください。

3. 各インタフェースをトークンで構成します。

   次の形式の `ifconfig` コマンドを使用して、ユーザー指定のインタフェース ID (トークン) を各インタフェースに割り当てます。

   ```
   ifconfig interface inet6  token address/64
   ```

   たとえば、インタフェース `qfe0` をトークンで構成するには、次のコマンドを使用します。

   ```
   # ifconfig qfe0 inet6 token ::1a:2b:3c:4d/64
   ```

   ユーザー指定のトークンを割り当てるインタフェースごとに、この手順を繰り返します。

4. (オプション) 新しい IPv6 アドレスがリブート後も保持されるように設定します。

   a. トークンを構成したインタフェースごとに、**/etc/hostname6.***interface* ファイルを編集または作成します。

   b. 各 **/etc/hostname6.***interface* ファイルの末尾に次のテキストを追加します。

      token ::*token-name*/64

      たとえば、`/etc/hostname6.`*interface* ファイルの末尾に次のテキストを追加します。

      ```
      token ::1a:2b:3c:4d/64
      ```

   システムをリブートしたあとに、`/etc/hostname6.`*interface* ファイルに構成したトークンがそのインタフェースの IPv6 アドレスに適用されます。この IPv6 アドレスは、それ以降何度リブートしても保持されます。

5. 変更に合わせて、IPv6 デーモンを更新します。

   ```
   # pkill -HUP in.ndpd
   ```

例 7–6 ユーザー指定のトークンを IPv6 インタフェースに構成する

次の例のインタフェース `bge0:1` には、自動構成された IPv6 アドレスが割り当てられています。サブネット接頭辞 `2001:db8:3c4d:152:/64` は、ノードのローカルリンク上のルーターから通知されます。インタフェース ID `2c0:9fff:fe56:8255` は、`bge0:1` の MAC アドレスから生成されます。

```
# ifconfig -a6
lo0: flags=2002000849 <UP,LOOPBACK,RUNNING,MULTICAST,IPv6,VIRTUAL> mtu 8252 index 1
```

```
        inet6 ::1/128
bge0: flags=2100801 <UP,MULTICAST,IPv6> mtu 1500 index 5
        inet6 fe80::2c0:9fff:fe56:8255/10
        ether 0:c0:9f:56:82:55
bge0:1: flags=2180801 <UP, MULTICAST,ADDRCONF,IPv6>mtu 1500 index 5
        inet6 2001:db8:3c4d:152:c0:9fff:fe56:8255/64
# ifconfig bge0 inet6 token ::1a:2b:3c:4d/64
# vi /etc/hostname6.bge0
token ::1a:2b:3c:4d/64
# pkill -HUP in.ndpd
# ifconfig -a6
lo0: flags=2002000849 <UP,LOOPBACK,RUNNING,MULTICAST,IPv6,VIRTUAL> mtu 8252 index 1
        inet6 ::1/128
bge0: flags=2100801 <UP,MULTICAST,IPv6> mtu 1500 index 5
        inet6 fe80::2c0:9fff:fe56:8255/10
        ether 0:c0:9f:56:82:55
bge0:1: flags=2180801 <UP, MULTICAST,ADDRCONF,IPv6>mtu 1500 index 5
        inet6 2001:db8:3c4d:152:1a:2b:3c:4d/64
```

トークンの構成が終了すると、`bge0:1` の 2 番目のステータス行のグローバルアドレスは、そのインタフェース ID に構成された `1a:2b:3c:4d` になります。

参照
- ネームサービスをサーバーの IPv6 アドレスで更新する方法については、205 ページの「ネームサービスの IPv6 サポート用の構成」を参照してください。
- サーバーのパフォーマンスを監視する方法については、第 8 章「TCP/IP ネットワークの管理 (手順)」を参照してください。

サーバー上での IPv6 が有効なインタフェースの管理

サーバーで IPv6 を使用することを計画するときは、サーバーのインタフェースの IPv6 を有効にするために、いくつかのことを決定する必要があります。それらの決定は、インタフェースの IPv6 アドレスのインタフェース ID (「トークン」とも呼ばれる) を構成するときに、どのような方法を採用するかに影響します。

▼ サーバーのインタフェースの IPv6 を有効にする方法

始める前に この手順では、次のことを前提としています。

- Oracle Solaris がサーバー上にすでにインストールされていること。
- Oracle Solaris のインストール時またはそのあとに、177 ページの「IPv6 インタフェースの構成」の手順を使用して、サーバーのインタフェースの IPv6 を有効にしています。

これらに該当する場合は、IPv6 がサポートされるようにアプリケーションソフトウェアをアップグレードします。IPv4 プロトコルスタックで動作するアプリ

ケーションの多くは、IPv6 でも正常に動作します。詳細は、92 ページの「IPv6 をサポートするためにネットワークサービスを準備する方法」を参照してください。

1. サーバー上で、**Primary Administrator** の役割を引き受けるか、スーパーユーザーになります。

   Primary Administrator 役割には、Primary Administrator プロファイルが含まれます。役割を作成してユーザーに役割を割り当てるには、『Oracle Solaris の管理: 基本管理』の第 2 章「Solaris 管理コンソールの操作 (タスク)」を参照してください。

2. サーバーと同じリンク上のルーターに **IPv6** サブネット接頭辞が構成されていることを確認します。

   詳細は、183 ページの「IPv6 ルーターの構成」を参照してください。

3. サーバーの **IPv6** が有効なインタフェースのインタフェース **ID** に適した方法を使用します。

   デフォルトでは、IPv6 アドレスの自動構成によって IPv6 アドレスのインタフェース ID 部分が作成されるときに、インタフェースの MAC アドレスが使用されます。インタフェースの IPv6 アドレスが既知の場合には、インタフェースが切り替わると、問題が発生することがあります。新しいインタフェースの MAC アドレスは、別のアドレスになります。アドレスが自動構成されると、新しいインタフェース ID が生成されます。

   - IPv6 が有効なインタフェースを置き換えないで使用する場合は、自動構成された IPv6 アドレスを使用します (84 ページの「IPv6 アドレスの自動構成」を参照)。
   - IPv6 が有効なインタフェースをローカルネットワークの外部には匿名で表示する必要がある場合は、ランダムに生成されたトークンをインタフェース ID に使用することを検討します。手順および例については、189 ページの「一時アドレスを構成する方法」を参照してください。
   - IPv6 が有効なインタフェースを定期的に切り替えて使用する場合は、インタフェース ID のトークンを作成します。手順および例については、192 ページの「ユーザー指定の IPv6 トークンを構成する方法」を参照してください。

IPv6 サポート用にトンネルを構成するためのタスク (タスクマップ)

次の表に、さまざまな種類の IPv6 トンネルを構成するための各種タスクの一覧を示します。表では、各タスクで実行する内容の説明と、タスクの具体的な実行手順が詳しく説明されている現在のマニュアル内のセクションを示しています。

タスク	説明	参照先
IPv6 over IPv4 トンネルを手動で構成します。	IPv4 ネットワークを経由する IPv6 トンネルを手動で構成します。より大きく、大部分が IPv4 である企業ネットワーク内にあるリモートの IPv6 ネットワークに到達するための解決方法です。	197 ページの「IPv6 over IPv4 トンネルを手動で構成する方法」
IPv6 over IPv6 トンネルを手動で構成します。	IPv6 ネットワークを経由する IPv6 トンネルを手動で構成します。通常、巨大な企業ネットワークで使用される解決方法です。	198 ページの「IPv6 over IPv6 トンネルを手動で構成する方法」
IPv4 over IPv6 トンネルを手動で構成します。	IPv6 ネットワークを経由する IPv4 トンネルを手動で構成します。IPv4 と IPv6 の両方のネットワークを持つ巨大なネットワークで使用される解決方法です。	199 ページの「IPv4 over IPv6 トンネルを構成する方法」
IPV6 over IPv4 トンネル (6to4 トンネル) を自動で構成します。	自動 6to4 トンネルを作成します。外部の IPv6 サイトにインターネット経由で到達するための解決方法です。	199 ページの「6to4 トンネルを構成する方法」
6to4 ルーターと 6to4 リレールーター間にトンネルを構成します。	`6to4relay` コマンドを使用して、6to4 リレールーターとの間のトンネルを有効にする解決方法です。	203 ページの「6to4 リレールーターとの間の 6to4 トンネルを構成する方法」

IPv6 サポート用のトンネルの構成

IPv6 ネットワークは、ほとんどの場合、巨大な IPv4 ネットワーク内で孤立しています。IPv6 ネットワーク上のノードは、企業 (内部) またはリモート (外部) の孤立した IPv6 ネットワーク上のノードと通信する必要もあります。このような場合、IPv6 ホストはトンネルのエンドポイントとしても機能しますが、通常は、IPv6 ルーター間にトンネルを構成します。トンネルの計画については、94 ページの「ネットワークトポロジにおけるトンネルの計画」を参照してください。

IPv6 ネットワーク用に自動的または手動で構成されたトンネルを設定できます。Oracle Solaris IPv6 実装がサポートするトンネルカプセル化の種類は、次のとおりです。

- IPv4 トンネル経由の IPv6
- IPv6 トンネル経由の IPv6
- IPv6 トンネル経由の IPv4

- 6to4 トンネル

トンネルの概念については、295 ページの「IPv6 トンネル」を参照してください。

▼ IPv6 over IPv4 トンネルを手動で構成する方法

この手順では、IPv6 ノードから IPv4 ネットワーク経由でリモートの IPv6 ノードに到達するトンネルを設定する方法について説明します。

1. トンネルのローカル側のエンドポイントに **Primary Administrator** または スーパーユーザーとしてログインします。

   Primary Administrator 役割には、Primary Administrator プロファイルが含まれます。役割を作成してユーザーに役割を割り当てるには、『Oracle Solaris の管理: 基本管理』の第 2 章「Solaris 管理コンソールの操作 (タスク)」を参照してください。

2. **/etc/hostname6.ip.tun** *n* ファイルを作成します。

   ここで、*n* はトンネル番号です (最初のトンネルがゼロ)。次に、次の手順に従って、エントリを追加します。

   a. トンネルソースアドレスとトンネル宛先アドレスを追加します。

      `tsrc` *IPv4-source-address* `tdst` *IPv4-destination-address* `up`

   b. (オプション) ソース **IPv6** アドレスと宛先 **IPv6** アドレスの論理インタフェースを追加します。

      `addif` *IPv6-source-address* *IPv6-destination-address*

      このインタフェースに対してアドレスを自動構成したい場合は、この手順を省きます。各トンネルに対するリンクローカルアドレスを構成する必要はありません。

3. システムをリブートします。

4. トンネルの反対側のエンドポイントでも、このタスクを繰り返します。

例 7–7 /etc/hostname6.ip.tun ファイルにおける手動 IPv6 over IPv4 トンネル用のエントリ

次の `/etc/hostname6.ip.tun` ファイルの例に、グローバルソースアドレスとグローバル宛先アドレスを手動で構成したトンネルを示します。

```
tsrc 192.168.8.20 tdst 192.168.7.19 up
addif 2001:db8:3c4d:8::fe12:528 2001:db8:3c4d:7:a00:20ff:fe12:1234 up
```

▼ IPv6 over IPv6 トンネルを手動で構成する方法

この手順では、IPv6 ノードから IPv6 ネットワーク経由でリモートの IPv6 ノードに到達するトンネルを設定する方法について説明します。

1 トンネルのローカル側のエンドポイントに **Primary Administrator** または スーパーユーザーとしてログインします。

Primary Administrator 役割には、Primary Administrator プロファイルが含まれます。役割を作成してユーザーに役割を割り当てるには、『Oracle Solaris の管理: 基本管理』の第 2 章「Solaris 管理コンソールの操作 (タスク)」を参照してください。

2 **/etc/hostname6.ip6.tun** *n* ファイルを作成します。

n には 0、1、2 などの値を使用します。次に、次の手順に従って、エントリを追加します。

a. トンネルソースアドレスとトンネル宛先アドレスを追加します。

```
tsrc IPv6-source-address tdst IPv6-destination-address
IPv6-packet-source-address IPv6-packet-destination-address up
```

b. (オプション) ソース **IPv6** アドレスと宛先 **IPv6** アドレスの論理インタフェースを追加します。

```
addif IPv6-source-address  IPv6-destination-address up
```

このインタフェースに対してアドレスを自動構成したい場合は、この手順を省きます。各トンネルに対するリンクローカルアドレスを構成する必要はありません。

3 システムをリブートします。

4 トンネルの反対側のエンドポイントでも、この作業を繰り返します。

例 7–8 /etc/hostname6.ip6.tun ファイルにおける IPv6 over IPv6 トンネル用のエントリ

次に、IPv6 over IPv6 トンネル用のエントリの例を示します。

```
tsrc 2001:db8:3c4d:22:20ff:0:fe72:668c tdst 2001:db8:3c4d:103:a00:20ff:fe9b:a1c3
fe80::4 fe80::61 up
```

▼ IPv4 over IPv6 トンネルを構成する方法

この手順では、IPv6 ネットワークを経由して、2 つの IPv4 ホスト間をつなげるトンネルを構成する方法について説明します。この手順を使用するのは、企業のネットワークが異機種混在の環境であり、IPv6 サブネットが IPv4 サブネットを分離している場合です。

1. トンネルのローカル **(IPv4)** 側のエンドポイントに **Primary Administrator** またはスーパーユーザーとしてログインします。

   Primary Administrator 役割には、Primary Administrator プロファイルが含まれます。役割を作成してユーザーに役割を割り当てるには、『Oracle Solaris の管理: 基本管理』の第 2 章「Solaris 管理コンソールの操作 (タスク)」を参照してください。

2. **`/etc/hostname.ip6.tun`***n* ファイルを作成します。

   n には 0、1、2 などの値を使用します。次に、次の手順に従って、エントリを追加します。

   a. トンネルソースアドレスとトンネル宛先アドレスを追加します。

      `tsrc` *IPv6-source-address* `tdst` *IPv6-destination-address*

   b. (オプション) ソース **IPv6** アドレスと宛先 **IPv6** アドレスの論理インタフェースを追加します。

      `addif` *IPv6-source-address* *IPv6-destination-address* `up`

3. ローカルホストをリブートします。

4. トンネルの反対側のエンドポイントでも、この作業を繰り返します。

例 **7–9** `/etc/hostname6.ip6.tun` ファイルにおける IPv4 over IPv6 トンネル用のエントリ

次に、IPv6 トンネル経由の IPv4 のエントリの例を示します。

```
tsrc 2001:db8:3c4d:114:a00:20ff:fe72:668c tdst 2001:db8:3c4d:103:a00:20ff:fe9b:a1c3
10.0.0.4 10.0.0.61 up
```

▼ 6to4 トンネルを構成する方法

企業の IPv6 ネットワークがリモートの IPv6 ネットワークと通信する必要がある場合、自動 6to4 トンネルを使用することを考えてください。6to4 トンネルを構成するプロセスには、境界ルーターを「*6to4*」ルーターとして構成する手順が含まれます。6to4 ルーターは、ローカルのネットワークとリモートの IPv6 ネットワークにあるエンドポイントルーター間における 6to4 トンネルのエンドポイントとして機能します。

始める前に　IPv6 ネットワーク上で 6to4 経路制御を構成する前に、次のことを行う必要があります。

- 予定されている 6to4 サイトのすべての適切なノード上で IPv6 を構成します (188 ページの「ホストとサーバーの IPv6 インタフェース構成の変更」を参照)。
- 6to4 ルーターとして使用するため、IPv4 ネットワークに接続された 1 台以上のルーターを選択します。
- IPv4 ネットワークに対して使用する 6to4 ルーターのインタフェースに一意の (全世界に 1 つしかない) IPv4 アドレスを構成します。この IPv4 アドレスは静的なものでなければいけません。

注 – 第 12 章「DHCP について (概要)」で説明されているように、動的に割り当てられた IPv4 アドレスは使用しないでください。動的に割り当てられたグローバルなアドレスは時間の経過とともに変わるため、IPv6 アドレス指定計画に悪影響を及ぼす可能性があります。

---

1 予想されている **6to4** ルーターに **Primary Administrator** またはスーパーユーザーとしてログインします。

Primary Administrator 役割には、Primary Administrator プロファイルが含まれます。役割を作成してユーザーに役割を割り当てるには、『Oracle Solaris の管理: 基本管理』の第 2 章「Solaris 管理コンソールの操作 (タスク)」を参照してください。

2 ルーター上に **6to4** 疑似インタフェースを構成します。つまり、**`/etc/hostname6.ip.6to4tun0`** ファイルを作成します。

- サブネット ID が 0 でホスト ID が 1 の推奨されている規則を使用する場合は、`/etc/hostname6.ip.6to4tun0` では次に示す短い形式を使用してください。

  `tsrc` *IPv4-address* `up`
- サブネット ID とホスト ID にほかの規則を使用する場合は、`/etc/hostname6.ip.6to4tun0` では次に示す長い形式を使用してください。

  `tsrc` *IPv4-address* `2002:`*IPv4-address*`:`*subnet-ID*`:`*interface-ID*`:/64 up`

`/etc/hostname6.ip.6to4tun0` の必須パラメータは次のとおりです。

`tsrc`　このインタフェースがトンネルソースとして使用されることを示します。

IPv4-address　6to4 擬似インタフェースとなる物理インタフェース上で構成される IPv4 アドレスをドット付きの 10 進数形式で指定します。

残りのパラメータはオプションです。しかし、省略可能なパラメータを 1 つでも指定した場合、ほかのすべての省略可能なパラメータも指定する必要があります。

`2002`　6to4 接頭辞を指定します。

IPv4-address　疑似インタフェースの IPv4 アドレスを 16 進数表記で指定します。

subnet-ID	0 以外のサブネット ID を16 進表記で指定します。
interface-ID	1 以外のインタフェース ID を指定します。
`/64`	5to4 接頭辞の長さが 64 ビットであることを示します。
`up`	6to4 インタフェースを「up」として構成します。

注 – ネットワーク上の 2 つの IPv6 トンネルは、同じソースアドレスと同じ宛先アドレスを持つことはできません。同じアドレスを指定するとパケットは削除されます。このような状況は、6to4 ルーターが `atun` コマンドを通してトンネリングを実施する場合にも発生する可能性があります。`atun` については、`tun(7M)` のマニュアルページを参照してください。

---

3 **(オプション) ルーター上でさらに 6to4 擬似インタフェースを作成します。**

予定されている各 6to4 擬似インタフェースには、すでに構成された一意の (全世界に 1 つしかない) IPv4 アドレスが必要です。

4 **6to4 ルーターをリブートします。**

5 **インタフェースのステータスを確認します。**

```
# ifconfig ip.6to4tun0 inet6
```

インタフェースが正しく構成されている場合は、次のようなメッセージが表示されます。

```
ip.6to4tun0: flags=2200041<UP,RUNNING,NONUD,IPv6> mtu 1480 index 11
        inet tunnel src 111.222.33.44
        tunnel hop limit 60
        inet6 2002:6fde:212c:10:/64
```

6 **6to4 経路制御を通知するために `/etc/inet/ndpd.conf` ファイルを編集します。**

詳細については、`ndpd.conf(4)` のマニュアルページを参照してください。

a. **最初の行で、通知を受け取るサブネットを指定します。**

`if` エントリを次の書式で作成してください。

```
if subnet-interface AdvSendAdvertisements 1
```

たとえば、インタフェース `hme0` に接続しているサブネットに対して 6to4 経路制御を通知するには、*subnet-interface* を `hme0` に置き換えます。

```
if hme0 AdvSendAdvertisements 1
```

b. 通知の **2** 行目として **6to4** 接頭辞を追加します。

`prefix` エントリを次の書式で作成してください。

`prefix 2002:`*IPv4-address*`:`*subnet-ID*`::/64` *subnet-interface*

7 ルーターをリブートします。

あるいは、`sighup` を `/etc/inet/in.ndpd` デーモンに発行しても、ルーター広告の送信を開始できます。これによって、各サブネット上の 6to4 接頭辞を受信する IPv6 ノードは、新しい 6to4 派生アドレスを自動構成します。

8 ノードに使用される **6to4** 派生の新しいアドレスを **6to4** サイトで使用されるネームサービスに追加します。

手順については、205 ページの「ネームサービスの IPv6 サポート用の構成」を参照してください。

例 7–10 6to4 ルーターの構成 (短い形式)

次に、`/etc/hostname6.ip.6to4tun0` の短い形式の例を示します。

```
# cat /etc/hostname6.ip.6to4tun0
tsrc 111.222.33.44 up
```

例 7–11 6to4 ルーターの構成 (長い形式)

次に、`/etc/hostname6.ip.6to4tun0` の長い形式の例を示します。

```
# cat /etc/hostname6.ip.6to4tun0
tsrc 111.222.33.44 2002:6fde:212c:20:1/64 up
```

例 7–12 6to4 疑似インタフェースを示す `ifconfig` の出力

次に、6to4 擬似インタフェースの `ifconfig` コマンドの出力例を示します。

```
# ifconfig ip.6to4tun0 inet6
ip.6to4tun0: flags=2200041<UP,RUNNING,NONUD,IPv6> mtu 1480 index 11
        inet tunnel src 192.168.87.188
        tunnel hop limit 60
        inet6 2002:c0a8:57bc::1/64
```

例 7–13 `/etc/inet/ndpd.conf` における 6to4 通知

次の `/etc/inet/ndpd.conf` ファイル例は、2 つのサブネット上の 6to4 経路制御を通知します。

```
if qfe0 AdvSendAdvertisements 1
prefix  2002:c0a8:57bc:10::/64 qfe0
```

```
if qfe1 AdvSendAdvertisements 1
prefix  2002:c0a8:57bc:2::/64 qfe1
```

参考 6to4 サイトにおける複数のルーターの構成

複数のルーターが存在するサイトの場合、6to4 ルーターの後ろに位置するルーターも 6to4 をサポートするように構成する必要がある場合があります。サイトに RIP が使用されている場合、6to4 以外のルーターそれぞれに、6to4 ルーターまでの静的ルートを指定する必要があります。市販の経路制御プロトコルを使用する場合は、6to4 ルーターとの間の静的なルートを構築する必要はありません。

▼ 6to4 リレールーターとの間の 6to4 トンネルを構成する方法

注意 – セキュリティー上の大きな問題のため、Oracle Solaris では、6to4 リレールーターのサポートはデフォルトでは無効になっています。6to4 リレールーターへのトンネルを作成するときのセキュリティー問題を参照してください。

始める前に 6to4 リレールーターとの間のトンネルを有効にする前に、次のタスクを完了しておく必要があります。

- 使用しているサイトの 6to4 ルーターを構成する (199 ページの「6to4 トンネルを構成する方法」を参照)
- 6to4 リレールーターとの間のトンネリングに伴うセキュリティー問題を検討する

1 **6to4 ルーターに Primary Administrator またはスーパーユーザーとしてログインします。**

Primary Administrator 役割には、Primary Administrator プロファイルが含まれます。役割を作成してユーザーに役割を割り当てるには、『Oracle Solaris の管理: 基本管理』の第 2 章「Solaris 管理コンソールの操作 (タスク)」を参照してください。

2 **次のどちらか一方を使用し、6to4 リレールーターとの間のトンネルを有効にします。**

- エニーキャスト 6to4 リレールーターとの間のトンネルを有効にします。

```
# /usr/sbin/6to4relay -e
```

-e オプションは、6to4 ルーターとエニーキャスト 6to4 リレールーターの間にトンネルを設定します。エニーキャスト 6to4 リレールーターは既知の IPv4 アドレス

192.88.99.1 を持っています。サイトに物理的にもっとも近いエニーキャストリレールーターが、6to4 トンネルのエンドポイントになります。このリレールーターは、6to4 サイトとネイティブ IPv6 サイト間のパケット転送を処理します。

エニーキャスト 6to4 リレールーターの詳細については、RFC 3068, "An Anycast Prefix for 6to4 Relay Routers" (ftp://ftp.rfc-editor.org/in-notes/rfc3068.txt) を参照してください。

- 特定の 6to4 リレールーターとの間のトンネルを有効にします。

  ```
  # /usr/sbin/6to4relay -e -a relay-router-address
  ```

  -a オプションは、特定のルーターアドレスが続くことを示します。*relay-router-address* には、トンネルを有効にするために使用する特定の 6to4 リレールーターの IPv4 アドレスを指定してください。

6to4 リレールーターとの間のトンネルは、6to4 トンネル擬似インタフェースが削除されるまでアクティブな状態を維持します。

3 **6to4 リレールーターとの間のトンネルが必要なくなったときには、このトンネルを削除します。**

```
# /usr/sbin/6to4relay -d
```

4 **(オプション) リブートを行なっても 6to4 リレールーターとの間のトンネルが持続するように設定します。**

サイトによっては、6to4 ルーターがリブートするたびに 6to4 リレールーターとの間のトンネルを元に戻さざるをえない場合があるでしょう。このようなシナリオをサポートするには、次を行う必要があります。

a. **/etc/default/inetinit ファイルを編集します。**

   変更が必要な行は、ファイルの最後にあります。

b. **ACCEPT6TO4RELAY=NO という行の値 "NO" を "YES" に変更します。**

c. **(オプション) 特定の 6to4 リレールーターとの間で、リブートを行なっても持続するトンネルを構築します。**

   パラメータ RELAY6TO4ADDR のために、アドレス 192.88.99.1 を、使用したい 6to4 リレールーターの IPv4 アドレスに変更してください。

例 7–14 6to4 リレールーターサポートのステータス情報の取得

/usr/bin/6to4relay コマンドを使用し、6to4 リレールーターのサポートが有効になっているかどうかを確認できます。次の例は、6to4 リレールーターのサポートを無効にした場合 (これが Oracle Solaris のデフォルト) の出力です。

```
# /usr/sbin/6to4relay
6to4relay: 6to4 Relay Router communication support is disabled.
```

6to4 リレールーターのサポートを有効にすると、次のメッセージが表示されます。

```
# /usr/sbin/6to4relay
6to4relay: 6to4 Relay Router communication support is enabled.
IPv4 remote address of Relay Router=192.88.99.1
```

ネームサービスの IPv6 サポート用の構成

このセクションでは、IPv6 サービスをサポートするように DNS ネームサービスと NIS ネームサービスを構成する方法について説明します。

注 – LDAP は IPv6 をサポートします。IPv6 固有な構成タスクは必要ありません。

DNS、NIS、および LDAP の管理の詳細については、『Solaris のシステム管理 (ネーミングとディレクトリサービス : DNS、NIS、LDAP 編)』を参照してください。

▼ DNS に対する IPv6 アドレスを追加する方法

1. プライマリまたはセカンダリの **DNS** サーバーに **Primary Administrator** またはスーパーユーザーとしてログインします。

   Primary Administrator 役割には、Primary Administrator プロファイルが含まれます。役割を作成してユーザーに役割を割り当てるには、『Oracle Solaris の管理: 基本管理』の第 2 章「Solaris 管理コンソールの操作 (タスク)」を参照してください。

2. 適切な **DNS** ゾーンファイルを編集して、**IPv6** が有効なノードごとに **AAAA** レコードを追加します。

   hostname `IN` `AAAA` *host-address*

3. **DNS** 逆ゾーンファイルを編集して、**PTR** レコードを追加します。

   hostaddress `IN` `PTR` *hostname*

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

例 7–15 DNS 逆ゾーンファイル

次に、逆ゾーンファイルにおける IPv6 アドレスの例を示します。

```
$ORIGIN    ip6.int.
8.2.5.0.2.1.e.f.f.f.9.2.0.0.a.0.6.5.2.9.0.0.0.0.0.0.0.0.2.0.0.0 \
    IN        PTR        vallejo.Eng.apex.COM.
```

IPv6 アドレスの NIS への追加

Solaris 10 11/06 以前のリリースでは、NIS 用に 2 つのマップが追加されていました。`ipnodes.byname` と `ipnodes.byaddr` です。これらのマップは、いずれも IPv4 と IPv6 のホスト名とアドレスの関連付けを含んでいました。IPv6 に対応するツールは、`ipnodes` NIS マップを使用していました。`hosts.byname` マップと `hosts.byaddr` マップは、IPv4 ホスト名とアドレスの関係しか含んでいませんでした。これらのマップは既存のアプリケーションのために変更されていません。`ipnodes` マップの管理は、`hosts.byname` マップと `hosts.byaddr` マップの管理方法と同様です。Solaris 10 11/06 の場合は、*hosts* マップを *ipnode* アドレスで更新すると、ipnode マップも同じ情報で更新されることに注意してください。

注 – Oracle Solaris 10 の後続のリリースでは、`ipnodes` マップは使用されません。`ipnodes` マップの IPv6 機能は、`hosts` マップで管理されるようになりました。

NIS マップを管理する手順については、『System Administration Guide: Naming and Directory Services (DNS, NIS, and LDAP)』の第 5 章「Setting Up and Configuring NIS Service」を参照してください。

▼ IPv6 ネームサービス情報を表示する方法

`nslookup` コマンドを使用すると、 IPv6 ネームサービス情報を表示できます。

1. 自分のユーザーアカウントで、**`nslookup`** コマンドを実行します。

   ```
   % /usr/sbin/nslookup
   ```

   デフォルトサーバー名とアドレスが表示され、`nslookup` コマンドの山括弧プロンプトが表示されます。

2. 特定のホストの情報を表示するには、山括弧プロンプトに次のコマンドを入力します。

   ```
   >set q=any
   >hostname
   ```

3. 次のコマンドを入力すると、**AAAA** レコードだけが表示されます。

   ```
   >set q=AAAA
   hostname
   ```

4 **exit** を入力して、**nslookup** コマンドを終了します。

例 **7–16** nslookup による IPv6 情報の表示

次に、IPv6 ネットワーク環境における nslookup コマンドの結果の例を示します。

```
%  /usr/sbin/nslookup
Default Server:  dnsserve.local.com
Address:  10.10.50.85
> set q=AAAA
> host85
Server:  dnsserve.local.com
Address:  10.10.50.85

host85.local.com      IPv6 address = 2::9256:a00:fe12:528
> exit
```

▼ DNS IPv6 PTR レコードの正確な更新を確認する方法

nslookup コマンドを使用して DNS IPv6 PTR レコードを表示します。

1 自分のユーザーアカウントで、**nslookup** コマンドを実行します。

```
% /usr/sbin/nslookup
```

デフォルトサーバー名とアドレスが表示され、 nslookup コマンドの山括弧プロンプトが表示されます。

2 **PTR** レコードを表示するには、山括弧プロンプトに次のコマンドを入力します。

```
>set q=PTR
```

3 **exit** を入力して、コマンドを終了します。

例 **7–17** nslookup コマンドによる PTR レコードの表示

次に、nslookup コマンドを使用して、PTR レコードを表示する例を示します。

```
%  /usr/sbin/nslookup
Default Server:  space1999.Eng.apex.COM
Address:  192.168.15.78
> set q=PTR
> 8.2.5.0.2.1.e.f.f.f.0.2.0.0.a.0.6.5.2.9.0.0.0.0.0.0.0.0.2.0.0.0.ip6.int

8.2.5.0.2.1.e.f.f.f.0.2.0.0.a.0.6.5.2.9.0.0.0.0.0.0.0.0.2.0.0.0.ip6.int name =
vallejo.ipv6.Eng.apex.COM
ip6.int nameserver = space1999.Eng.apex.COM
> exit
```

▼ NIS による IPv6 情報を表示する方法

ypmatch コマンドを実行して NIS で IPv6 情報を表示するには、次のように操作します。

- 自分のアカウントで次のコマンドを入力すると、**NIS** 内の **IPv6** アドレスが表示されます。

  % **ypmatch** *hostname* **hosts ipnodes.***byname*

  指定した *hostname* についての情報が表示されます。

注 – Solaris 10 11/06 よりあとの Oracle Solaris リリースには、ipnodes マップは含まれなくなりました。ipnodes マップの IPv6 機能は、hosts マップで管理されるようになりました。

例 7–18 ypmatch コマンドによる IPv6 アドレスの出力

Solaris 10 11/06 以前のリリースの場合、ipnodes.byname データベースに対して ypmatch 操作を実行すると、結果は次の例のようになります。

```
% ypmatch farhost hosts ipnodes.byname
2001:0db8:3c4d:15:a00:20ff:fe12:5286       farhost
```

▼ ネームサービスに依存しない IPv6 情報を表示する方法

この手順は、Solaris 10 11/06 以前のリリースでのみ使用できます。後続のリリースでは、同じ操作を hosts データベースに対して実行できます。

- 自分のユーザーアカウントで、次のコマンドを入力します。

  % **getent ipnodes** *hostname*

  指定した *hostname* についての情報が表示されます。

例 7–19 ipnodes データベース内の IPv6 情報の表示

次に、getent コマンドからの出力の例を示します。

```
% getent ipnodes vallejo

2001:0db8:8512:2:56:a00:fe87:9aba    myhost myhost
fe80::56:a00:fe87:9aba     myhost myhost
```

◆ ◆ ◆ 第 8 章

TCP/IP ネットワークの管理 (手順)

この章では、TCP/IP ネットワークを管理するためのタスクについて説明します。この章の内容は次のとおりです。

注 – ネットワークインタフェースの監視については、211 ページの「ifconfig コマンドによるインタフェース構成の監視」 を参照してください。

これらのタスクでは、サイトで TCP/IP ネットワークが IPv4 専用またはデュアルスタック IPv4/IPv6 で動作していると仮定します。IPv6 をサイトに実装する予定でいるが、まだ実装していない場合は、次の章を参照してください。

- IPv6 実装を計画する方法については、第 4 章「IPv6 ネットワークの計画 (手順)」を参照してください。
- IPv6 を構成して、デュアルスタックネットワーク環境を作成する方法については、第 7 章「IPv6 ネットワークの構成 (手順)」を参照してください。

主な TCP/IP 管理タスク (タスクマップ)

次の表に、ネットワーク情報の表示など、初期構成後に行うその他のネットワーク管理タスクの一覧を示します。表では、各タスクで実行する内容の説明と、タスクの具体的な実行手順が詳しく説明されている現在のドキュメント内のセクションを示しています。

タスク	説明	参照先
インタフェースについての構成情報を表示します。	システム上にある各インタフェースの現在の構成を判断します。	211 ページの「特定のインタフェースに関する情報を入手する方法」
インタフェースアドレス割り当てを表示します。	ローカルシステム上にあるすべてのインタフェースのアドレス割り当てを判断します。	213 ページの「インタフェースアドレスの割り当てを表示する方法」
プロトコル別の統計を表示します。	特定のシステム上におけるネットワークプロトコルのパフォーマンスを監視します。	216 ページの「プロトコル別の統計情報を表示する方法」
ネットワークのステータスを表示します。	すべてのソケットおよび経路制御テーブルのエントリを表示して、システムを管理します。IPv4 の inet アドレスファミリと IPv6 の inet6 アドレスファミリも表示されます。	219 ページの「ソケットのステータスを表示する方法」
ネットワークインタフェースのステータスを表示します。	ネットワークインタフェースのパフォーマンスを監視します。転送の問題を解決するときに役立ちます。	218 ページの「ネットワークインタフェースのステータスを表示する方法」
パケット転送のステータスを表示します。	ネットワークで送信されるパケットの状態を監視します。	220 ページの「特定のアドレスタイプのパケット転送に関するステータスを表示する方法」
IPv6 関連コマンドの出力表示を制御します。	`ping` コマンド、`netstat` コマンド、`ifconfig` コマンド、`traceroute` コマンドの出力を制御します。`inet_type` という名前のファイルを作成します。そのファイル内の `DEFAULT_IP` 変数を設定します。	224 ページの「IP 関連コマンドの表示出力を制御する方法」
ネットワークトラフィックを監視します。	`snoop` コマンドを使用して、すべての IP パケットを表示します。	231 ページの「IPv6 ネットワークトラフィックを監視する方法」

タスク	説明	参照先
ネットワークのルーターが知っているすべてのルートをトレースします。	traceroute コマンドを使用して、すべてのルートを表示します。	227 ページの「すべてのルートをトレースする方法」

ifconfig コマンドによるインタフェース構成の監視

ifconfig コマンドを使用して、IP アドレスを手動でインタフェースに割り当て、インタフェースのパラメータを手動で構成します。さらに、Oracle Solaris 起動スクリプトは ifconfig コマンドを実行して、6to4 トンネルエンドポイントなどの疑似インタフェースを構成します。

このドキュメントでは、ifconfig コマンドのさまざまなオプションを使用する作業が数多く含まれています。ifconfig コマンド、そのオプション、およびその変数の詳細については、ifconfig(1M) のマニュアルページを参照してください。ifconfig の基本構文は次のとおりです。

ifconfig *interface [protocol-family]*

▼ 特定のインタフェースに関する情報を入手する方法

ifconfig コマンドを使用して、特定システムのインタフェースに関する基本情報を判断します。たとえば、単純な ifconfig 照会は、次の内容を返します。

- システム上にあるすべてのインタフェースのデバイス名
- これらのインタフェースに割り当てられているすべての IPv4 アドレス と、もしあれば、すべての IPv6 アドレス
- これらのインタフェースが現在構成されているかどうか

次の手順に、ifconfig コマンドを使用して、システムのインタフェースについての基本構成情報を取得する方法を示します。

1. ローカルホスト上で、**Primary Administrator** の役割を引き受けるか、スーパーユーザーになります。

   Primary Administrator 役割には、Primary Administrator プロファイルが含まれます。役割を作成してユーザーに役割を割り当てるには、『Oracle Solaris の管理: 基本管理』の第 2 章「Solaris 管理コンソールの操作 (タスク)」を参照してください。

2. 特定のインタフェースについての情報を取得します。

   # ifconfig *interface*

ifconfig コマンドからの出力の書式は次のとおりです。

- ステータス行

  ifconfig コマンド出力の 1 行目には、そのインタフェースに現在関連付けられているインタフェース名とステータスフラグが表示されます。ステータス行には、特定のインタフェースとインデックス番号に構成されている最大転送単位 (MTU) も表示されます。状態行を使用すると、インタフェースの現在の状態を判断できます。

- IP アドレス情報行

  ifconfig 出力の 2 行目には、インタフェースに構成されている IPv4 アドレスまたは IPv6 アドレスが表示されます。IPv4 アドレスの場合、構成されているネットマスクとブロードキャストアドレスも表示されます。

- MAC アドレス行

  ifconfig コマンドをスーパーユーザーまたはそれと同等な役割で実行した場合、ifconfig 出力には 3 行目が表示されます。IPv4 アドレスの場合、3 行目には、インタフェースに割り当てられている MAC アドレス (Ethernet 層アドレス) が表示されます。IPv6 アドレスの場合、3 行目には、IPv6 の in.ndpd デーモンが MAC アドレスから生成したリンクローカルアドレスが表示されます。

例 8–1 ifconfig コマンドからのインタフェース基本情報

次の例に、ifconfig コマンドを使用して、特定のホスト上にある eri インタフェースについての情報を取得する方法を示します。

```
# ifconfig eri
eri0: flags=863<UP,BROADCAST,RUNNING,MULTICAST,IPv4> mtu 1500 index 1
      inet 10.0.0.112 netmask ffffff80 broadcast 10.8.48.127
      ether 8:0:20:b9:4c:54
```

次の表では、ifconfig による照会での変数情報、画面上での変数の表示形式、および、提供される情報の種類について説明しています。前述の出力を例として使用します。

変数	画面出力	説明
インタフェース名	eri0	ifconfig コマンドでステータスが要求されたインタフェースのデバイス名を示します。
インタフェースのステータス	flags=863<UP	インタフェースのステータスを表示します。そのインタフェースに現在関連するフラグがすべて表示されます。ここで、インタフェースが現在起動されているか (UP) または起動されていないか (DOWN) を判断できます。

変数	画面出力	説明
ブロードキャストのステータス	BROADCAST	インタフェースが IPv4 ブロードキャストをサポートすることを示します。
転送のステータス	RUNNING	システムがパケットをインタフェース経由で転送していることを示します。
マルチキャストのステータス	MULTICAST, IPv4	インタフェースがマルチキャスト転送をサポートすることを示します。この例のインタフェースは IPv4 マルチキャスト転送をサポートします。
最大転送単位	mtu 1500	当該インタフェースの最大転送サイズが 1500 オクテットであることを示します。
IP アドレス	inet 10.0.0.112	インタフェースに割り当てられている IPv4 アドレスまたは IPv6 アドレスを表示します。この例のインタフェース eri0 は IPv4 アドレス 10.0.0.112 を持っています。
ネットマスク	netmask ffffff80	特定のインタフェースの IPv4 ネットマスクを表示します。IPv6 アドレスはネットマスクを使用しません。
MAC アドレス	ether 8:0:20:b9:4c:54	インタフェースの Ethernet 層アドレスを表示します。

▼ インタフェースアドレスの割り当てを表示する方法

ルーターとマルチホームホストは複数のインタフェースを持っており、多くの場合、各インタフェースには複数の IP アドレスが割り当てられています。ifconfig コマンドを使用すると、システムの特定のインタフェースに割り当てられているすべてのアドレスを表示できます。また、ifconfig コマンドを使用すると、IPv4 アドレスまたは IPv6 アドレスのどちらか一方の割り当てだけを表示できます。さらに、インタフェースの MAC アドレスを表示するには、まず、スーパーユーザーでログインするか、適切な役割になる必要があります。

ifconfig コマンドの詳細については、ifconfig(1M) のマニュアルページを参照してください。

1. ローカルシステムで「ネットワーク管理者」役割になるか、スーパーユーザーになります。

   役割には、認証と特権コマンドが含まれます。役割の詳細については、『Solaris のシステム管理: セキュリティーサービス』の「RBAC の構成 (タスクマップ)」を参照してください。

2. すべてのインタフェースについての情報を取得します。

ifconfig -a コマンドのバリエーションを使用すると、次のことができます。

- システム上にあるすべてのインタフェースのすべてのアドレスを表示します。

  # ifconfig -a
- システム上にあるすべてのインタフェースに割り当てられているすべての IPv4 アドレスを表示します。

  # ifconfig -a4
- ローカルシステムが、IPv6 が有効である場合、システム上にあるすべてのインタフェースに割り当てられているすべての IPv6 アドレスを表示します。

  ifconfig -a6

例 8–2 すべてのインタフェースについてのアドレス指定情報の表示

次の例に、プライマリネットワークインタフェース qfe0 だけを持つホスト用のエントリを示します。それにもかかわらず、ifconfig 出力を見ると、インタフェース qfe0 には現在、次の 3 つの書式のアドレスが割り当てられています。 つまり、ループバック (lo0)、IPv4 (inet)、および IPv6 (inet6) です。この出力では、IPv6 セクションのインタフェース qfe0 の行に IPv6 リンクローカルアドレスが表示されています。qfe0 の 2 番目のアドレスは qfe0:1 行に表示されます。

```
% ifconfig -a
lo0: flags=1000849 <UP,LOOPBACK,RUNNING,MULTICAST,IPv4> mtu 8232 index 1
        inet 127.0.0.1 netmask ff000000
qfe0: flags=1004843 <UP,BROADCAST,RUNNING,MULTICAST,IPv4> mtu 1500 index 2
        inet 10.0.0.112 netmask ffffff80 broadcast 10.0.0.127
        ether 8:0:20:b9:4c:54
lo0: flags=2000849 <UP,RUNNING,MULTICAST,IPv6> mtu 8252 index 1
        inet6 ::1/128
qfe0: flags=2000841 <UP,RUNNING,MULTICAST,IPv6> mtu 1500 index 2
        ether 8:0:20:b9:4c:54
        inet6 fe80::a00:20ff:feb9:4c54/10
qfe0:1: flags=2080841 <UP,RUNNING,MULTICAST,ADDRCONF,IPv6> mtu 1500 index 2
        inet6 2001:db8:3c4d:48:a00:20ff:feb9:4c54/64
```

例 8–3 すべての IPv4 インタフェースについてのアドレス指定情報の表示

次の例に、マルチホームホストに構成されている IPv4 アドレスを示します。この書式の ifconfig コマンドを実行するために、スーパーユーザーとしてログインする必要はありません。

```
% ifconfig -a4
lo0: flags=1000849 <UP,LOOPBACK,RUNNING,MULTICAST,IPv4> mtu 8232 index 1
        inet 127.0.0.1 netmask ff000000
qfe0: flags=1004843 <UP,BROADCAST,RUNNING,MULTICAST,IPv4> mtu 1500 index 2
        inet 10.0.0.112 netmask ffffff80 broadcast 10.0.0.127
        ether 8:0:20:b9:4c:54
qfe1: flags=1004843 <UP,BROADCAST,RUNNING,MULTICAST,IPv4> mtu 1500 index 2
        inet 10.0.0.118 netmask ffffff80 broadcast 10.0.0.127
        ether 8:0:20:6f:5e:17
```

例 8–4 すべての IPv6 インタフェースについてのアドレス指定情報の表示

次の例に、特定のホストに構成されている IPv6 アドレスだけを示します。この書式の ifconfig コマンドを実行するために、スーパーユーザーとしてログインする必要はありません。

```
% ifconfig -a6
lo0: flags=2000849 <UP,LOOPBACK,RUNNING,MULTICAST,IPv6> mtu 8252 index 1
        inet6 ::1/128
qfe0: flags=2000841 <UP,RUNNING,MULTICAST,IPv6> mtu 1500 index 2
        ether 8:0:20:b9:4c:54
        inet6 fe80::a00:20ff:feb9:4c54/10
qfe0:1: flags=2080841 <UP,RUNNING,MULTICAST,ADDRCONF,IPv6> mtu 1500 index 2
        inet6 2001:db8:3c4d:48:a00:20ff:feb9:4c54/64
```

この ifconfig コマンドの出力を見ると、ホスト上にある単一のインタフェースには、次の 3 つの書式の IPv6 アドレスが割り当てられていることがわかります。

`lo0`
: IPv6 ループバックアドレス。

`inet6 fe80::a00:20ff:feb9:4c54/10`
: プライマリネットワークインタフェースに割り当てられているリンクローカルアドレス。

`inet6 2001:db8:3c4d:48:a00:20ff:feb9:4c54/64`
: IPv6 アドレス (サブネット接頭辞を含む)。出力にある ADDRCONF というキーワードは、このアドレスがホストによって自動構成されたことを示します。

netstat コマンドによるネットワークのステータスの監視

netstat コマンドは、ネットワークのステータスとプロトコル統計を表示します。TCP、SCTP、および UDP の各エンドポイントのステータスは表形式で表示できます。経路制御テーブル情報やインタフェース情報も表示できます。

netstat コマンドは、さまざまな種類のネットワークデータを表示します。表示するデータはコマンド行オプションで選択できます。この表示は、特にシステム管理に役立ちます。次に、netstat コマンドの基本構文を示します。

`netstat [-m] [-n] [-s] [-i | -r] [-f` *address-family*`]`

このセクションでは、netstat コマンドで最も一般的に使用されるオプションについて説明します。netstat のすべてのオプションの詳細については、netstat(1M) のマニュアルページを参照してください。

▼ プロトコル別の統計情報を表示する方法

netstat の -s オプションは、UDP、TCP、SCTP、ICMP、および IP のプロトコルについて、プロトコル別の統計情報を表示します。

注 – netstat コマンドからの出力は、Oracle Solaris ユーザーアカウントで取得できます。

- プロトコルのステータスを表示します。

```
$ netstat -s
```

例 8–5 ネットワークプロトコルの統計

次の例に、netstat -s コマンドの出力を示します。出力の一部は省略されています。この出力は、プロトコルが問題を持っている場所を示すことがあります。たとえば、ICMPv4 と ICMPv6 からの統計情報は、このプロトコルがどこにエラーを検出したかを示します。

```
RAWIP
        rawipInDatagrams    =  4701     rawipInErrors       =     0
        rawipInCksumErrs    =     0     rawipOutDatagrams   =     4
        rawipOutErrors      =     0

UDP
        udpInDatagrams      = 10091     udpInErrors         =     0
        udpOutDatagrams     = 15772     udpOutErrors        =     0

TCP     tcpRtoAlgorithm     =     4     tcpRtoMin           =   400
        tcpRtoMax           = 60000     tcpMaxConn          =    -1
        .
        .
        tcpListenDrop       =     0     tcpListenDropQ0     =     0
        tcpHalfOpenDrop     =     0     tcpOutSackRetrans   =     0

IPv4    ipForwarding        =     2     ipDefaultTTL        =   255
        ipInReceives        =300182     ipInHdrErrors       =     0
        ipInAddrErrors      =     0     ipInCksumErrs       =     0
        .
        .
        ipsecInFailed       =     0     ipInIPv6            =     0
        ipOutIPv6           =     3     ipOutSwitchIPv6     =     0

IPv6    ipv6Forwarding      =     2     ipv6DefaultHopLimit =   255
        ipv6InReceives      = 13986     ipv6InHdrErrors     =     0
        ipv6InTooBigErrors  =     0     ipv6InNoRoutes      =     0
        .
        .
        rawipInOverflows    =     0     ipv6InIPv4          =     0

       ipv6OutIPv4         =     0     ipv6OutSwitchIPv4   =     0
```

```
ICMPv4  icmpInMsgs          = 43593     icmpInErrors        =     0
        icmpInCksumErrs     =     0     icmpInUnknowns      =     0
        .
        .
        icmpInOverflows     =     0

ICMPv6  icmp6InMsgs         = 13612     icmp6InErrors       =     0
        icmp6InDestUnreachs =     0     icmp6InAdminProhibs =     0
        .
        .
        icmp6OutGroupQueries=     0     icmp6OutGroupResps  =     2
        icmp6OutGroupReds   =     0

IGMP:
      12287 messages received
          0 messages received with too few bytes
          0 messages received with bad checksum
      12287 membership queries received
SCTP  sctpRtoAlgorithm     =  vanj
      sctpRtoMin           =  1000
      sctpRtoMax           = 60000
      sctpRtoInitial       =  3000
      sctpTimHearBeatProbe =     2
      sctpTimHearBeatDrop  =     0
      sctpListenDrop       =     0
      sctpInClosed         =     0
```

▼ 転送プロトコルのステータスを表示する方法

netstat コマンドを使用すると、転送プロトコルのステータスを表示できます。詳細については、netstat(1M) のマニュアルページを参照してください。

1. システム上の **TCP** 転送プロトコルと **SCTP** 転送プロトコルのステータスを表示します。

   ```
   $ netstat
   ```

2. システム上の特定の転送プロトコルのステータスを表示します。

   ```
   $ netstat -P transport-protocol
   ```

   transport-protocol 変数の値は、 tcp、sctp、または udp です。

例 8–6 TCP 転送プロトコルと SCTP 転送プロトコルのステータスの表示

次の例に、基本的な netstat コマンドの出力を示します。IPv4 専用の情報が表示されています。

```
$ netstat

TCP: IPv4
   Local Address      Remote Address    Swind Send-Q  Rwind Recv-Q      State
-------------------- -------------------- ----- ------  ----- ------     -------
```

```
lhost-1.login      abc.def.local.Sun.COM.980 49640      0     49640    0 ESTABLISHED
lhost-1.login      ghi.jkl.local.Sun.COM.1020 49640     1     49640    0 ESTABLISHED
remhost-1.1014     mno.pqr.remote.Sun.COM.nfsd 49640    0     49640    0 TIME_WAIT
SCTP:
Local Address    Remote Address  Swind  Send-Q  Rwind  Recv-Q StrsI/O  State
---------------- --------------  -----  ------ ------ ------  ------   -------
 *.echo            0.0.0.0           0       0 102400      0   128/1   LISTEN
 *.discard         0.0.0.0           0       0 102400      0   128/1   LISTEN
 *.9001            0.0.0.0           0       0 102400      0   128/1   LISTEN
```

例 8–7 特定の転送プロトコルのステータスの表示

次の例に、netstat コマンドに -P オプションを指定したときの結果を示します。

```
$ netstat -P tcp

TCP: IPv4
   Local Address     Remote Address    Swind Send-Q  Rwind Recv-Q      State
---------------- -------------------- ----- ------  ----- ------     -------
lhost-1.login      abc.def.local.Sun.COM.980 49640      0     49640    0 ESTABLISHED
lhost.login        ghi.jkl.local.Sun.COM.1020 49640     1     49640    0 ESTABLISHED
remhost.1014       mno.pqr.remote.Sun.COM.nfsd 49640    0     49640    0 TIME_WAIT

TCP: IPv6
 Local Address    Remote Address        Swind Send-Q Rwind Recv-Q   State If
---------------- ---------------------- ------ ----- ------ ----------- -----
localhost.38983   localhost.32777       49152      0 49152      0 ESTABLISHED
localhost.32777   localhost.38983       49152      0 49152      0 ESTABLISHED
localhost.38986   localhost.38980       49152      0 49152      0 ESTABLISHED
```

▼ ネットワークインタフェースのステータスを表示する方法

netstat コマンドの i オプションは、ローカルシステムに構成されているネットワークインタフェースの状態を表示します。このオプションを使用すると、各ネットワーク上で送受信しているパケット数がわかります。

- ネットワーク上にあるインタフェースのステータスを表示します。

  ```
  $ netstat -i
  ```

例 8–8 ネットワークインタフェースのステータスの表示

次の例に、ホストのインタフェースを通る IPv4 と IPv6 のパケットフローのステータスを示します。

たとえば、サーバーについて表示される入力パケットカウント (Ipkts) はクライアントがブートを試みるたびに増加しているのに、出力パケットカウント (Opkts) が変化しないことがあります。これは、サーバーがクライアントからのブート要求パケットを見ていることを意味します。しかし、サーバーはそれらのパケットに応答

する方法を知りません。この混乱は、hosts データベース、ipnodes データベース、または ethers データベース内に誤ったアドレスがあることが原因であると考えられます。

しかし、入力パケットカウントが長時間にわたり変化しない場合は、マシンがパケットをまったく見ていません。この場合は、上記と違って、ハードウェアの問題の可能性が高くなります。

```
Name  Mtu  Net/Dest      Address        Ipkts  Ierrs Opkts  Oerrs Collis Queue
lo0   8232 loopback      localhost      142    0     142    0     0      0
hme0  1500 host58        host58        1106302 0     52419  0     0      0

Name  Mtu  Net/Dest      Address                    Ipkts  Ierrs Opkts  Oerrs Collis
lo0   8252 localhost     localhost                   142    0     142    0     0
hme0  1500 fe80::a00:20ff:feb9:4c54/10 fe80::a00:20ff:feb9:4c54 1106305 0 52422 0  0
```

▼ ソケットのステータスを表示する方法

netstat コマンドの -a オプションを使用すると、ローカルホスト上にあるソケットのステータスを表示できます。

- 次のコマンドを入力すると、ソケットのステータスと経路制御テーブルエントリのステータスを表示できます。

  この netstat コマンドのオプションは、ユーザーアカウントで使用できます。

  ```
  % netstat -a
  ```

例 8–9 すべてのソケットと経路制御テーブルエントリの表示

netstat -a コマンドの出力には、膨大な統計が含まれます。次の例に、典型的な netstat -a コマンドの出力の一部を示します。

```
UDP: IPv4
   Local Address         Remote Address     State
-------------------- -------------------- -------
      *.bootpc                            Idle
host85.bootpc                             Idle
      *.*                                 Unbound
      *.*                                 Unbound
      *.sunrpc                            Idle
      *.*                                 Unbound
      *.32771                             Idle
      *.sunrpc                            Idle
      *.*                                 Unbound
      *.32775                             Idle
      *.time                              Idle
        .
        .
      *.daytime                           Idle
      *.echo                              Idle
```

```
      *.discard                            Idle

UDP: IPv6
   Local Address                 Remote Address                  State      If
------------------------------ ------------------------------ ---------- -----
      *.*                                                     Unbound
      *.*                                                     Unbound
      *.sunrpc                                                Idle
      *.*                                                     Unbound
      *.32771                                                 Idle
      *.32778                                                 Idle
      *.syslog                                                Idle
      .
      .
TCP: IPv4
   Local Address        Remote Address    Swind Send-Q Rwind Recv-Q  State
-------------------- -------------------- ----- ------ ----- ------ -------
      *.*                  *.*                0      0 49152      0 IDLE
localhost.4999             *.*                0      0 49152      0 LISTEN
      *.sunrpc             *.*                0      0 49152      0 LISTEN
      *.*                  *.*                0      0 49152      0 IDLE
      *.sunrpc             *.*                0      0 49152      0 LISTEN
      .
      .
      *.printer            *.*                0      0 49152      0 LISTEN
      *.time               *.*                0      0 49152      0 LISTEN
      *.daytime            *.*                0      0 49152      0 LISTEN
      *.echo               *.*                0      0 49152      0 LISTEN
      *.discard            *.*                0      0 49152      0 LISTEN
      *.chargen            *.*                0      0 49152      0 LISTEN
      *.shell              *.*                0      0 49152      0 LISTEN
      *.shell              *.*                0      0 49152      0 LISTEN
      *.kshell             *.*                0      0 49152      0 LISTEN
      *.login
       .
       .
           *.*                0      0 49152      0 LISTEN
   *TCP: IPv6
 Local Address            Remote Address        Swind Send-Q Rwind Recv-Q   State If
---------------------- ---------------------- ----- ------ ----- ------    ----
   *.*                      *.*                  0      0 49152      0      IDLE
   *.sunrpc                 *.*                  0      0 49152      0      LISTEN
   *.*                      *.*                  0      0 49152      0      IDLE
   *.32774                  *.*                  0      0 49152
```

▼ 特定のアドレスタイプのパケット転送に関するステータスを表示する方法

`netstat` コマンドの `-f` オプションを使用すると、特定のアドレスファミリのパケット転送に関する統計を表示できます。

- **IPv4** パケットまたは **IPv6** パケットの転送に関する統計を表示します。

  ```
  $ netstat -f inet | inet6
  ```

IPv4 パケット転送に関する情報を表示するには、netstat -f の引数として inet を指定します。IPv6 パケット転送に関する情報を表示するには、netstat -f の引数として inet6 を指定します。

例 8–10 IPv4 パケット転送のステータス

次に、netstat - f inet コマンドの出力例を示します。

```
TCP: IPv4
   Local Address        Remote Address    Swind Send-Q Rwind Recv-Q  State
-------------------- -------------------- ----- ------ ----- ------ -------
host58.734         host19.nfsd       49640      0 49640      0 ESTABLISHED
host58.38063       host19.32782      49640      0 49640      0 CLOSE_WAIT
host58.38146       host41.43601      49640      0 49640      0 ESTABLISHED
host58.996         remote-host.login 49640      0 49206      0 ESTABLISHED
```

例 8–11 IPv6 パケット転送のステータス

次に、netstat - f inet6 コマンドの出力例を示します。

```
TCP: IPv6
 Local Address          Remote Address        Swind Send-Q Rwind Recv-Q   State    If
------------------ ------------------------- ----- ------ ----- ------ --------- -----
localhost.38065         localhost.32792       49152   0 49152      0    ESTABLISHED
localhost.32792         localhost.38065       49152   0 49152      0    ESTABLISHED
localhost.38089         localhost.38057       49152   0 49152      0    ESTABLISHED
```

▼ 既知のルートのステータスを表示する方法

netstat コマンドの -r オプションは、ローカルホストの経路制御テーブルを表示します。このテーブルには、ホストが知っているすべてのルートのステータスが表示されます。netstat の r オプションは、ユーザーアカウントで実行できます。

- **IP 経路制御テーブルを表示します。**

```
$ netstat -r
```

例 8–12 netstat コマンドによる経路制御テーブルの出力

次に、netstat - r コマンドの出力例を示します。

```
Routing Table: IPv4
  Destination           Gateway           Flags  Ref   Use   Interface
-------------------- -------------------- ----- ----- ------ ---------
host15               myhost               U         1  31059  hme0
10.0.0.14            myhost               U         1      0  hme0
default              distantrouter        UG        1      2  hme0
localhost            localhost            UH        42019361  lo0
```

```
Routing Table: IPv6
  Destination/Mask            Gateway                   Flags Ref   Use   If
--------------------------- --------------------------- ----- --- ------ -----
2002:0a00:3010:2::/64    2002:0a00:3010:2:1b2b:3c4c:5e6e:abcd U  1      0 hme0:1
fe80::/10                fe80::1a2b:3c4d:5e6f:12a2    U       1     23 hme0
ff00::/8                 fe80::1a2b:3c4d:5e6f:12a2    U       1      0 hme0
default                  fe80::1a2b:3c4d:5e6f:12a2    UG      1      0 hme0
localhost                localhost                   UH      9  21832 lo0
```

次の表では、netstat -r コマンドの画面出力の各種パラメータの意味について説明します。

パラメータ	説明
送信先 Destination/Mask	ルートの宛先エンドポイントであるホストを指定します。IPv6 経路制御テーブルには、6to4 トンネルのエンドポイントの接頭辞 (2002:0a00:3010:2::/64) がルートの宛先エンドポイントとして示されていることに注目してください。
Gateway	パケットの転送に使用するゲートウェイを指定します。
Flags	ルートの現在のステータスを示します。U フラグはルートが up 状態であること、G フラグはルートがゲートウェイへのものであることを示します。
Use	送信したパケットの数を示します。
Interface	転送元のエンドポイントである、ローカルホスト上の特定のインタフェースを示します。

ping コマンドによるリモートホストの検証

ping コマンドを使用すると、リモートホストのステータスを判断できます。ping を実行すると、ICMP プロトコルは、指定されたホストにデータグラムを送って、応答を求めます。ICMP は、TCP/IP ネットワーク上のエラー処理を担当するプロトコルです。ping を使用すると、指定したリモートホストに IP 接続が存在するかどうかを判断できます。

次に、ping の基本構文を示します。

/usr/sbin/ping *host [timeout]*

この構文において、*host* はリモートホストの名前です。省略可能な *timeout* 引数は、ping コマンドがリモートホストに到達しようと試行する秒数を示します。デフォルトは 20 秒です。構文とオプションの詳細については、ping(1M) のマニュアルページを参照してください。

▼ リモートホストが動作しているかを確認する方法

- 次の書式の **ping** コマンドを使用します。

  ```
  $ ping hostname
  ```

  ホスト *hostname* が ICMP 転送を受け入れる場合、次のメッセージが表示されます。

  ```
  hostname is alive
  ```

  このメッセージは、*hostname* が ICMP の要求に応答したことを示します。*hostname* がダウン状態にあるかまたは ICMP パケットを受け取れなかった場合は、ping コマンドから次の応答が返されます。

  ```
  no answer from hostname
  ```

▼ ホストでパケットが失われていないかを確認する方法

-ping コマンドの s オプションを使用すると、リモートホストは動作しているが、パケットが失われているかどうかを判断できます。

- 次の書式の **ping** コマンドを使用します。

  ```
  $ ping -s hostname
  ```

例 8–13 パケットの消失を検出するための ping 出力

ping -s *hostname* コマンドは、割り込み文字が送信されるまで、あるいは、タイムアウトが発生するまで、指定されたホストにパケットを送信し続けます。 画面上には次のように出力されます。

```
& ping -s host1.domain8
PING host1.domain8 : 56 data bytes
64 bytes from host1.domain8.COM (172.16.83.64): icmp_seq=0. time=1.67 ms
64 bytes from host1.domain8.COM (172.16.83.64): icmp_seq=1. time=1.02 ms
64 bytes from host1.domain8.COM (172.16.83.64): icmp_seq=2. time=0.986 ms
64 bytes from host1.domain8.COM (172.16.83.64): icmp_seq=3. time=0.921 ms
64 bytes from host1.domain8.COM (172.16.83.64): icmp_seq=4. time=1.16 ms
64 bytes from host1.domain8.COM (172.16.83.64): icmp_seq=5. time=1.00 ms
64 bytes from host1.domain8.COM (172.16.83.64): icmp_seq=5. time=1.980 ms

^C

----host1.domain8  PING Statistics----
7 packets transmitted, 7 packets received, 0% packet loss
round-trip (ms)  min/avg/max/stddev = 0.921/1.11/1.67/0.26
```

パケットの消失という統計は、ホストがパケットを失っているかどうかを示します。ping が失敗する場合、ifconfig コマンドと netstat コマンドからの報告を使用

して、ネットワークのステータスをチェックします。詳細については、211 ページの「ifconfig コマンドによるインタフェース構成の監視」と215 ページの「netstat コマンドによるネットワークのステータスの監視」を参照してください。

ネットワークステータス表示の管理と記録

次の作業に、一般的なネットワークコマンドを使用して、ネットワークのステータスをチェックする方法を示します。

▼ IP 関連コマンドの表示出力を制御する方法

netstat コマンドと ifconfig コマンドの出力を制御すると、IPv4 情報だけを表示したり、IPv4 と IPv6 の両方の情報を表示したりできます。

1 **/etc/default/inet_type** ファイルを作成します。

2 ネットワークの要求に基づいて、次のエントリのうちの 1 つを **/etc/default/inet_type** ファイルに追加します。

- IPv4 情報だけを表示するには、次のように入力します。

  ```
  DEFAULT_IP=IP_VERSION4
  ```

- IPv4 情報と IPv6 情報を表示するには、次のいずれかを入力します。

  ```
  DEFAULT_IP=BOTH
  ```

  または

  ```
  DEFAULT_IP=IP_VERSION6
  ```

  inet_type ファイルの詳細については、inet_type(4) のマニュアルページを参照してください。

注 – ifconfig コマンドの -4 フラグと -6 フラグの設定は、inet_type ファイルに設定された値より優先します。また、netstat コマンドの -f フラグの設定も、inet_type ファイルに設定された値より優先します。

例 8–14 IPv4 情報と IPv6 情報を選択する出力の制御

- DEFAULT_IP=BOTH 変数または DEFAULT_IP=IP_VERSION6 変数を inet_type ファイルで設定する場合、次の出力が得られます。

```
% ifconfig -a
lo0: flags=1000849 mtu 8232 index 1
        inet 10.10.0.1 netmask ff000000
```

```
qfe0: flags=1000843 mtu 1500 index 2
        inet 10.46.86.54 netmask ffffff00 broadcast 10.46.86.255
        ether 8:0:20:56:a8
lo0: flags=2000849 mtu 8252 index 1
        inet6 ::1/128
qfe0: flags=2000841 mtu 1500 index 2
        ether 8:0:20:56:a8
        inet6 fe80::a00:fe73:56a8/10
qfe0:1: flags=2080841 mtu 1500 index 2
        inet6 2001:db8:3c4d:5:a00:fe73:56a8/64
```

- inet_typeファイルで、DEFAULT_IP=IP_VERSION4 変数を定義すると、次の出力が得られます。

```
% ifconfig -a
lo0: flags=849 mtu 8232
        inet 10.10.0.1 netmask ff000000
qfe0: flags=843 mtu 1500
        inet 10.46.86.54 netmask ffffff00 broadcast 10.46.86.255
        ether 8:0:20:56:a8
```

▼ IPv4 経路制御デーモンの活動を記録する方法

IPv4 ルーティングデーモン routed の動作が疑わしい場合、このデーモンの活動をトレースするログを開始できます。routed デーモンを起動すると、このログにはすべてのパケット転送が記録されます。

1. ローカルホスト上で、**Primary Administrator** の役割を引き受けるか、スーパーユーザーになります。

   Primary Administrator 役割には、Primary Administrator プロファイルが含まれます。役割を作成してユーザーに役割を割り当てるには、『Oracle Solaris の管理: 基本管理』の第 2 章「Solaris 管理コンソールの操作 (タスク)」を参照してください。

2. ルーティングデーモンの活動のログファイルを作成します。

   ```
   # /usr/sbin/in.routed /var/log-file-name
   ```

注意 – ビジー状態のネットワークでは、このコマンドによりほとんど絶え間なく出力が生じることがあります。

例 8–15 in.routed デーモンのネットワークログ

次の例に、225 ページの「IPv4 経路制御デーモンの活動を記録する方法」の手順で作成したログの開始部分を示します。

```
-- 2003/11/18 16:47:00.000000 --
Tracing actions started
RCVBUF=61440
Add interface lo0  #1   127.0.0.1      -->127.0.0.1/32
```

```
   <UP|LOOPBACK|RUNNING|MULTICAST|IPv4> <PASSIVE>
Add interface hme0 #2   10.10.48.112    -->10.10.48.0/25
    <UP|BROADCAST|RUNNING|MULTICAST|IPv4>
turn on RIP
Add    10.0.0.0        -->10.10.48.112      metric=0  hme0  <NET_SYN>
Add    10.10.48.85/25  -->10.10.48.112      metric=0  hme0  <IF|NOPROP>
```

▼ IPv6 近傍検索デーモンの活動をトレースする方法

IPv6 の `in.ndpd` デーモンの動作が疑わしい場合、このデーモンの活動をトレースするログを開始できます。中断されるまで、トレースの結果は標準出力に表示されます。`in.ndpd` デーモンを起動すると、このトレースにはすべてのパケット転送が記録されます。

1. **IPv6 のローカルノード上で、Primary Administrator の役割を引き受けるか、スーパーユーザーになります。**

   Primary Administrator 役割には、Primary Administrator プロファイルが含まれます。役割を作成してユーザーに役割を割り当てるには、『Oracle Solaris の管理: 基本管理』の第 2 章「Solaris 管理コンソールの操作 (タスク)」を参照してください。

2. **`in.ndpd` デーモンのトレースを起動します。**

   ```
   # /usr/lib/inet/in.ndpd -t
   ```

3. **トレースを終了するには、Ctrl-C を押します。**

例 8–16 `in.ndpd` デーモンのトレース

次の例に、`in.ndpd` のトレースの開始部分を示します。

```
# /usr/lib/inet/in.ndpd -t
Nov 18 17:27:28 Sending solicitation to  ff02::2 (16 bytes) on hme0
Nov 18 17:27:28         Source LLA: len 6 <08:00:20:b9:4c:54>
Nov 18 17:27:28 Received valid advert from fe80::a00:20ff:fee9:2d27 (88 bytes) on hme0
Nov 18 17:27:28         Max hop limit: 0
Nov 18 17:27:28         Managed address configuration: Not set
Nov 18 17:27:28         Other configuration flag: Not set
Nov 18 17:27:28         Router lifetime: 1800
Nov 18 17:27:28         Reachable timer: 0
Nov 18 17:27:28         Reachable retrans timer: 0
Nov 18 17:27:28         Source LLA: len 6 <08:00:20:e9:2d:27>
Nov 18 17:27:28         Prefix: 2001:08db:3c4d:1::/64
Nov 18 17:27:28                 On link flag:Set
Nov 18 17:27:28                 Auto addrconf flag:Set
Nov 18 17:27:28                 Valid time: 2592000
Nov 18 17:27:28                 Preferred time: 604800
Nov 18 17:27:28         Prefix: 2002:0a00:3010:2::/64
Nov 18 17:27:28                 On link flag:Set
Nov 18 17:27:28                 Auto addrconf flag:Set
Nov 18 17:27:28                 Valid time: 2592000
Nov 18 17:27:28                 Preferred time: 604800
```

traceroute コマンドによる経路制御情報の表示

traceroute コマンドは、IP パケットが通るリモートシステムまでのルートをトレースします。traceroute の技術的な詳細については、traceroute(1M) のマニュアルページを参照してください。

traceroute コマンドを使用すると、経路制御の誤構成や経路制御パスの異常を発見できます。特定のホストが到達不可能な場合には、traceroute を使用して、パケットがどの経路をたどってリモートホストに到達し、どこで障害が起きている可能性があるかを調べることができます。

また、traceroute コマンドは、経路に沿った各ゲートウェイのターゲットホストとの間の往復時間も表示します。この情報は、2 つのホスト間のどこでトラフィックが遅くなっているかを分析する際に利用できます。

▼ リモートホストまでのルートを発見する方法

- 次のコマンドを入力すると、リモートホストまでのルートを発見できます。

  % **traceroute** *destination-hostname*

  この書式の traceroute コマンドは、ユーザーアカウントで使用できます。

例 8–17 traceroute コマンドによるリモートホストまでのルートの表示

次の traceroute コマンドからの出力に、パケットがローカルシステム nearhost からリモートシステム farhost まで通る 7 ホップパスを示します。また、パケットが各ホップを通過する時間も示します。

```
istanbul% traceroute farhost.faraway.com
    traceroute to farhost.faraway.com (172.16.64.39), 30 hops max, 40 byte packets
     1  frbldg7c-86 (172.16.86.1)  1.516 ms  1.283 ms  1.362 ms
     2  bldg1a-001 (172.16.1.211)  2.277 ms  1.773 ms  2.186 ms
     3  bldg4-bldg1 (172.16.4.42)  1.978 ms  1.986 ms  13.996 ms
     4  bldg6-bldg4 (172.16.4.49)  2.655 ms  3.042 ms  2.344 ms
     5  ferbldg11a-001 (172.16.1.236)  2.636 ms  3.432 ms  3.830 ms
     6  frbldg12b-153 (172.16.153.72)  3.452 ms  3.146 ms  2.962 ms
     7  sanfrancisco (172.16.64.39)  3.430 ms  3.312 ms  3.451 ms
```

▼ すべてのルートをトレースする方法

この手順では、traceroute コマンドの -a オプションを使用して、すべてのルートをトレースします。

- ローカルシステムで次のコマンドを入力します。

  % **traceroute -a***host-name*

この書式の traceroute コマンドは、ユーザーアカウントで使用できます。

例 8–18 デュアルスタックホストまでのすべてのルートのトレース

次の例に、デュアルスタックホストまでの考えられるルートをすべて示します。

```
% traceroute -a v6host.remote.com
traceroute: Warning: Multiple interfaces found; using 2::56:a0:a8 @ eri0:2
traceroute to v6host (2001:db8:4a3b::102:a00:fe79:19b0),30 hops max, 60 byte packets
 1  v6-rout86 (2001:db8:4a3b:56:a00:fe1f:59a1)  35.534 ms  56.998 ms *
 2  2001:db8::255:0:c0a8:717  32.659 ms  39.444 ms *
 3  farhost.faraway.COM (2001:db8:4a3b::103:a00:fe9a:ce7b)  401.518 ms  7.143 ms *
 4  distant.remote.com (2001:db8:4a3b::100:a00:fe7c:cf35)  113.034 ms  7.949 ms *
 5  v6host (2001:db8:4a3b::102:a00:fe79:19b0)  66.111 ms *  36.965 ms

traceroute to v6host.remote.com  (192.168.10.75),30 hops max,40 byte packets
 1  v6-rout86 (172.16.86.1)  4.360 ms  3.452 ms  3.479 ms
 2  flrmpj17u.here.COM (172.16.17.131)  4.062 ms  3.848 ms  3.505 ms
 3  farhost.farway.com (10.0.0.23)  4.773 ms *  4.294 ms
 4  distant.remote.com (192.168.10.104)  5.128 ms  5.362 ms *
 5  v6host  (192.168.15.85)  7.298 ms  5.444 ms *
```

snoop コマンドによるパケット転送の監視

snoop コマンドを使用すると、データ転送の状態を監視できます。snoop コマンドは、ネットワークパケットを取り込んで、その内容を指定された書式で表示します。取得したパケットについては、そのまま表示することも、ファイルに保存することも可能です。snoop が中間ファイルに書き込む場合、トレースのビジー状態でパケットロスはほとんど発生しません。そのあと、snoop 自体はファイルの解釈に使用されます。

デフォルトのインタフェースにおいて、パケットをプロミスキュアスモードで取り込むには、ネットワーク管理者役割になるか、スーパーユーザーになる必要があります。サマリー形式では、snoop は最高レベルのプロトコルに関連するデータだけを表示します。たとえば NFS パケットでは、NFS 情報のみが表示されます。RPC、UDP、IP、および Ethernet のフレーム情報は抑止されますが、verbose (詳細表示) オプションのいずれかを選択してあれば表示できます。

頻繁かつ定期的に snoop を使用して、システムが正常に動作している場合の状態を把握してください。最近の白書や RFC を参照したり、NFS や NIS といった特定分野の専門家からアドバイスを受けたりするのも、パケットの分析に役立ちます。snoop とそのオプションの使用法については、snoop(1M) のマニュアルページを参照してください。

▼ すべてのインタフェースからのパケットをチェックする方法

1. ローカルホスト上で、ネットワーク管理者役割になるか、スーパーユーザーになります。

   役割には、認証と特権コマンドが含まれます。役割の詳細については、『Solaris のシステム管理: セキュリティーサービス』の「RBAC の構成 (タスクマップ)」を参照してください。

2. システムに接続されているインタフェースについての情報を出力します。

   ```
   # ifconfig -a
   ```

   snoop コマンドは通常、最初の非ループバックデバイス (通常はプライマリネットワークインタフェース) を使用します。

3. **Example 8–19** に示すように、例 **8–19** コマンドを引数なしで入力して、パケットの取り込みを開始します。

4. **Ctrl-C** キーを押してプロセスを停止します。

例 8–19 snoop コマンドの出力

基本の snoop コマンドは、デュアルスタックホストに対して、次のような出力を返します。

```
% snoop
Using device /dev/hme (promiscuous mode)
router5.local.com -> router5.local.com ARP R 10.0.0.13, router5.local.com is
    0:10:7b:31:37:80
router5.local.com -> BROADCAST     TFTP Read "network-confg" (octet)
farhost.remote.com -> myhost       RLOGIN C port=993
myhost ->   nisserve2          NIS C MATCH 10.0.0.64 in ipnodes.byaddr
nisserve2 ->    myhost             NIS R MATCH No such key
blue-112 -> slave-253-2        NIS C MATCH 10.0.0.112 in ipnodes.byaddr
myhost -> DNSserver.local.com      DNS C 192.168.10.10.in-addr.arpa. Internet PTR ?
DNSserver.local.com  myhost        DNS R 192.168.10.10.in-addr.arpa. Internet PTR
   niserve2.
.
.
.
fe80::a00:20ff:febb:e09 -> ff02::9 RIPng R (5 destinations)
```

この出力に取り込まれたパケットはリモートログインの様子を示しています。この中には、アドレス解決のための NIS サーバーと DNS サーバーへの問い合わせが含まれます。また、ローカルルーターからの定期的な ARP パケットや、IPv6 リンクローカルアドレスから `in.ripngd` への通知も含まれます。

▼ snoop の出力をファイルに取り込む方法

1 ローカルホスト上で、ネットワーク管理者役割になるか、スーパーユーザーになります。

役割には、認証と特権コマンドが含まれます。役割の詳細については、『Solaris のシステム管理: セキュリティーサービス』の「RBAC の構成 (タスクマップ)」を参照してください。

2 **snoop** セッションをファイルに取り込みます。

```
# snoop -o filename
```

次に例を示します。

```
# snoop -o /tmp/cap
Using device /dev/eri (promiscuous mode)
30 snoop: 30 packets captured
```

この例では、30 個のパケットが `/tmp/cap` というファイルに取り込まれています。ディスク容量が十分にあれば、ファイルはどのディレクトリにでも格納できます。取り込んだパケットの数はコマンド行に表示され、Ctrl-C を押せばいつでも終了できます。

snoop 自体によってホストマシン上にネットワーク負荷がかかるので、結果に誤差が生じる場合があります。実際の結果を表示するには、第 3 のシステムから snoop を実行します。

3 **snoop** 出力取り込みファイルを検査します。

```
# snoop -i filename
```

例 8–20 snoop 出力取り込みファイルの内容

次に、`snoop -i` コマンドから返される出力など、さまざまな取り込みの例を示します。

```
# snoop -i /tmp/cap
1   0.00000 fe80::a00:20ff:fee9:2d27 -> fe80::a00:20ff:fecd:4375
    ICMPv6 Neighbor advertisement
...
10  0.91493    10.0.0.40 -> (broadcast)  ARP C Who is 10.0.0.40, 10.0.0.40 ?
34  0.43690 nearserver.here.com  -> 224.0.1.1  IP  D=224.0.1.1 S=10.0.0.40 LEN=28,
      ID=47453, TO =0x0, TTL=1
35  0.00034  10.0.0.40 -> 224.0.1.1    IP  D=224.0.1.1 S=10.0.0.40 LEN=28, ID=57376,
     TOS=0x0, TTL=47
```

▼ IPv4 サーバー/クライアント間のパケットを確認する方法

1. **snoop** を実行するシステムから、クライアントまたはサーバーのいずれかに接続されたハブを外します。

   この第 3 のシステム (snoop システム) はサーバーとクライアント間のすべてのトラフィックを監視するので、snoop のトレースには実際のネットワーク上の状態が反映されます。

2. **snoop** システム上で、ネットワーク管理者役割になるか、スーパーユーザーになります。

   役割には、認証と特権コマンドが含まれます。役割の詳細については、『Solaris のシステム管理: セキュリティーサービス』の「RBAC の構成 (タスクマップ)」を参照してください。

3. **snoop** をオプションなしで入力して、その出力をファイルに保存します。

4. 出力を検査および解釈します。

   snoop 取り込みファイルの詳細については、「RFC 1761, Snoop Version 2 Packet Capture File Format (`http://www.ietf.org/rfc/rfc1761.txt?number=1761`)」を参照してください。

▼ IPv6 ネットワークトラフィックを監視する方法

snoop コマンドを使用すると、IPv6 パケットだけを表示できます。

1. ローカルノード上で、ネットワーク管理者役割になるか、スーパーユーザーになります。

   役割には、認証と特権コマンドが含まれます。役割の詳細については、『Solaris のシステム管理: セキュリティーサービス』の「RBAC の構成 (タスクマップ)」を参照してください。

2. **IPv6** パケットを取り込みます。

   ```
   # snoop ip6
   ```

   snoop コマンドの詳細については、snoop(1M) のマニュアルページを参照してください。

例 8–21 IPv6 ネットワークトラフィックだけの表示

次に、あるノード上で `snoop ip6` コマンドを実行したときに返される典型的な出力の例を示します。

```
# snoop ip6
fe80::a00:20ff:fecd:4374 -> ff02::1:ffe9:2d27 ICMPv6 Neighbor solicitation
fe80::a00:20ff:fee9:2d27 -> fe80::a00:20ff:fecd:4375 ICMPv6 Neighbor
      solicitation
fe80::a00:20ff:fee9:2d27 -> fe80::a00:20ff:fecd:4375 ICMPv6 Neighbor
      solicitation
fe80::a00:20ff:febb:e09 -> ff02::9      RIPng R (11 destinations)
fe80::a00:20ff:fee9:2d27 -> ff02::1:ffcd:4375 ICMPv6 Neighbor solicitation
```

デフォルトアドレス選択の管理

Oracle Solaris では、単一のインタフェースに複数の IP アドレスを付与することができます。たとえば、ネットワーク多重パス (IPMP) のような技術を使用すると、複数のネットワークインタフェースカード (NIC) が同じ IP リンク層に接続できます。このようなリンクは 1 つまたは複数の IP アドレスを持つことができます。さらに、IPv6 が有効なシステム上のインタフェースは、1 つの IPv6 リンクローカルアドレス、少なくとも 1 つの IPv6 経路制御アドレス、および (少なくとも 1 つのインタフェースに) 1 つの IPv4 アドレスを持ちます。

システムがトランザクションを起動すると、アプリケーションは `getaddrinfo` ソケットへの呼び出しを作成します。`getaddrinfo` は、宛先システム上で使用されている可能なアドレスを発見します。そのあと、カーネルはこのリストに優先度を付けて、パケットに使用するのに最適な宛先を見つけます。このプロセスのことを「宛先アドレス順番付け」と呼びます。そのあと、Oracle Solaris カーネルは、パケットに最適な宛先アドレスに対して、適切なソースアドレスの書式を選択します。このプロセスのことを「アドレス選択」と呼びます。宛先アドレス順番付けの詳細については、`getaddrinfo(3SOCKET)` のマニュアルページを参照してください。

IPv4 専用システムとデュアルスタック IPv4/IPv6 システムは両方とも、デフォルトアドレス選択を実行する必要があります。ほとんどの状況では、デフォルトアドレス選択メカニズムを変更する必要はありません。しかし、IPMP をサポートしたり、6to4 アドレス書式を選択したりする場合は、アドレス書式の優先度を変更する必要があります。

▼ IPv6 アドレス選択ポリシーテーブルを管理する方法

次の手順では、アドレス選択ポリシーテーブルを変更する方法について説明します。IPv6 デフォルトアドレス選択の概念については、277 ページの「`ipaddrsel` コマンド」を参照してください。

注意 – 次のタスクに示す理由がない場合は、IPv6 アドレス選択ポリシーテーブルを変更しないでください。このポリシーテーブルを間違って変更すると、ネットワーク上で問題が発生する可能性があります。次の手順に示すように、このポリシーテーブルは必ずバックアップを保存してください。

1 **Primary Administrator** 役割を引き受けるか、スーパーユーザーになります。

Primary Administrator 役割には、Primary Administrator プロファイルが含まれます。役割を作成してユーザーに役割を割り当てるには、『Oracle Solaris の管理: 基本管理』の第 2 章「Solaris 管理コンソールの操作 (タスク)」を参照してください。

2 現在の **IPv6** アドレス選択ポリシーテーブルを調査します。

```
# ipaddrsel
# Prefix                  Precedence Label
::1/128                           50 Loopback
::/0                              40 Default
2002::/16                         30 6to4
::/96                             20 IPv4_Compatible
::ffff:0.0.0.0/96                 10 IPv4
```

3 デフォルトアドレス選択ポリシーテーブルのバックアップを作成します。

```
# cp /etc/inet/ipaddrsel.conf /etc/inet/ipaddrsel.conf.orig
```

4 テキストエディタを使用して、**/etc/inet/ipaddrsel.conf** を自分用にカスタマイズします。

`/etc/inet/ipaddrsel` のエントリには、次の構文を使用します。

prefix/prefix-length precedence label [# comment]

次に、デフォルトアドレス選択ポリシーテーブルに一般的に行われる変更の例を示します。

- 6to4 アドレスに最高の優先度を付ける場合。

```
2002::/16                       50 6to4
::1/128                         45 Loopback
```

6to4 アドレス書式の優先度は現在、最高の50 です。Loopback の優先度は、以前は 50 でしたが、現在は 45 です。ほかのアドレス書式の優先度は変わりません。

- 特定の宛先アドレスとの通信において、特定のソースアドレスを使用するように指示する場合。

```
::1/128                         50 Loopback
2001:1111:1111::1/128           40 ClientNet
2001:2222:2222::/48             40 ClientNet
::/0                            40 Default
```

このエントリは、物理インタフェースが 1 つしかないホストの場合に役立ちます。ここで、`2001:1111:1111::1/128` は、ネットワーク `2001:2222:2222::/48` 内にある宛先に向けられたすべてのパケットのソースアドレスとして優先されま

す。優先度 40 は、このインタフェースに構成されたほかのアドレス書式よりも、ソースアドレス 2001:1111:1111::1/128 を優先することを指示します。

- IPv6 アドレスよりも IPv4 アドレスを優先する場合。

```
::ffff:0.0.0.0/96               60 IPv4
::1/128                         50 Loopback
.
.
```

このテーブルでは、IPv4 書式 ::ffff:0.0.0.0/96 の優先度をデフォルトの 10 からテーブル内で最高の 60 に変更しています。

5 変更したポリシーテーブルをカーネルにロードします。

```
ipaddrsel -f /etc/inet/ipaddrsel.conf
```

6 変更したポリシーテーブルに問題がある場合は、**IPv6** デフォルトアドレス選択ポリシーテーブルを復元します。

```
# ipaddrsel -d
```

▼ 現在のセッションだけの IP6 アドレス選択テーブルを変更する方法

/etc/inet/ipaddrsel.conf ファイルを編集すると、その変更はリブート後も適用されます。変更したポリシーテーブルを現在のセッションだけに適用したい場合、次の手順に従います。

1 **Primary Administrator** 役割を引き受けるか、スーパーユーザーになります。

Primary Administrator 役割には、Primary Administrator プロファイルが含まれます。役割を作成してユーザーに役割を割り当てるには、『Oracle Solaris の管理: 基本管理』の第 2 章「Solaris 管理コンソールの操作 (タスク)」を参照してください。

2 **/etc/inet/ipaddrsel** の内容を *filename* にコピーします (*filename* は自分が選択した名前)。

```
# cp /etc/inet/ipaddrsel filename
```

3 必要に応じて、*filename* 内のポリシーテーブルを編集します。

4 変更したポリシーテーブルをカーネルにロードします。

```
# ipaddrsel -f filename
```

システムをリブートするまで、カーネルは新しいポリシーテーブルを使用します。

◆ ◆ ◆ 第 9 章

9 ネットワークの問題の障害追跡 (手順)

この章では、ネットワークで発生する一般的な問題の解決方法について説明します。この章の内容は次のとおりです。

ネットワークの問題の障害追跡における新機能

Solaris 10 7/07 では、`/etc/inet/ipnodes` ファイルは廃止されました。個々の手順で説明されているとおり、`/etc/inet/ipnodes` は以前の Solaris 10 リリースにのみ使用してください。

一般的なネットワーク障害追跡について

ネットワークに問題が発生すると、まず、1 つまたは複数のホストで通信の損失が発生するという兆候が見られるようになります。あるホストを初めてネットワークに追加したときに、そのホストがまったく動作しない場合は、構成ファイルのどれかに問題があることが考えられます。また、ネットワークインタフェースカードに問題がある可能性もあります。1 つのホストに突然問題が生じた場合は、ネットワークインタフェースに原因があると考えられます。ネットワーク上のホストが互いに通信できるが、ほかのネットワークとは通信できない場合は、ルーターに原因があると考えられます。 あるいは、ほかのネットワークに原因があるかもしれません。

`ifconfig` コマンドを使用すると、ネットワークインタフェースについての情報を取得できます。`netstat` コマンドを使用すると、経路制御テーブルやプロトコルの統計を表示できます。サードパーティーのネットワーク診断プログラムから、さまざまな障害追跡ユーティリティーが提供されています。詳細は、サードパーティーのドキュメントを参照してください。

ネットワークのパフォーマンスを低下させる問題の原因は、明確にはわかりません。しかし、たとえば、ping のようなツールを使用することで、ホストでのパケットの消失など、問題の原因を突き止めることはできます。

基本的な診断チェックの実行

ネットワークに問題がある場合、一連のソフトウェアチェックを実行すると、基本的なソフトウェア関連の問題は診断および修正できます。

▼ 基本的なネットワークソフトウェアチェックの実行方法

1. ローカルシステムで「ネットワーク管理者」役割になるか、スーパーユーザーになります。

   役割には、認証と特権コマンドが含まれます。役割の詳細については、『Solaris のシステム管理: セキュリティーサービス』の「RBAC の構成 (タスクマップ)」を参照してください。

2. **netstat** コマンドを使用すると、ネットワーク情報を表示できます。

   netstat コマンドの構文と詳細については、215 ページの「netstat コマンドによるネットワークのステータスの監視」と netstat(1M) のマニュアルページを参照してください。

3. **hosts** データベース (および、**Solaris 10 11/06** 以前のリリースで **IPv6** を使用している場合は **ipnodes** データベース) をチェックして、エントリが正しくて最新であることを確認します。

   /etc/inet/hosts データベースについては、244 ページの「hosts データベース」と hosts(4) のマニュアルページを参照してください。/etc/inet/ipnodes データベースについては、247 ページの「ipnodes データベース」と ipnodes(4) のマニュアルページを参照してください。

4. 逆アドレス解決プロトコル (**RARP**) を実行している場合、**ethers** データベースの **Ethernet** アドレスをチェックして、エントリが正しくて最新であることを確認します。

5. **telnet** コマンドを使用して、ローカルホストに接続してみます。

   telnet コマンドの構文と詳細については、telnet(1) のマニュアルページを参照してください。

6 ネットワークデーモン **inetd** が動作していることを確認します。

```
# ps -ef | grep inetd
```

次の出力で、inetd デーモンが動作していることを確認します。

```
root 57 1 0 Apr 04 ? 3:19 /usr/sbin/inetd -s
```

7 **IPv6** がネットワーク上で有効な場合、**IPv6** デーモン **in.ndpd** が動作していることを確認します。

```
# ps -ef | grep in.ndpd
```

次の出力で、in.ndpd デーモンが動作していることを確認します。

```
root 123  1 0  Oct 27 ?  0:03 /usr/lib/inet/in.ndpd
```

IPv6 を配備するときの一般的な問題

このセクションでは、サイトに IPv6 を計画および配備しているときに遭遇する可能性のある一般的な問題について説明します。実際の計画タスクについては、第 4 章「IPv6 ネットワークの計画 (手順)」を参照してください。

IPv4 ルーターを IPv6 用にアップグレードできない

既存の装置をアップグレードできない場合、IPv6 に対応した装置を購入するしかない場合もあります。装置に付属するドキュメントを参照して、IPv6 をサポートするために行う必要がある、装置固有の手順があるかどうかを調べてください。

IPv6 サポート用にアップグレードできない IPv4 ルーターもあります。この状況が自分のトポロジに適応する場合、IPv6 ルーターが IPv4 ルーターの隣にくるように物理的に配線します。このようにすれば、IPv6 ルーターから IPv4 ルーター経由でトンネルできます。トンネルを構成する手順については、195 ページの「IPv6 サポート用にトンネルを構成するためのタスク (タスクマップ)」を参照してください。

サービスを IPv6 用にアップグレードしたあとの問題

サービスを IPv6 サポート用に準備しているとき、次のような状況に遭遇する場合があります。

- あるアプリケーションを IPv6 用に移植したのに、IPv6 サポートがデフォルトで有効にならない場合。このようなアプリケーションは、IPv6 が有効になるように構成する必要があります。
- 複数のサービスを実行するサーバーにおいて、IPv4 専用のサービスと IPv4 と IPv6 両用のサービスが混在している場合、次のような状況に遭遇します。クライアントがこれら両方の種類のサービスを使用する必要がある場合、サーバー側で混乱が生じます。

現在の ISP が IPv6 をサポートしない

IPv6 を配備したいが、現在の ISP が IPv6 アドレス指定を提供しない場合、ISP を変更するのではなく、次の代替方法を考えてみてください。

- 別の ISP から IPv6 通信用に 2 番目の回線 ISP を購入します。この解決方法には、高い費用がかかります。
- 「仮想 *ISP*」を取得します。仮想 ISP はサイトに IPv6 接続を提供しますが、実際の回線は提供しません。その代わりに、サイトから IPv4 ISP 経由で仮想 ISP に到達するトンネルを作成します。
- 自分のサイトから ISP 経由でほかの IPv6 サイトに到達する 6to4 トンネルを使用します。あるアドレスに対して、6to4 ルーターの登録済み IPv4 アドレスを、IPv6 アドレスの公開トポロジ部分として使用します。

6to4 リレールーターへのトンネルを作成するときのセキュリティー問題

本来、6to4 ルーターと 6to4 リレールーター間のトンネルは安全ではありません。これらのルーター間のトンネルには、次のようなセキュリティー問題が内在しています。

- 6to4 リレールーターはパケットのカプセル化とカプセル化の解除を行いますが、パケット内に含まれるデータのチェックは行いません。
- アドレスのスプーフィングは、6to4 リレールーターとの間で構築されるトンネルにおける際立った問題です。着信トラックについては、6to4 ルーターはリレールーターの IPv4 アドレスを送信元の IPv6 アドレスと対応させることができないという問題があります。このため、IPv6 ホストのアドレスは簡単にスプーフィングされかねません。6to4 リレールーターのアドレスもスプーフィングの可能性があります。
- デフォルトの設定では、6to4 ルーターと 6to4 リレールーター間に信頼できるメカニズムは存在しません。したがって、6to4 ルーターは 6to4 リレールーターが信頼できるものであるかどうかを識別できず、正規の 6to4 リレールーターであるかすら確認できません。このようなことから、6to4 サイトと宛先の IPv6 サイト間に信頼関係が存在していることか、あるいは攻撃を受けるという可能性を両サイトとも受け入れることが求められます。

これらの問題を始めとする 6to4 リレールーターのセキュリティー問題については、Internet Draft『Security Considerations for 6to4』で説明されています。一般には、6to4 リレールーターのサポートは次のような場合だけ検討してください。

- 信頼できるプライベートな IPv6 ネットワークとの間で 6to4 サイトが通信を行う場合。たとえば、独立した 6to4 サイトとネイティブ IPv6 サイトから構成されるキャンパスネットワーク上などでこのサポートを有効にすると便利かもしれません。
- ビジネス上の理由で、6to4 サイトと特定のネイティブ IPv6 ホストとの通信を避けることができない場合。
- Internet Draft『Security Considerations for 6to4』で提唱されている検査と信頼できるモデルを導入した場合。

◆ ◆ ◆ 第 10 章

TCP/IP と IPv4 の詳細 (リファレンス)

この章では、ネットワーク構成ファイルの種類、目的、ファイルエントリの書式など、TCP/IP ネットワークの参照情報を提供します。また、既存のネットワークデータベースについても詳しく説明します。さらにこの章では、定義されているネットワーククラスとサブネット番号に基づいて、IPv4 アドレスが構成される仕組みについても説明します。

この章では、次の内容について説明します。

TCP/IP と IPv4 の新機能の詳細

Solaris 10 7/07 では、/etc/inet/ipnodes ファイルは廃止されました。個々の手順で説明されているとおり、/etc/inet/ipnodes は以前の Solaris 10 リリースにのみ使用してください。

TCP/IP 構成ファイル

ネットワーク上の各システムは、次の TCP/IP 構成ファイルとネットワークデータベースから TCP/IP 構成情報を取得します。

- /etc/hostname.*interface* ファイル
- /etc/nodename ファイル
- /etc/defaultdomain ファイル
- /etc/defaultrouter ファイル (オプション)
- hosts データベース
- Solaris 10 11/06 以前のリリースでは ipnodes データベース

- netmasks データベース (オプション)

Oracle Solaris インストールプログラムは、インストール処理の一環として上記のファイルを作成します。これらのファイルは、この「TCP/IP 構成ファイル」の節の説明に従って手作業で編集することもできます。hosts データベースと netmasks データベースは、Oracle Solaris ネットワークで使用できるネームサービスによって読み取られるネットワークデータベースのうちの 2 つです。ネットワークデータベースの概念についての詳細は、252 ページの「ネットワークデータベースと nsswitch.conf ファイル」を参照してください。Solaris 10 11/06 以前のリリースの ipnodes ファイルについては、247 ページの「ipnodes データベース」を参照してください。

/etc/hostname.*interface* ファイル

このファイルは、ローカルホスト上の物理的なネットワークインタフェースを定義します。ローカルシステムには、少なくとも 1 つの /etc/hostname.*interface* ファイルが存在する必要があります。Oracle Solaris インストールプログラムは、インストールプロセス中に検出された最初のインタフェースに対して /etc/hostname.*interface* ファイルを作成します。このインタフェースは通常、いちばん小さなデバイス番号 (たとえば、eri0) を持っており、プライマリネットワークインタフェースと呼ばれます。インストールプログラムが追加インタフェースを検出した場合は、インストールプロセスの一環としてそれらも任意で構成できます。

注 – 同じインタフェース用に別のホスト名ファイルを作成する場合、そのホスト名ファイルの名前も hostname.[0–9]* の形式 (hostname.qfe0.a123など) にする必要があります。hostname.qfe0.bak や hostname.qfe0.old のような名前は無効であり、システムのブート中にスクリプトに無視されます。

また、特定のインタフェースに対応するホスト名ファイルは 1 つだけにする必要があります。/etc/hostname.qfe と /etc/hostname.qfe.a123 のように、インタフェースの代替ホスト名ファイルを有効なファイル名で作成した場合、ブートスクリプトは両方のホスト名ファイルの内容を参照することによって構成を試みるため、エラーが発生します。これらのエラーを回避するには、特定の構成で使用したくないホスト名ファイルに無効なファイル名を付けます。

インストール後、新しいネットワークインタフェースをシステムに追加した場合は、そのインタフェースに対して、/etc/hostname.*interface* ファイルを作成する必要があります (153 ページの「システムインストール後に物理インタフェースを構成する方法」を参照)。また、Oracle Solaris ソフトウェアが新しいネットワークインタフェースを認識し、使用できるようにするには、インタフェースのデバイスドライバが適切なディレクトリに読み込まれるようにする必要があります。新しいネットワークインタフェースに付属しているドキュメントを参照し、正しいインタフェース名とデバイスドライバの使用方法を確認してください。

基本的な /etc/hostname.*interface* ファイルにはエントリが 1 つだけ含まれます。 それは、ネットワークインタフェースに関連付けられた、ホスト名または IPv4 アドレスのどちらかです。IPv4 アドレスは、一般的な 10 進ドット表記または CIDR 表記で表現できます。/etc/hostname.*interface* ファイルのエントリとしてホスト名を使用する場合は、そのホスト名が /etc/inet/hosts ファイルにも存在している必要があります。

たとえば、tenere というシステムのプライマリネットワークインタフェースが smc0 であるとします。この場合、/etc/hostname.smc0 ファイルのエントリとして、10 進ドット表記または CIDR 表記の IPv4 アドレス、あるいはホスト名 tenere を使用できます。

注 – IPv6 は /etc/hostname6.*interface* ファイルを使用して、ネットワークインタフェースを定義します。詳細は、275 ページの「IPv6 インタフェース構成ファイル」を参照してください。

/etc/nodename ファイル

このファイルにはエントリが 1 つ含まれます。つまり、ローカルシステムのホスト名です。たとえば、timbuktu というシステムでは、/etc/nodename ファイルに timbuktu というエントリが入ります。

/etc/defaultdomain ファイル

このファイルにはエントリが 1 つ含まれます。このエントリは、ローカルホストのネットワークが属する管理ドメインの完全に修飾されたドメイン名です。ネットワーク管理者は、この名前を Oracle Solaris インストールプログラムに指示したり、また後日、このファイルを編集できます。ネットワークドメインについての詳細は、『Solaris のシステム管理 (ネーミングとディレクトリサービス : DNS、NIS、LDAP 編)』を参照してください。

/etc/defaultrouter ファイル

このファイルには、ネットワークに直接接続されている各ルーターのエントリを含めることができます。このエントリは、ネットワーク間のルーターとして機能するネットワークインタフェースの名前です。/etc/defaultrouter ファイルがあるということは、システムが静的経路選択をサポートする構成であることを示します。

hosts データベース

hosts データベースには、ネットワーク上のシステムの IPv4 アドレスとホスト名が含まれています。NIS、DNS、または LDAP ネームサービスを使用している場合、hosts データベースは、ホスト情報用として指定されているデータベースに格納されます。たとえば、NIS を実行するネットワークでは、hosts データベースは hostsbyname ファイルに格納されます。

ネームサービス用にローカルファイルを使用する場合、hosts データベースは /etc/inet/hosts ファイルに格納されます。このファイルには、プライマリネットワークインタフェースのホスト名と IPv4 アドレス、システムに備わっているほかのネットワークインタフェース、このシステムが検査する必要があるほかのネットワークアドレスが含まれています。

注 – BSD ベースのオペレーティングシステムとの互換性を保つため、/etc/hosts ファイルは /etc/inet/hosts へのシンボリックリンクとなっています。

/etc/inet/hosts ファイルの形式

/etc/inet/hosts ファイルは、次のような基本的な構文を使用します。構文についての詳細は、hosts(4) のマニュアルページを参照してください。

IPv4-address hostname [nicknames] [#comment]

IPv4-address	ローカルホストが認識する必要のある各インタフェースの IPv4 アドレスが含まれます。
hostname	設定時にシステムに割り当てたホスト名と、ローカルホストが認識しなければならない増設ネットワークインタフェースに割り当てたホスト名が含まれます。
[nickname]	ホストのニックネームが含まれます (オプション)。
[#comment]	コメント用のフィールドです (オプション)。

初期 /etc/inet/hosts ファイル

Oracle Solaris インストールプログラムをシステムで実行すると、このプログラムは初期 /etc/inet/hosts ファイルを構成します。このファイルには、ローカルホストにとって必要最小限のエントリが含まれています。エントリには、ループバックアドレス、ホストの IPv4 アドレス、ホスト名が入っています。

たとえば、図 5–1 で示されるシステム tenere に対して、Oracle Solaris インストールプログラムは次の /etc/inet/hosts ファイルを作成する可能性があります。

例 10–1　システム tenere 用の /etc/inet/hosts ファイル

```
127.0.0.1     localhost         loghost    #loopback address
192.168.200.3   tenere                      #host name
```

ループバックアドレス

例 10–1 では、IPv4 アドレス 127.0.0.1 はループバックアドレスです。ループバックアドレスは、ローカルシステムがプロセス間通信するために使用する予約済みネットワークインタフェースです。このアドレスを使用して、ホストは自分自身にパケットを送信できます。ifconfig コマンドは、ループバックアドレスを使用して、構成と検査を行います (211 ページの「ifconfig コマンドによるインタフェース構成の監視」を参照)。TCP/IP ネットワークのシステムはすべて、ローカルホスト上での IPv4 ループバックに IP アドレス 127.0.0.1 を使用する必要があります。

ホスト名

IPv4 アドレス 192.168.200.1 と名前 tenere は、ローカルシステムのアドレスとホスト名です。これらは、システムのプライマリネットワークインタフェースに割り当てられます。

複数のネットワークインタフェース

システムには複数のネットワークインタフェースを持つものがあり、これらはルーターまたはマルチホームホストとなります。システムに接続されるネットワークインタフェースごとに、専用の IP アドレスとそれに割り当てる名前が必要です。インストール時には、プライマリネットワークインタフェースを構成する必要があります。特定のシステムがインストール時に複数のインタフェースを持っている場合は、Oracle Solaris インストールプログラムも増設インタフェースに関するプロンプトを表示します。1 つ以上の増設インタフェースをこの時点で構成したり、あとで手動構成を行うことができます。

Oracle Solaris のインストール後、インタフェース情報をシステムの /etc/inet/hosts ファイルに追加することで、ルーターまたはマルチホームホストの増設インタフェースを構成できます。ルーターとマルチホームホストを構成する方法については、124 ページの「IPv4 ルーターの構成」と 133 ページの「マルチホームホストの構成」を参照してください。

例 10–2 は、図 5–1 のシステム timbuktu に関する /etc/inet/hosts ファイルを示しています。

例 10–2　システム timbuktu 用の /etc/inet/hosts ファイル

```
127.0.0.1        localhost     loghost
192.168.200.70   timbuktu      #This is the local host name
192.168.201.10   timbuktu-201  #Interface to network 192.9.201
```

これらの 2 つのインタフェースを使用して、timbuktu はルーターとしてネットワーク 192.168.200 と 192.168.201 に接続します。

ネームサービスの hosts データベースに対する影響

NIS と DNS ネームサービス、および LDAP ディレクトリサービスは、ホスト名とアドレスを 1 つまたは複数のサーバーに格納します。これらのサーバーは、各サーバーのネットワーク上のすべてのホストとルーター (もしあれば) に関する情報を含む hosts データベースを保持しています。これらのサービスの詳細については、『Solaris のシステム管理 (ネーミングとディレクトリサービス : DNS、NIS、LDAP 編)』を参照してください。

ローカルファイルがネームサービスを提供する場合

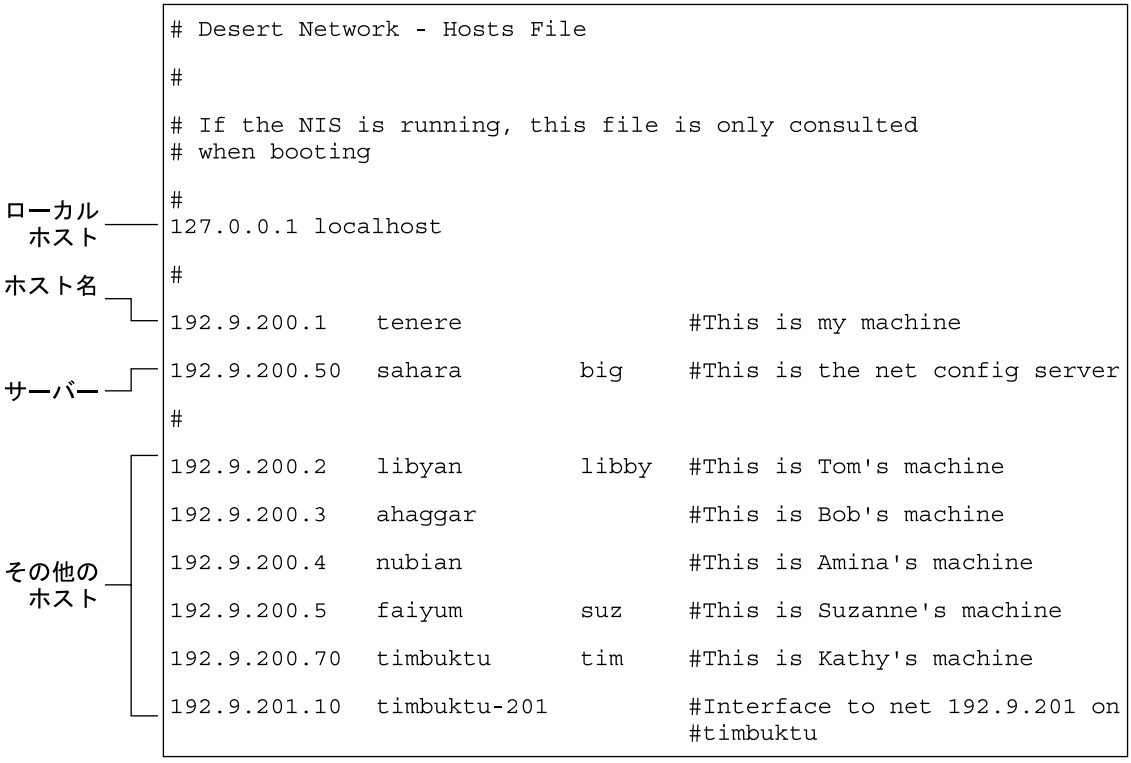
ネームサービスにローカルファイルを使用するネットワークでは、ローカルファイルモードで実行するシステムがそれぞれの /etc/inet/hosts ファイルでネットワーク上のほかのシステムの IPv4 アドレスやホスト名を参照します。したがって、これらのシステムの /etc/inet/hosts ファイルには、次の情報が含まれている必要があります。

- ループバックアドレス
- ローカルシステム (プライマリネットワークインタフェース) の IPv4 アドレスとホスト名
- このシステムに接続している増設ネットワークインタフェース (もしあれば) の IPv4 アドレスとホスト名
- ローカルネットワーク上のすべてのホストの IPv4 アドレスとホスト名
- このシステムが認識する必要のあるルーター (もしあれば) の IPv4 アドレスとホスト名
- このシステムでホスト名を使用して参照したいシステムの IPv4 アドレス

図 10–1 は、システム tenere の /etc/inet/hosts ファイルを示しています。このシステムはローカルファイルモードで実行されます。このファイルには、192.9.200 ネットワーク上のすべてのシステムの IPv4 アドレスとホスト名が含まれています。また、このファイルにはインタフェース名 timbuktu-201 とその IPv4 アドレスが含まれています。このインタフェースは 192.9.200 ネットワークを 192.9.201 ネットワークに接続しています。

ネットワーククライアントとして構成されたシステムは、そのループバックアドレスと IPv4 アドレス用にローカルの /etc/inet/hosts ファイルを使用します。

図 10–1 ローカルファイルモードで実行されるシステム用の /etc/inet/hosts ファイル



ipnodes データベース

注 – Solaris 10 11/06 よりあとのリリースには、ipnodes データベースは含まれなくなりました。これらの後続のリリースでは、ipnodes の IPv6 機能は hosts データベースに移行されます。

/etc/inet/ipnodes ファイルは IPv4 アドレスと IPv6 アドレスの両方を格納します。また、IPv4 アドレスは、一般的な 10 進ドット表記または CIDR 表記として保存できます。このファイルはローカルデータベースとして、ホスト名を IPv4 アドレスや IPv6 アドレスに関連付けます。ホスト名やそのアドレスは、/etc/inet/ipnodes などの静的ファイルには保存しないでください。ただし、テスト目的の場合は IPv4 アドレスを /etc/inet/hosts に保存するのと同じ方法で IPv6 アドレスを保存します。ipnodes ファイルでは、hosts ファイルと同じフォーマット変換を使用します。/etc/inet/hosts の詳細については、244 ページの「hosts データベース」を参照してください。ipnodes ファイルについては、ipnodes(4) のマニュアルページを参照してください。

IPv6 が有効なアプリケーションは /etc/inet/ipnodes データベースを使用します。既存の /etc/hosts データベースには、IPv4 アドレスだけが含まれていますが、既存のアプリケーションの便宜上、このデータベースは変更されません。ipnodes データベースが存在しない場合、IPv6 が有効なアプリケーションは既存の hosts データベースを使用します。

注 – アドレスを追加する必要がある場合、IPv4 アドレスは hosts ファイルと ipnodes ファイルの両方に追加しなければなりません。IPv6 アドレスは ipnodes ファイルにだけ追加します。

例 10–3 /etc/inet/ipnodes ファイル

次の例のように、ホスト名アドレスは、ホスト名でグループにまとめる必要があります。

```
#
# Internet IPv6 host table
# with both IPv4 and IPv6 addresses
#
::1     localhost
2001:db8:3b4c:114:a00:20ff:fe78:f37c   farsite.com farsite farsite-v6
fe80::a00:20ff:fe78:f37c     farsite-11.com farsitell
192.168.85.87                    farsite.com farsite farsite-v4
2001:db8:86c0:32:a00:20ff:fe87:9aba   nearsite.com nearsite nearsite-v6
fe80::a00:20ff:fe87:9aba     nearsite-11.com nearsitell
10.0.0.177                   nearsite.com nearsite nearsite-v4 loghost
```

netmasks データベース

ネットワーク構成の一部として netmasks データベースを編集する必要があるのは、ネットワークのサブネット化を設定している場合だけです。 netmasks データベースは、各ネットワークとそれに対応するサブネットマスクのリストで構成されています。

注 – サブネットを作成するときは、新規の各ネットワークはそれぞれ独立した物理ネットワークであることが必要です。単一の物理ネットワークにサブネット化を適用することはできません。

サブネット化とは

「サブネット化」は、大規模なインターネットワークにおいて、限られた 32 ビットの IPv4 アドレス空間を最大限活用し、経路制御テーブルの大きさを縮小する方法です。どのようなアドレスクラスの場合も、サブネット化によってホストアドレス空

間の一部をネットワークアドレスに割り当て、ネットワーク数を増やすことができます。新規のネットワークアドレスに割り当てられるホストアドレス空間の部分を、「サブネット番号」と言います。

IPv4 アドレス空間を有効活用できることのほかに、サブネット化には管理上の利点もいくつかあります。ネットワークの数が増えるに伴って、経路制御はきわめて複雑になってきます。たとえば、小規模の組織なら、個々のローカルネットワークにクラス C の番号を割り当てることができます。しかし、組織が成長するにつれて、多数の異なるネットワーク番号を管理することは、非常に複雑な作業になってきます。このような場合の改善策の 1 つとして、組織内の主要部門に対してそれぞれクラス B のネットワーク番号を割り当てる方法が考えられます。たとえば、エンジニアリング部門に 1 つのクラス B ネットワーク、運営部門に別のクラス B を割り当てるということが可能です。その上で、サブネット化によって得られたネットワーク番号を使用して、個々のクラス B ネットワークをさらに多くのネットワークに分割できます。これによって、ルーター間でやりとりしなければならない経路制御情報の量も減少します。

IPv4 アドレス用のネットワークマスクの作成

サブネット化プロセスの一環として、ネットワーク全体の「ネットマスク」を選択する必要があります。ネットマスクは、ホストアドレス空間の中で、どの位置の何個のビットがサブネット番号を表し、どの位置の何個のビットがホスト番号を表すかを決定します。完全な IPv4 アドレスは 32 ビットで構成されることを思い出してください。ホストアドレス空間を表すために使用できるビット数は、アドレスクラスによって異なりますが、最大 24 ビット、最小 8 ビットです。ネットマスクは `netmasks` データベース内に指定します。

サブネットの使用を予定している場合は、TCP/IP を構成する前にネットマスクを決定する必要があります。ネットワーク構成の一環としてオペレーティングシステムをインストールすることを予定している場合は、Oracle Solaris インストールプログラムは、ネットワークのネットマスクを指定するよう求めます。

58 ページの「IPv4 アドレス指定スキームの設計」 に説明されているとおり、32 ビットの IP アドレスは、ネットワーク部とホスト部で構成されています。32 ビットは 4 個のバイトに分かれます。各バイトは、ネットワーククラスに応じて、ネットワーク番号かホスト番号のどちらかに割り当てられます。

たとえば、クラス B の IPv4 アドレスでは、左側の 2 バイトがネットワーク番号に割り当てられ、右側の 2 バイトがホスト番号に割り当てられます。クラス B の IPv4 アドレス `172.16.10` では、右側の 2 バイトをホストに割り当てることができます。

サブネット化を行う場合は、ホスト番号に割り当てるバイトの中の一部のビットを、サブネットアドレスとして使用する必要があります。たとえば、ホストアドレス空間が 16 ビットであれば、65,534 個のホストのアドレス指定が可能です。3 番目のバイトをサブネットアドレス用に使用して、4 番目のバイトをホストアドレス用に使

用するとすれば、最大 254 のネットワークのアドレスと、それぞれについて最大 254 ずつのホストのアドレスを指定できます。

ホストアドレスのバイトのどのビットがサブネットアドレスに使用され、どのビットがホストアドレスに使用されるかは、「サブネットマスク」によって決まります。サブネットマスクは、バイト中のどのビットをサブネットアドレス用とするかを選択するために使用します。ネットマスクのビットは連続していなければなりませんが、バイトの境界に整列している必要はありません。

ネットマスクは、ビット単位の論理 AND 演算子を使用することによって、IPv4 アドレスに適用できます。この演算によって、アドレスのネットワーク番号とサブネット番号の位置が選択されます。

ネットマスクは、2 進数で表現できます。2 進数と 10 進数は計算機を使用して換算できます。次の例では、ネットマスクの 10 進数形式と 2 進数形式の両方を示してあります。

ネットマスク `255.255.255.0` を IPv4 アドレス `172.16.41.101` に適用すると、結果の IPv4 アドレスは `172.16.41.0` になります。

`172.16.41.101` & `255.255.255.0` = `172.16.41.0`

2 進数形式では、この演算は次のようになります。

10000001.10010000.00101001.01100101 (IPv4 アドレス)

次の論理積 (AND) をとります。

11111111.11111111.11111111.00000000 (IPv4 ネットマスク)

これでシステムは、ネットワーク番号 `172.16.41` をネットワーク番号 `172.16` の代わりに検索します。ネットワークのネットワーク番号が `172.16.41` である場合、システムはこの番号をチェックし、検出します。IPv4 アドレス空間の 3 番目のバイトには最大 254 個の値を割り当てることができるので、サブネット化によって、254 個のネットワーク用のアドレス空間を作ることができます。サブネット化を使用しなければ、ネットワークは 1 つだけです。

ネットワークを 2 つだけ追加するためのアドレス空間を確保する場合は、次のようなサブネットマスクを使用します。

`255.255.192.0`

このネットマスクの結果は次のようになります。

11111111.11111111.1100000.00000000

この結果、まだ 14 ビットはホストアドレス用に使用できます。全桁 0 と全桁 1 は予約済みなので、少なくとも 2 ビットをホスト番号用として確保する必要があります。

/etc/inet/netmasks ファイル

ネットワークで NIS または LDAP を実行している場合、これらのネームサービスのサーバーに netmasks データベースが格納されます。ローカルファイルをネームサービスとして使用するネットワークの場合、この情報は /etc/inet/netmasks ファイル内に格納されます。

注 – BSD ベースのオペレーティングシステムと互換性を確保するため、/etc/netmasks ファイルは /etc/inet/netmasks へのシンボリックリンクとなっています。

次のコード例に示すのは、クラス B ネットワーク用のサンプルの /etc/inet/netmasks ファイルです。

例 10–4　クラス B ネットワーク用の /etc/inet/netmasks ファイル

```
# The netmasks file associates Internet Protocol (IPv4) address
# masks with IPv4 network numbers.
#
#     network-number    netmask
#
# Both the network-number and the netmasks are specified in
# “decimal dot” notation, e.g:
#
#        128.32.0.0   255.255.255.0
192.168.0.0  255.255.255.0
```

/etc/netmasks ファイルが存在しない場合は、テキストエディタで作成してください。構文は次のとおりです。

network-number　*netmask-number*

詳細は、netmasks(4) のマニュアルページを参照してください。

ネットマスク番号を作成するときは、ISP または Internet Registry から割り当てられたネットワーク番号 (サブネット番号ではない) とネットマスク番号を、/etc/inet/netmasks ファイルに入力します。各サブネットマスクはそれぞれ単独の行に入れてください。

次に例を示します。

```
128.78.0.0        255.255.248.0
```

/etc/inet/hosts ファイルには、ネットワーク番号のシンボリック名も入力できます。こうすることによって、ネットワーク番号の代わりにこれらのネットワーク名をコマンドへのパラメータとして使用できます。

inetd インターネットサービスデーモン

inetd デーモンは、システムの起動時にインターネット標準サービスを起動したり、システムの実行中にサービスを再起動したりできます。SMF (サービス管理機能) は、標準インターネットサービスを変更したり、inetd デーモンに追加サービスを開始させるために使用します。

inetd が起動したサービスを管理するには、次の SMF コマンドを使用します。

svcadm　起動、停止、再開などのサービスの管理操作を行います。詳細は、svcadm(1M) のマニュアルページを参照してください。

svcs　サービスのステータスを照会します。詳細は、svcs(1) のマニュアルページを参照してください。

inetadm　サービスのプロパティーの表示と変更を行います。詳細は、inetadm(1M) のマニュアルページを参照してください。

特定のサービスの inetadm プロファイルの proto フィールドの値は、サービスが実行されるトランスポート層プロトコルを示します。サービスが IPv4 専用の場合、proto フィールドには tcp、udp、または sctp を指定します。

- SMF コマンドの使用方法については、『Oracle Solaris の管理: 基本管理』の「SMF コマンド行管理ユーティリティー」を参照してください。
- SMF コマンドを使用して SCTP で実行されるサービスを追加するタスクについては、142 ページの「SCTP プロトコルを使用するサービスを追加する方法」を参照してください。
- IPv4 要求と IPv6 要求の両方を処理するサービスの追加については、252 ページの「inetd インターネットサービスデーモン」を参照してください。

ネットワークデータベースと nsswitch.conf ファイル

ネットワークデータベースは、ネットワークの構成に必要な情報を提供するファイルです。ネットワークデータベースには次のものがあります。

- hosts
- netmasks
- ethers データベース
- bootparams

- protocols
- services
- networks

構成工程の一環として、ネットワークをサブネット化する場合は、hosts データベースと netmasks データベースを編集します。システムをネットワーククライアントとして構成するには、bootparams と ethers の 2 つのネットワークデータベースを使用します。残りのデータベースはオペレーティングシステムが使用するもので、編集が必要になることはほとんどありません。

nsswitch.conf ファイルはネットワークデータベースではありませんが、このファイルは関連するネットワークデータベースと共に構成する必要があります。nsswitch.conf は、特定のシステムに、 ローカルファイル、NIS、DNS、または LDAP のどのネームサービスを使用するかを指定します。

ネットワークデータベースへのネームサービスの影響

ネットワークデータベースの形式は、ネットワークに選択したネームサービスの種類によって異なります。たとえば、hosts データベースには、少なくとも、ローカルシステムとそのシステムに直接接続されているネットワークインタフェースのホスト名と IPv4 アドレスだけは含まれています。しかし、ネットワークで使用するネームサービスの種類によっては、その他の IPv4 アドレスとホスト名も hosts データベースに含まれていることがあります。

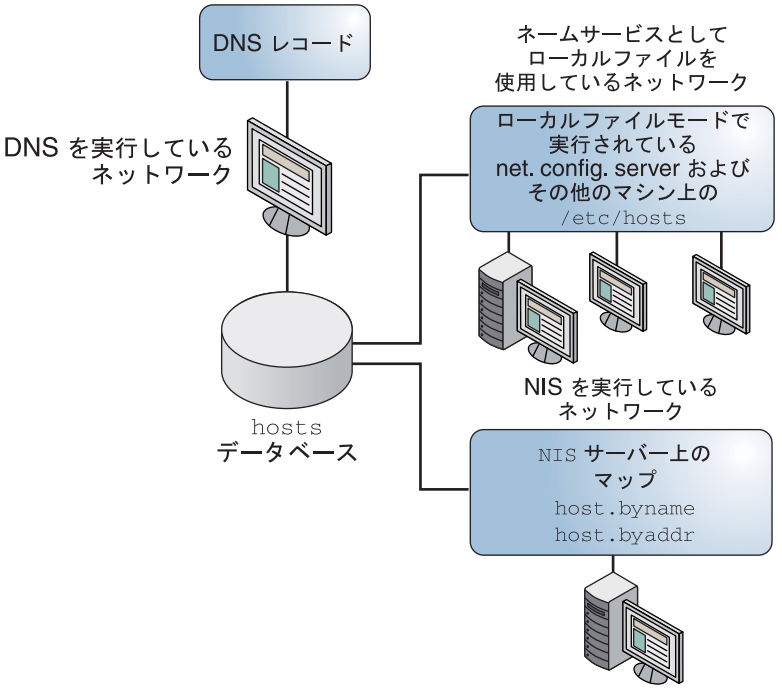
ネットワークデータベースは、次のように使用されます。

- ネームサービスにローカルファイルを使用するネットワークは、/etc/inet ディレクトリと /etc ディレクトリにあるファイルに依存します。
- NIS は、NIS マップと呼ばれるデータベースを使用します。
- DNS は、ホスト情報を含むレコードを使用します。

注 – DNS ブートファイルとデータファイルは、ネットワークデータベースに直接は対応していません。

次の図に、これらのネームサービスで使用する hosts データベースの形式を示します。

図 10–2　ネームサービスが使用する hosts データベースの形式



次の表に、ネットワークデータベースと対応するローカルファイルおよび NIS マップを示します。

注 – Solaris 10 11/06 よりあとの Oracle Solaris リリースでは、ipnodes データベースは削除されます。

表 10–1　ネットワークデータベースと対応ネームサービスファイル

ネットワークデータベース	ローカルファイル	NIS マップ
hosts	/etc/inet/hosts	hosts.byaddr hosts.byname
netmasks	/etc/inet/netmasks	netmasks.byaddr
ethers	/etc/ethers	ethers.byname ethers.byaddr
bootparams	/etc/bootparams	bootparams
protocols	/etc/inet/protocols	protocols.byname protocols.bynumber
services	/etc/inet/services	services.byname
networks	/etc/inet/networks	networks.byaddr networks.byname

このドキュメントでは、ネームサービス用にローカルファイルを使用するネットワークから見たネットワークデータベースについて説明します。

- hosts データベースについては、244 ページの「hosts データベース」に記載されています。
- netmasks データベースについては、248 ページの「netmasks データベース」に記載されています。
- Solaris 10 11/06 以前のリリースの場合、ipnodes データベースについては、247 ページの「ipnodes データベース」に記載されています。

NIS、DNS、および LDAP でのネットワークデータベースの対応関係については、『Solaris のシステム管理 (ネーミングとディレクトリサービス : DNS、NIS、LDAP 編)』を参照してください。

nsswitch.conf ファイル

/etc/nsswitch.conf ファイルは、ネットワークデータベースの検索順序を定義します。Oracle Solaris インストールプログラムは、インストール中にネットワーク管理者が指定するネームサービスに基づいて、ローカルシステム用のデフォルトの /etc/nsswitch.conf ファイルを作成します。「None」オプションを指定して、ローカルファイルをネームサービスとして使用することを指示した場合は、nsswitch.conf ファイルは次の例のようになります。

例 10–5　ネームサービスにファイルを使用するネットワーク用の nsswitch.conf

```
# /etc/nsswitch.files:
#
# An example file that could be copied over to /etc/nsswitch.conf;
# it does not use any naming service.
#
# "hosts:" and "services:" in this file are used only if the
# /etc/netconfig file contains "switch.so" as a
# nametoaddr library for "inet" transports.

passwd:          files
group:           files
hosts:           files
networks:        files
protocols:       files
rpc:             files
ethers:          files
netmasks:        files
bootparams:      files
publickey:       files
# At present there isn't a 'files' backend for netgroup; the
# system will figure it out pretty quickly,
# and won't use netgroups at all.
netgroup:        files
automount:       files
aliases:         files
```

例 10–5　ネームサービスにファイルを使用するネットワーク用の nsswitch.conf　(続き)

```
services:       files
sendmailvars:   files
```

このファイルについての詳細は、nsswitch.conf(4) のマニュアルページを参照してください。基本構文は、次のとおりです。

database name-service-to-search

database フィールドには、オペレーティングシステムが検索する多くの種類のデータベースの 1 つを指定できます。たとえば、passwd や aliases などのようにユーザーに影響を与えるデータベースでも、またネットワークデータベースでも指定できます。ネットワークデータベースの場合、*name-service-to-search* パラメータの値は、files、nis、nis+ のどれかです。hosts データベースの場合は、検索するネームサービスとして dns も値に指定できます。nis+ と files のように、複数のネームサービスも指定できます。

例 10–5 では、表示されている検索オプションは files だけです。したがって、ローカルシステムは、/etc ディレクトリと /etc/inet ディレクトリに入っているファイルから、ネットワークデータベース情報のほか、セキュリティーと自動マウントに関する情報を入手します。

nsswitch.conf の変更

/etc ディレクトリには、Oracle Solaris インストールプログラムが作成する nsswitch.conf ファイルが含まれています。そのほかに、次のネームサービス用のテンプレートファイルも含まれています。

- nsswitch.files
- nsswitch.nis

あるネームサービスから別のネームサービスに変更したい場合は、対応するテンプレートを nsswitch.conf にコピーできます。また、nsswitch.conf ファイルを選択的に編集して、個々のデータベースを見つけるために検索するデフォルトのネームサービスを変更できます。

たとえば、NIS を実行するネットワークでは、ネットワーククライアントについての nsswitch.conf ファイルの変更が必要な場合があります。bootparams データベースと ethers データベースの検索順序では、最初のオプションとして files、次に nis が指定されている必要があります。次のコード例に、正しい検索順序を示します。

例 10–6　NIS を実行するネットワーク上のクライアントのための nsswitch.conf

```
# /etc/nsswitch.conf:#
.
.
```

例 10–6 NIS を実行するネットワーク上のクライアントのための nsswitch.conf (続き)

```
passwd:         files nis
group:          files nis

# consult /etc "files" only if nis is down.
hosts:          nis     [NOTFOUND=return] files
networks:       nis     [NOTFOUND=return] files
protocols:      nis     [NOTFOUND=return] files
rpc:            nis     [NOTFOUND=return] files
ethers:         files   [NOTFOUND=return] nis
netmasks:       nis     [NOTFOUND=return] files
bootparams:     files   [NOTFOUND=return] nis
publickey:      nis
netgroup:       nis

automount:      files nis
aliases:        files nis

# for efficient getservbyname() avoid nis
services:       files nis
sendmailvars:   files
```

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

bootparams データベース

bootparams データベースには、ネットワーククライアントモードでブートされるように構成されたシステムが使用する情報が含まれています。ネットワーククライアントを持つネットワークの場合は、このデータベースの編集が必要になります。手順については、111 ページの「ネットワーククライアントの構成」を参照してください。このデータベースは、/etc/bootparams ファイルに入力した情報をもとにして構築されます。

このデータベースの構文についての詳細は、bootparams(4) のマニュアルページに含まれています。基本構文は、次のとおりです。

system-name file-key-server-name:pathname

個々のディスクレスまたはネットワーククライアントシステムについて、エントリが 1 つずつ含まれています。 各エントリに入っている情報は、クライアント名、キーのリスト、サーバー名、パス名です。各エントリの最初の項目は、クライアントシステムの名前です。最初の項目以外は、すべてオプションです。次に例を示します。

例 10–7 bootparams データベース

```
myclient   root=myserver : /nfsroot/myclient  \
swap=myserver : /nfsswap//myclient \
dump=myserver : /nfsdump/myclient
```

この例の dump= はダンプファイルを探さないようクライアントホストに指示しています。

bootparams のワイルドカードエントリ

ほとんどの場合、ワイルドカードエントリを使用するのは、bootparams データベースを編集してクライアントをサポートするときです。次のようにしてワイルドカードエントリを使用します。

```
*  root=server:/path dump=:
```

アスタリスク (*) ワイルドカードは、このエントリが、bootparams データベース内で明示的に指定されていないすべてのクライアントに適用されることを示します。

ethers データベース

ethers データベースは、/etc/ethers ファイルに入力した情報から構築されます。このデータベースは、ホスト名を「メディアアクセス制御」(MAC) アドレスに関連付けます。ethers データベースの作成が必要になるのは、RARP デーモンを実行する場合だけです。つまり、ネットワーククライアントを構成する場合だけです。

RARP は、このファイルを使用して、MAC アドレスを IP アドレスにマッピングします。RARP デーモン in.rarpd を実行する場合は、ethers ファイルを設定し、このファイルをこのデーモンを実行するすべてのホストに格納して、ネットワークに対する変更が反映されるようにする必要があります。

このデータベースの構文についての詳細は、ethers(4) のマニュアルページに含まれています。基本構文は、次のとおりです。

MAC-address hostname #comment

MAC-address	ホストの MAC アドレス
hostname	ホストの公式名
#comment	ファイルのエントリに追加する注釈

MAC アドレスは装置の製造元から提供されます。システムのブートプロセス中に、システムが MAC アドレスを表示しない場合は、お使いのハードウェアのマニュアルを参照してください。

ethers データベースにエントリを追加するときは、ホスト名が、ニックネームではなく、hosts データベース内と ipnodes データベース内 (Solaris 10 11/06 以前のリリースの場合) の基本名に一致していることを確認してください (次のコード例を参照)。

例 10–8 ethers データベース内のエントリ

```
8:0:20:1:40:16  fayoum
8:0:20:1:40:15  nubian
8:0:20:1:40:7   sahara    # This is a comment
8:0:20:1:40:14  tenere
```

その他のネットワークデータベース

残りのネットワークデータベースについては、編集が必要になることはほとんどありません。

networks データベース

networks データベースは、ネットワーク名をネットワーク番号と関連付けて、いくつかのアプリケーションが番号ではなく名前を使用または表示できるようにします。networks データベースは、/etc/inet/networks ファイルの中の情報に基づいています。このデータベースには、このネットワークがルーターを介して接続されるすべてのネットワークの名前が含まれています。

初期 networks データベースは、Oracle Solaris インストールプログラムが構成します。ただし、既存のネットワークトポロジに新たなネットワークを追加する場合は、このデータベースを更新する必要があります。

/etc/inet/networks の構文についての詳細は、networks(4) のマニュアルページに含まれています。基本形式は、次のとおりです。

network-name network-number nickname(s) #comment

network-name	ネットワークの公式名
network-number	ISP またはインターネットレジストリが割り当てた番号
nickname	そのネットワークの別名
#comment	ファイルのエントリに追加する注釈

networks ファイルは必要に応じて更新する必要があります。netstat プログラムは、このデータベース内の情報を使用してステータステーブルを作成します。

次のコード例に、/etc/networks ファイルのサンプルを示します。

例 10–9 /etc/networks ファイル

```
#ident    "@(#)networks    1.4    92/07/14 SMI"    /* SVr4.0 1.1    */
#
# The networks file associates Internet Protocol (IP) network
# numbers with network names. The format of this file is:
#
```

例 10–9 /etc/networks ファイル (続き)

```
#     network-name              network-number              nicnames . . .

# The loopback network is used only for intra-machine communication
loopback              127

#
# Internet networks
#
arpanet     10       arpa  # Historical
#
# local networks

eng   192.168.9 #engineering
acc   192.168.5 #accounting
prog  192.168.2 #programming
```

protocols データベース

protocols データベースには、システムにインストールされている TCP/IP プロトコルとプロトコル番号の一覧が含まれています。このデータベースは、Oracle Solaris インストールプログラムによって自動的に作成されます。このファイルの管理が必要になることは、ほとんどありません。

このデータベースの詳しい構文については、protocols(4) のマニュアルページを参照してください。次のコード例に、/etc/inet/protocols ファイルのサンプルを示します。

例 10–10 /etc/inet/protocols ファイル

```
#
# Internet (IP) protocols
#
ip    0   IP    # internet protocol, pseudo protocol number
icmp  1   ICMP  # internet control message protocol
tcp   6   TCP   # transmission control protocol
udp  17   UDP   # user datagram protocol
```

services データベース

services データベースは、TCP と UDP サービスの名前および既知のポート番号の一覧を含んでいます。このデータベースは、ネットワークサービスを呼び出すプログラムにより使用されます。Oracle Solaris インストールプログラムは、services データベースを自動的に作成します。通常は、このデータベースの管理作業を必要としません。

構文についての詳細は、services(4) のマニュアルページを参照してください。次に、典型的な /etc/inet/services ファイルからの抜粋を示します。

例 10–11 /etc/inet/サービス ファイル

```
#
# Network services
#
echo      7/udp
echo      7/tcp
echo      7/sctp6
discard   9/udp     sink null
discard   11/tcp
daytime   13/udp
daytime   13/tcp
netstat   15/tcp
ftp-data  20/tcp
ftp       21/tcp
telnet    23/tcp
time      37/tcp    timeserver
time      37/udp    timeserver
name      42/udp    nameserver
whois     43/tcp    nickname
```

Oracle Solaris のルーティングプロトコル

このセクションでは、Oracle Solaris でサポートされている、ルーティング情報プロトコル (RIP) および ICMP ルーター発見 (RDISC) の、2 つのルーティングプロトコルについて説明します。RIP と RDISC は、どちらも標準 TCP/IP プロトコルです。Oracle Solaris で使用できるルーティングプロトコルの完全な一覧については、表 5–1 および表 5–2 を参照してください。

ルーティング情報プロトコル (RIP)

RIP は、システムのブート時に自動的に起動するルーティングデーモンである in.routed によって実装されます。s オプションを指定した in.routed をルーターで実行すると、in.routed は、到達可能なすべてのネットワークへのルートをカーネルルーティングテーブルに組み入れ、すべてのネットワークインタフェースを経由する「到達可能性」を通知します。

ホストで q オプションを指定して実行すると、in.routed はルーティング情報を引き出しますが、到達可能性の通知は行いません。ホストでは、ルーティング情報は次の 2 つの方法で抽出できます。

- S フラグ (大文字の「S」、「省スペースモード」の意) を指定しない。 in.routed は、ルーターで実行するときとまったく同じようにフルルーティングテーブルを作成します。
- S フラグを指定する。 in.routed は、使用可能なルーターについてデフォルトのルートを 1 つずつ含む最小カーネルテーブルを作成します。

ICMP ルーター発見 (RDISC) プロトコル

ホストは、ルーターから経路制御情報を取得するときに、RDISC を使用します。したがって、ホストが RDISC を実行しているとき、各ルーターは、経路制御情報の交換のために、RIP などのような別のプロトコルも実行している必要があります。

RDISC は、ルーターとホストの両方で実行される in.routed によって実装されます。ホストでは、in.routed は RDISC を使用して、RDISC によってホストに通知を行うルーターからデフォルトのルートを検出します。in.routed は、ルーターで RDISC を使用して、直接接続されているネットワーク上のホストにデフォルトのルートを通知します。in.routed(1M) のマニュアルページと gateways(4) のマニュアルページを参照してください。

ネットワーククラス

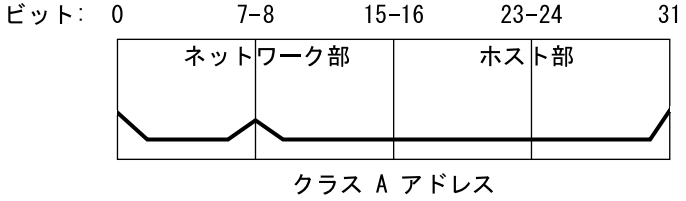
注 – 多くの旧式ネットワークは今もクラスをベースにしていますが、クラスベースのネットワーク番号は IANA から取得できなくなりました。

この節では、IPv4 ネットワーククラスについて詳しく説明します。32 ビットの IPv4 アドレス空間は、ネットワーク部のビット数が多かったり少なかったりするなど、クラスによって使い方が異なります。3 つのクラスとは、クラス A、クラス B、クラス C です。

クラス A ネットワーク番号

クラス A ネットワーク番号は、IPv4 アドレスの最初の 8 ビットを「ネットワーク部」として使用します。残りの 24 ビットは、次の図のように IPv4 アドレスのホスト部を含んでいます。

図 10–3 クラス A アドレスのバイト割り当て

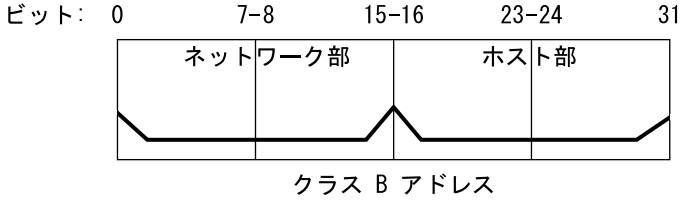


クラス A ネットワーク番号の最初のバイトに割り当てられる値は 0 - 127 の範囲です。IPv4 アドレス 75.4.10.4 について考えてみます。最初のバイトの 75 という値は、このホストがクラス A ネットワーク内にあることを示しています。残りのバイトの 4.10.4 はホストアドレスを形成します。クラス A 番号の最初のバイトだけが IANA で登録されます。残りの 3 バイトをどのように使用するかは、そのネットワーク番号の所有者の自由です。クラス A のネットワークとして存在可能なのは 127 個だけです。この範囲内の各番号が、それぞれ最大 16,777,214 個のホストを収容できます。

クラス B ネットワーク番号

クラス B ネットワーク番号は、16 ビットをネットワーク番号に、16 ビットをホスト番号に使用します。クラスB ネットワーク番号の最初のバイトは、128 - 191 の範囲です。番号 172.16.50.56 では、最初の 2 バイト 172.16 が IANA で登録され、ネットワークアドレスを構成します。残りの 2 バイトの 50.56 にはホストアドレスが含まれ、これはネットワーク番号の所有者が任意に割り当てることができます。次の図に、クラス B のアドレスを示します。

図 10–4 クラス B アドレスのバイト割り当て

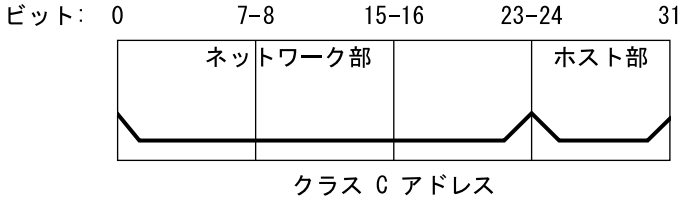


クラス B は通常、ネットワーク上に多くのホストを抱える組織に割り当てられます。

クラス C ネットワーク番号

クラス C ネットワーク番号は、24 ビットをネットワーク番号に、8 ビットをホスト番号に使用します。クラス C ネットワーク番号は、ホストが少ない (最大で 254) ネットワークに適しています。クラス C ネットワーク番号は、IPv4 アドレスの最初の 3 バイトを占めます。ネットワーク番号の所有者が自由に割り当てることができるのは、4 番目のバイトだけです。次の図に、クラス C アドレスのバイトを示します。

図 10–5 クラス C アドレスのバイト割り当て



クラス C ネットワーク番号の最初のバイトは、192 - 223 の範囲です。2 番目と 3 番目のバイトは、それぞれ 1 - 255 の範囲です。クラス C アドレスは通常、`192.168.2.5` のようになります。最初の 3 バイト `192.168.2` はネットワーク番号です。この例の最後のバイト、つまり 5 がホスト番号です。

◆ ◆ ◆ 第 11 章

11 IPv6 の詳細 (リファレンス)

この章では、次の Oracle Solaris の IPv6 実装に関する参照情報について説明します。

IPv6 の概要については、第 3 章「IPv6 の紹介 (概要)」を参照してください。IPv6 対応ネットワークを構成するタスクについては、第 7 章「IPv6 ネットワークの構成 (手順)」を参照してください。

IPv6 の新機能の詳細

Solaris 10 7/07 では、`/etc/inet/ipnodes` ファイルは廃止されました。個々の手順で説明されているとおり、`/etc/inet/ipnodes` は以前の Solaris 10 リリースにのみ使用してください。

IPv6 アドレス指定書式の詳細

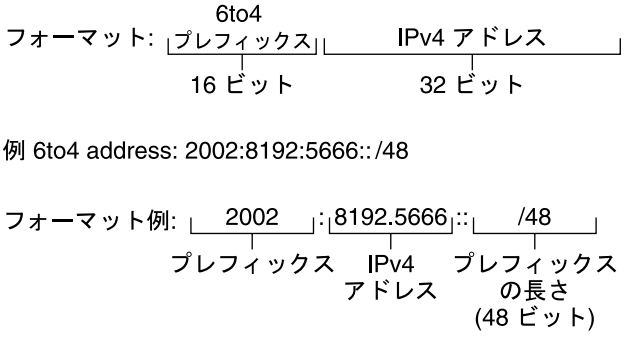
第 3 章「IPv6 の紹介 (概要)」では、ユニキャストサイトアドレスとリンクローカルアドレスというもっとも一般的な IP v 6 アドレス指定書式について紹介します。このセクションでは、第 3 章「IPv6 の紹介 (概要)」で説明しなかった IPv6 アドレス指定の詳細な書式について説明します。

6to4 派生のアドレス指定

ルーターまたはホストエンドポイントから 6to4 トンネルを構成する計画がある場合は、エンドポイントシステム上の `/etc/inet/ndpd.conf` ファイルにある 6to4 サイト接頭辞を通知する必要があります。6to4 トンネルの構成に関する概要とタスクについては、199 ページの「6to4 トンネルを構成する方法」を参照してください。

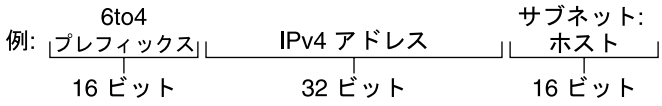
次の図に、6to4 サイト接頭辞の構成を示します。

図 11–1　6to4 サイト接頭辞の構成

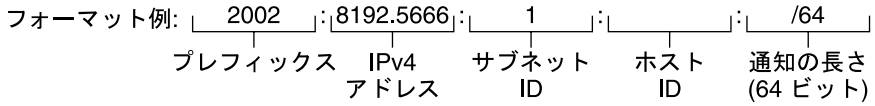


次の図は、`ndpd.conf` ファイルに指定する 6to4 サイトのサブネット接頭辞の構成を示しています。

図 11–2　6to4 サブネット接頭辞の構成



例 6to4 address: 2002:8192.5666:1: :/64



この表では、6to4 サブネット接頭辞を構成する要素と、各要素の長さおよび定義について説明しています。

構成要素	長さ	定義
接頭辞	16 ビット	6to4 接頭辞ラベル 2002 (0x2002)。
IPv4 アドレス	32 ビット	6to4 インタフェースですでに構成されている一意の IPv4 アドレス。通知のために、ドット付きの 10 進表記ではなく 16 進表記の IPv4 アドレスを指定します。
サブネット ID	16 ビット	サブネット ID。これは、6to4 サイトにおけるリンクで一意となる値でなければなりません。

ホストにおける 6to4 派生のアドレス指定

ルーター広告によって 6to4 派生の接頭辞を受信する際、IPv6 ホストはインタフェース上の 6to4 派生アドレスを自動的に構成し直します。アドレスの書式は次のとおりです。

prefix:IPv4-address:subnet-ID:interface-ID/64

6to4 インタフェースを持つホスト上で `ifconfig -a` コマンドを実行すると、次のような出力が返されます。

```
qfe1:3: flags=2180841<UP,RUNNING,MULTICAST,ADDRCONF,ROUTER,IPv6>
 mtu 1500 index 7
        inet6 2002:8192:56bb:9258:a00:20ff:fea9:4521/64
```

この出力では、`inet6` に続く文字列が 6to4 派生アドレスです。

この表では、6to4 派生アドレスの構成要素、各要素の長さ、および各要素が提供する情報について説明しています。

アドレスの構成要素	長さ	定義
prefix	16 ビット	`2002`。これは 6to4 接頭辞です。
IPv4-address	32 ビット	`8192:56bb`。これは、6to4 ルーターで構成されている 6to4 擬似インタフェースに対する 16 進表記による IPv4 アドレスです。
subnet-ID	16 ビット	`9258`。これは、このホストの所属先であるサブネットのアドレスです。
interface-ID	64 ビット	`a00:20ff:fea9:4521`。これは、6to4 に構成されているホストインタフェースのインタフェース ID です。

IPv6 マルチキャストアドレスの詳細

IPv6 マルチキャストアドレスは、同じ情報またはサービスを「マルチキャストグループ」という定義済みのインタフェースのグループに分配する方法を提供します。 通常、マルチキャストグループのインタフェースは異なるノード上にあります。1 つのインタフェースが所属できるマルチキャストグループは複数設定できます。マルチキャストアドレスに送信されたパケットは、このマルチキャストグループのすべてのメンバーに送信されます。マルチキャストアドレスの使用法の例としては、情報のブロードキャストがあります。これは、IPv4 ブロードキャストアドレスに似た機能です。

次の表に、マルチキャストアドレスの書式を示します。

表 11–1　IPv6 マルチキャストアドレス書式

8 ビット	4 ビット	4 ビット	8 ビット	8 ビット	64 ビット	32 ビット
11111111	*FLGS*	SCOP	*Reserved*	*Plen*	*Network prefix*	*Group ID*

次に、各フィールドの内容をサマリーします。

- 11111111 – アドレスをマルチキャストアドレスとして識別します。
- *FLGS* – 4 つのフラグ「0,0,P,T」のセットです。最初の 2 つのフラグはゼロである必要があります。P フィールドは、次の値のうちの 1 つとなります。
  - 0 = ネットワーク接頭辞に基づいて割り当てられていないマルチキャストアドレス
  - 1 = ネットワーク接頭辞に基づいて割り当てられているマルチキャストアドレス

  P を 1 に設定した場合、T も 1 に設定する必要があります。
- *Reserved* - ゼロの予約値。

- *Plen* - サブネットを識別するサイト接頭辞内のビット数。サイト接頭辞に基づいて割り当てられているマルチキャストアドレス用。
- *Group ID* - 恒久的または動的に割り当てられたマルチキャストグループの識別子。

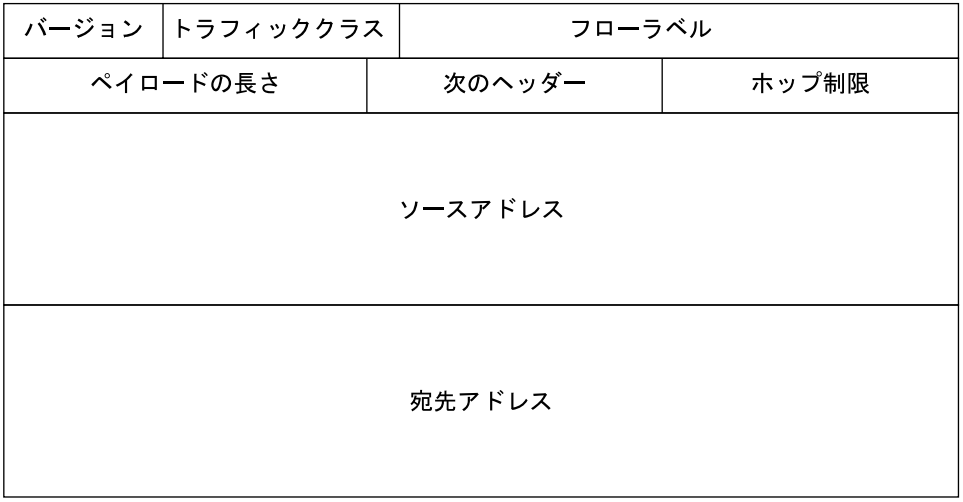
マルチキャストアドレスの書式の詳細については、RFC 3306, "Unicast-Prefix-based IPv6 Multicast Addresses (`ftp://ftp.rfc-editor.org/in-notes/rfc3306.txt`) を参照してください。

いくつかの IPv6 マルチキャストアドレスは、Internet Assigned Numbers Authority (IANA) によって恒久的に割り当てられます。これらの例としては、IPv6 ノードと IPv6 ルーターに必要な All Nodes Multicast Addresses と All Routers Multicast Addresses などがあります。IPv6 マルチキャストアドレスは動的に割り当てることもできます。マルチキャストアドレスとマルチキャストグループの適切な使用方法の詳細については、RFC 3307, "Allocation Guidelines for IPv6 Multicast Addresses" を参照してください。

IPv6 パケットヘッダーの書式

IPv6 プロトコルは、基本 IPv6 ヘッダー、IPv6 拡張ヘッダーを含むヘッダーセットを定義します。次の図は、IPv6 ヘッダーに使用されるフィールドとその順序を示します。

図 11–3 IPv6 基本ヘッダーの書式



次のリストは、各ヘッダーフィールドの機能について説明します。

- バージョン – 4 ビットインターネットプロトコルバージョン番号。IPv6 では 6 です。
- トラフィッククラス – 8 ビットトラフィッククラスフィールド。
- フローラベル – 20 ビットフィールド。
- ペイロードの長さ – オクテット単位で表す 16 ビット符号なし整数。IPv6 ヘッダーに続くパケットの残りです。
- 次のヘッダー – 8 ビットセレクタ。IPv6 ヘッダーのすぐ後ろに続くヘッダーのタイプを識別します。IPv4 プロトコルフィールドと同じ値を使用します。
- ホップ制限 – 8 ビット符号なし整数。パケットを送信するノードごとに値が 1 ずつ減ります。ホップ制限がゼロになるとパケットが廃棄されます。
- ソースアドレス – 128 ビット。パケットの初期送信側のアドレス。
- 宛先アドレス – 128 ビット。パケットの予定受信側のアドレス。オプションの経路制御ヘッダーがある場合、必ずしも受信側とは限りません。

IPv6 拡張ヘッダー

IPv6 オプションは、IPv6 ヘッダーとトランスポート層の間の独立した拡張ヘッダーにあります。パケットが最終的な宛先に到着するまで、その配送パスに存在するルーターは、ほとんどの場合 IPv6 拡張ヘッダーを確認または処理しません。そのため、オプションを含むパケットを処理するルーターのパフォーマンスが大幅に改善されました。IPv4 では、オプションがある場合、ルーターですべてのオプションを調べる必要がありました。

IPv4 オプションとは異なり、IPv6 拡張ヘッダーの長さは任意です。またパケットに組み込むことのできるオプションの合計数が 40 バイト以内に限定されない点があります。この機能とその処理方法によって、IPv4 では非現実的であった機能を IPv6 オプションが使用できるようになりました。

後続のオプションヘッダー (およびそのあとのトランスポートプロトコル) を処理する際のパフォーマンスを強化するため、IPv6 オプションは常に 8 オクテットの整数倍の長さです。この 8 オクテットの整数倍という長さにより、後続ヘッダーのバイト境界が維持されます。

次の IPv6 拡張ヘッダーが現在、定義されています。

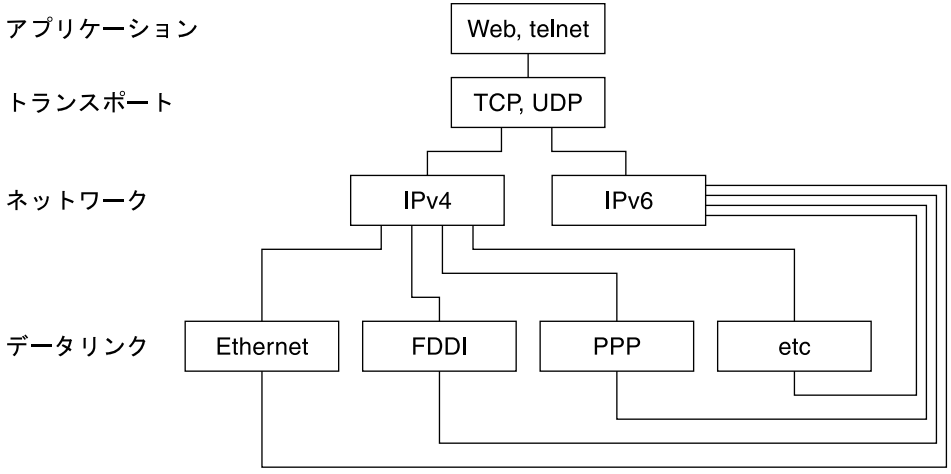
- 経路制御 – 拡張経路制御 (IPv4 ルーズソースルートにあたる)
- 断片化 – 断片化および再結合
- 認証 – 整合性および認証、セキュリティー
- セキュリティーペイロードのカプセル化 – 機密性
- ホップバイホップオプション – ホップごとの処理が必要な特別なオプション
- 宛先オプション – 宛先ノードが判断するオプション情報

デュアルスタックプロトコル

「デュアルスタック」とは、アプリケーションからネットワーク層に至るプロトコルスタックのすべてのレベルの完全な複製をいいます。完全な複製の例として、OSI プロトコルとTCP/IP プロトコルを両方とも実行するシステムがあります。

Oracle Solaris は「デュアルスタック」です。これは、Oracle Solaris で IPv4 プロトコルと IPv6 プロトコルの両方が実装されていることを意味します。オペレーティングシステムをインストールするとき、IPv6 プロトコルを IP 層で有効にするか、デフォルトの IPv4 プロトコルだけを使用するかを選択できます。TCP/IP スタックの残りは同じです。結果として、同じ転送プロトコル、TCP、UDP、および SCTP を IPv4 と IPv6 の両方で実行できます。また、同じアプリケーションを IPv4 と IPv6 の両方で実行できます。図 11–4 は、インターネットプロトコル群のさまざまな層において、IPv4 プロトコルと IPv6 プロトコルがデュアルスタックとしてどのように機能するかを示しています。

図 11–4　デュアルスタックプロトコルアーキテクチャー



デュアルスタックシナリオでは、ホストとルーターの両方のサブネットは、IPv4 に加えて、IPv6 をサポートするようにアップグレードされます。デュアルスタックアプローチによって、アップグレードしたノードは、IPv4 を使用して IPv4 専用ノードと常に相互運用できます。

Oracle Solaris の IPv6 の実装

このセクションでは、Oracle Solaris で IPv6 が有効なファイル、コマンド、およびデーモンについて説明します。

IPv6 構成ファイル

このセクションでは、IPv6 実装の一部である構成ファイルについて説明します。

- 272 ページの「ndpd.conf 構成ファイル」
- 275 ページの「IPv6 インタフェース構成ファイル」
- 276 ページの「/etc/inet/ipaddrsel.conf 構成ファイル」

ndpd.conf 構成ファイル

/etc/inet/ndpd.conf ファイルは、 近傍検索デーモン in.ndpd が使用するオプションを構成するために使用されます。ルーターの場合、ndpd.conf は、主にサイト接頭辞をリンクに通知されるように構成するときに使用します。ホストの場合、ndpd.conf は、アドレスの自動構成を無効にしたり、一時アドレスを構成したりするときに使用します。

次の表に、ndpd.conf ファイルで使用されるキーワードを示します。

表 11–2 /etc/inet/ndpd.conf キーワード

変数	説明
ifdefault	すべてのインタフェースのルーターの動作を指定します。次の構文を使用してルーターパラメータと対応する値を設定します。 ifdefault [*variable-value*]
prefixdefault	接頭辞通知のデフォルトの動作を指定します。次の構文を使用してルーターパラメータと対応する値を設定します。 prefixdefault [*variable-value*]
if	インタフェース別パラメータを設定します。構文は次のとおりです。 if *interface* [*variable-value*]
prefix	インタフェース別接頭辞情報を通知します。構文は次のとおりです。 prefix *prefix/length interface* [*variable-value*]

ndpd.conf ファイルでは、この表にあるキーワードといっしょに、いくつかのルーター設定変数を使用します。これらの変数の詳細については、RFC 2461, Neighbor Discovery for IP Version 6 (IPv6) (http://www.ietf.org/rfc/rfc2461.txt?number=2461) を参照してください。

次の表に、インタフェースを構成するための変数と、その簡単な説明を示します。

表 11–3 /etc/inet/ndpd.conf インタフェース構成変数

変数	デフォルト	定義
AdvRetransTimer	0	ルーターが送信する通知メッセージにおいて、Retrans Timer フィールドの値を指定します。
AdvCurHopLimit	インターネットの現在の直径	ルーターが送信する通知メッセージにおいて、現在のホップ制限に設定する値を指定します。
AdvDefaultLifetime	3 + MaxRtrAdvInterval	ルーター広告のデフォルトの寿命を指定します。
AdvLinkMTU	0	ルーターが送信する最大転送単位 (MTU) の値を指定します。ゼロは、ルーターが MTU オプションを指定しないことを意味します。
AdvManaged Flag	False	ルーター広告において、Manage Address Configuration フラグに構成する値を指定します。
AdvOtherConfigFlag	False	ルーター広告において、Other Stateful Configuration フラグに構成する値を指定します。
AdvReachableTime	0	ルーターが送信する通知メッセージにおいて、Reachable Time フィールドの値を指定します。
AdvSendAdvertisements	False	ノードが通知を送信し、ルーター要請に応答するかどうかを指定します。ルーター広告機能を有効にするには、ndpd.conf ファイルにおいて、この変数を明示的に「TRUE」に設定する必要があります。詳細については、184 ページの「IPv6 対応のルーターを構成する方法」を参照してください。
DupAddrDetect Transmits	1	近傍検索プロトコルがローカルノードのアドレスの重複アドレス検出中に送信する、連続近傍要請メッセージの数を定義します。
MaxRtrAdvInterval	600 秒	非要請マルチキャスト通知を送信する間隔の最大時間を指定します。
MinRtrAdvInterval	200 秒	非要請マルチキャスト通知を送信する間隔の最小時間を指定します。
StatelessAddrConf	True	ノードがその IPv6 アドレスを構成するときに、ステートレスアドレス自動構成を使用するかどうかを制御します。ndpd.conf で False が宣言されている場合、そのアドレスは手動で構成する必要があります。詳細については、192 ページの「ユーザー指定の IPv6 トークンを構成する方法」を参照してください。

表 11–3 /etc/inet/ndpd.conf インタフェース構成変数 (続き)

変数	デフォルト	定義
TmpAddrsEnabled	False	あるノードのすべてのインタフェースまたは特定のインタフェースに対して、一時アドレスを作成するかどうかを指定します。詳細については、189 ページの「一時アドレスを構成する方法」を参照してください。
TmpMaxDesyncFactor	600 秒	in.ndpd を起動するときに、優先寿命変数 TmpPreferredLifetime から引くランダム数を指定します。TmpMaxDesyncFactor 変数の目的は、ネットワーク上のすべてのシステムが同時に一時アドレスを再生成することを防ぐことです。TmpMaxDesyncFactor を使用すると、このランダム数の上限値を変更できます。
TmpPreferredLifetime	False	一時アドレスの優先寿命を設定します。詳細については、189 ページの「一時アドレスを構成する方法」を参照してください。
TmpRegenAdvance	False	一時アドレスのアドレス劣化までの先行時間を指定します。詳細については、189 ページの「一時アドレスを構成する方法」を参照してください。
TmpValidLifetime	False	一時アドレスの有効寿命を設定します。詳細については、189 ページの「一時アドレスを構成する方法」を参照してください。

次の表に、IPv6 接頭辞を構成するときに使用する変数を示します。

表 11–4 /etc/inet/ndpd.conf 接頭辞構成変数

変数	デフォルト	定義
AdvAutonomousFlag	True	Prefix Information オプションの Autonomous Flag フィールドに格納される値を指定します。
AdvOnLinkFlag	True	Prefix Information オプションのオンリンクフラグ (“L-bit”) に格納される値を指定します。
AdvPreferredExpiration	設定なし	接頭辞の優先満了日を指定します。
AdvPreferredLifetime	604800 秒	Prefix Information オプションの優先寿命に格納される値を指定します。
AdvValidExpiration	設定なし	接頭辞の有効満了日を指定します。
AdvValidLifetime	2592000 秒	構成している接頭辞の有効寿命を指定します。

例 11–1 /etc/inet/ndpd.conf ファイル

次に、ndpd.conf ファイルでキーワードや構成変数を使用する例を示します。変数を有効にするには、コメント (#) を削除します。

例 11–1 /etc/inet/ndpd.conf ファイル (続き)

```
# ifdefault       [variable-value ]*
# prefixdefault [variable-value ]*
# if ifname    [variable-value ]*
# prefix prefix/length ifname
#
#  Per interface configuration variables
#
#DupAddrDetectTransmits
#AdvSendAdvertisements
#MaxRtrAdvInterval
#MinRtrAdvInterval
#AdvManagedFlag
#AdvOtherConfigFlag
#AdvLinkMTU
#AdvReachableTime
#AdvRetransTimer
#AdvCurHopLimit
#AdvDefaultLifetime
#
# Per Prefix:  AdvPrefixList configuration variables
#
#
#AdvValidLifetime
#AdvOnLinkFlag
#AdvPreferredLifetime
#AdvAutonomousFlag
#AdvValidExpiration
#AdvPreferredExpiration

ifdefault AdvReachableTime 30000 AdvRetransTimer 2000
prefixdefault AdvValidLifetime 240m AdvPreferredLifetime 120m

if qe0 AdvSendAdvertisements 1
prefix 2:0:0:56::/64 qe0
prefix fec0:0:0:56::/64 qe0

if qe1 AdvSendAdvertisements 1
prefix 2:0:0:55::/64 qe1
prefix fec0:0:0:56::/64 qe1

if hme1 AdvSendAdvertisements 1
prefix  2002:8192:56bb:1::/64 qfe0

if hme1 AdvSendAdvertisements 1
prefix  2002:8192:56bb:2::/64 hme1
```

IPv6 インタフェース構成ファイル

起動時、IPv6 は /etc/hostname6.*interface* ファイルを使用して、IPv6 論理インタフェースを自動的に定義します。Oracle Solaris のインストール中に IPv6 Enabled オプションを選択した場合、インストールプログラムは、/etc/hostname.*interface* ファイルに加えて、プライマリネットワークインタフェース用の /etc/hostname6.*interface* ファイルを作成します。

インストール中に複数の物理インタフェースが検出された場合、このようなインタフェースを構成するかどうかをたずねられます。インストールプログラムは、指定された追加のインタフェースごとに、IPv4 物理インタフェース構成ファイルと IPv6 論理インタフェース構成ファイルを作成します。

IPv4 インタフェースと同様に、IPv6 インタフェースも Oracle Solaris インストール後に手動で構成できます。新しいインタフェースには `/etc/hostname6.`*interface* ファイルを作成します。インタフェースを手動で構成する方法については、第 6 章「ネットワークインタフェースの管理 (作業)」を参照してください。

ネットワークインタフェース構成ファイル名の構文は次のとおりです。

```
hostname.interface
hostname6.interface
```

interface 変数の構文は次のとおりです。

dev`[.`*module*`[.`*module* `...]]`*PPA*

dev ネットワークインタフェースデバイスを示します。デバイスは `eri` や `qfe` などの物理ネットワークインタフェースか、トンネルなどの論理インタフェースです。詳細については、275 ページの「IPv6 インタフェース構成ファイル」を参照してください。

Module plumb される際にデバイスにプッシュされる 1 つまたは複数の `STREAMS` モジュールのリスト。

PPA 物理的な接続ポイントを示します。

構文 [.[.]] も可能です。

例 11–2 IPv6 インタフェース構成ファイル

次に、有効な IPv6 構成ファイル名の例を示します。

```
hostname6.qfe0
hostname.ip.tun0
hostname.ip6.tun0
hostname6.ip6to4tun0
hostname6.ip.tun0
hostname6.ip6.tun0
```

`/etc/inet/ipaddrsel.conf` 構成ファイル

`/etc/inet/ipaddrsel.conf` ファイルには、IPv6 デフォルトアドレス選択ポリシーテーブルが含まれます。Oracle Solaris をインストールしたときに IPv6 を有効にした場合、このファイルには、表 11–5 に示す内容が含まれます。

/etc/inet/ipaddrsel.conf ファイルの内容は編集できます。しかし、このファイルを変更することは極力避けるべきです。どうしても変更が必要な場合、手順については、232 ページの「IPv6 アドレス選択ポリシーテーブルを管理する方法」を参照してください。ippaddrsel.conf の詳細については、278 ページの「IPv6 アドレス選択ポリシーテーブルを変更する理由」と ipaddrsel.conf(4) のマニュアルページを参照してください。

IPv6 関連のコマンド

このセクションでは、Oracle Solaris IPv6 実装で追加されたコマンドについて説明します。また、IPv6 をサポートするために行われた既存のコマンドへの変更についても説明します。

ipaddrsel コマンド

ipaddrsel コマンドを使用すると、IPv6 デフォルトアドレス選択ポリシーテーブルを変更できます。

Oracle Solaris カーネルは IPv6 デフォルトアドレス選択ポリシーテーブルを使用して、IPv6 パケットヘッダーに対して、宛先アドレス順序付けやソースアドレス選択を実行します。/etc/inet/ipaddrsel.conf ファイルには、このポリシーテーブルが含まれます。

次の表に、このポリシーテーブルのデフォルトアドレス書式とその優先度のリストを示します。IPv6 アドレス選択に関する技術的な詳細については、inet6(7P) のマニュアルページを参照してください。

表 11–5　IPv6 アドレス選択ポリシーテーブル

接頭辞	優先度	定義
::1/128	50	ループバック
::/0	40	デフォルト
2002::/16	30	6to4
::/96	20	IPv4 互換
::ffff:0:0/96	10	IPv4

この表では、IPv6 接頭辞 (::1/128 と ::/0) は 6to4 アドレス (2002::/16) と IPv4 アドレス (::/96 と ::ffff:0:0/96) よりも優先されます。したがって、カーネルは、別の IPv6 宛先に向かうパケットに対して、インタフェースのグローバル IPv6 アドレスをデフォルトで選択します。インタフェースの IPv4 アドレスの優先度は、特に IPv6 宛

先に向かうパケットに対しては低くなります。選択した IPv6 ソースアドレスを考えて、カーネルは宛先アドレスにも IPv6 書式を使用します。

IPv6 アドレス選択ポリシーテーブルを変更する理由

ほとんどの場合、IPv6 デフォルトアドレス選択ポリシーテーブルを変更する必要はありません。どうしてもポリシーテーブルを管理する必要がある場合は、`ipaddrsel` コマンドを使用します。

次のような場合、ポリシーテーブルの変更をお勧めします。

- システムが 6to4 トンネル用のインタフェースを持っている場合、6to4 アドレスにより高いアドレスに変更できます。
- 特定の宛先アドレスと通信するときだけ特定のソースアドレスを使用したい場合、これらのアドレスをポリシーテーブルに追加します。そのあと、`ifconfig` コマンドを使用して、これらのアドレスが優先されるようにフラグを立てます。
- IPv4 アドレスを IPv6 アドレスよりも優先させたい場合、`::ffff:0:0/96` の優先度をより大きな値に変更します。
- 旧式のアドレスにより高い優先度を割り当てる必要がある場合は、旧式のアドレスをポリシーテーブルに追加します。たとえば、IPv6 内でサイトのローカルアドレスが旧式であると仮定します。これらのアドレスには、`fec0::/10` という接頭辞があります。この場合、ポリシーテーブルを変更すると、サイトのローカルアドレスにより高いポリシーを与えることができます。

`ipaddrsel` コマンドの詳細については、`ipaddrsel(1M)` のマニュアルページを参照してください。

`6to4relay` コマンド

「*6to4* トンネリング」を使用すると、孤立した 6to4 サイト間で通信できます。しかし、6to4 以外のネイティブ IPv6 サイトにパケットを転送する場合は、6to4 ルーターは 6to4 リレールーターとのトンネルを確立する必要があります。このトンネルが確立されると、「*6to4* リレールーター」によって 6to4 パケットが IPv6 ネットワークに転送され、最終的にネイティブ IPv6 サイトに送信されます。6to4 有効化サイトがネイティブな IPv6 サイトとデータを交換する必要がある場合、`6to4relay` コマンドを使用して、適切なトンネルを有効にします。

リレールーターの使用は安全とは言えないため、Oracle Solaris のデフォルト設定ではリレールーターとの間のトンネリングは無効になっています。このシナリオを実践に移す場合は、6to4 リレールーターとの間のトンネル構築に伴って発生する問題点をあらかじめ慎重に検討してください。6to4 リレールーターの詳細については、301 ページの「6to4 リレールーターとの間のトンネルについての考慮事項」を参照してください。6to4 リレールーターのサポートを有効にする場合、その関連手順については、199 ページの「6to4 トンネルを構成する方法」を参照してください。

6to4relay の構文

6to4relay コマンドの構文は次のとおりです。

```
6to4relay -e [-a IPv4-address] -d -h
```

-e	6to4 ルーターとエニーキャスト 6to4 リレールーター間のトンネルサポートを有効にします。このオプションを指定すると、トンネルのエンドポイントアドレスが 192.88.99.1 (6to4 リレールーターのエニーキャストグループのデフォルトアドレス) に設定されます。
-a *IPv4-address*	6to4 ルーターと指定された *IPv4-address* の 6to4 リレールーター間にトンネルのサポートを有効にします。
-d	6to4 リレールーターとの間のトンネリングのサポートを無効にします。これは、Oracle Solaris のデフォルトの設定です。
-h	6to4relay のヘルプを表示します。

詳細は、6to4relay(1M) のマニュアルページを参照してください。

例 11–3 6to4 リレールーターサポートのデフォルトのステータスの表示

引数を指定せずに 6to4relay コマンドを実行すると、6to4 リレールーターサポートの現在のステータスが表示されます。次の例に、Oracle Solaris における IPv6 実装のデフォルトを示します。

```
# /usr/sbin/6to4relay
6to4relay:6to4 Relay Router communication support is disabled
```

例 11–4 6to4 リレールーターサポートを有効にしたステータスの表示

リレールーターサポートが有効に設定されている場合には、6to4relay を実行すると次のように表示されます。

```
# /usr/sbin/6to4relay
6to4relay:6to4 Relay Router communication support is enabled
IPv4 destination address of Relay Router=192.88.99.1
```

例 11–5 6to4 リレールーターを指定したステータスの表示

6to4relay コマンドに -a オプションと IPv4 アドレスを指定した場合、192.88.99.1 ではなく、- a オプションに指定した IPv4 アドレスが表示されます。

6to4relay は、 -d、 -e、および -a *IPv4 address* オプションが成功したかどうかを報告しません。しかし、これらのオプションの実行時に発生した可能性のあるエラーは表示します。

IPv6 をサポートするための ifconfig コマンドの拡張

ifconfig コマンドにより、IPv6 インタフェースとトンネリングモジュールを plumb できるようになりました。ifconfig は、ioctl の拡張セットを使用して、IPv4 と IPv6 の両方のネットワークインタフェースを構成します。次に、IPv6 操作をサポートする ifconfig オプションについて説明します。ifconfig に関連する IPv4 と IPv6 の両方のタスクについては、211 ページの「ifconfig コマンドによるインタフェース構成の監視」を参照してください。

index	インタフェースインデックスを設定します。
tsrc/tdst	トンネルソース / 宛先を設定します。
addif	論理インタフェースの次の候補を作成します。
removeif	指定された IP アドレスの論理インタフェースを削除します。
destination	インタフェースにポイントツーポイント宛先アドレスを設定します。
set	インタフェースにアドレスとネットマスクのどちらか、または両方を設定します。
subnet	インタフェースのサブネットアドレスを設定します。
xmit/-xmit	インタフェースにおけるパケット伝送を使用可能または使用不能にします。

第 7 章「IPv6 ネットワークの構成 (手順)」では、IPv6 の構成手順について説明します。

例 11–6 ifconfig コマンドの -addif オプションによる IPv6 論理インタフェースの追加

次の形式の ifconfig コマンドは、hme0:3 論理インタフェースを作成します。

```
# ifconfig hme0 inet6 addif up
Created new logical interface hme0:3
```

次の形式の ifconfig は、新しいインタフェースの作成を確認します。

```
# ifconfig hme0:3 inet6
hme0:3: flags=2000841<UP,RUNNING,MULTICAST,IPv6> mtu 1500 index 2
        inet6  inet6 fe80::203:baff:fe11:b321/10
```

例 11–7 ifconfig コマンドの -removeif オプションによる IPv6 論理インタフェースの削除

次の形式の ifconfig コマンドは、hme0:3 論理インタフェースを削除します。

```
# ifconfig hme0:3 inet6 down

# ifconfig hme0 inet6 removeif 1234::5678
```

例 11–8 ifconfig コマンドによる IPv6 トンネルソースの構成

```
# ifconfig ip.tun0 inet6 plumb index 13
```

物理インタフェース名に関連するトンネルを開きます。

```
# ifconfig ip.tun0 inet6
ip.tun0: flags=2200850<POINTOPOINT,RUNNING,MULTICAST,NONUD,
#IPv6> mtu 1480 index 13
        inet tunnel src 0.0.0.0
        inet6 fe80::/10 --> ::
```

トンネルデバイスを使用して、そのデバイスのステータスを報告するように、TCP/IP に必要なストリームを構成します。

```
# ifconfig ip.tun0 inet6 tsrc 120.46.86.158 tdst 120.46.86.122
```

トンネルのソースアドレスと宛先アドレスを構成します。

```
# ifconfig ip.tun0 inet6
ip.tun0: flags=2200850<POINTOPOINT,RUNNING,MULTICAST,NONUD,
IPv6> mtu 1480 index 13
        inet tunnel src 120.46.86.158  tunnel dst 120.46.86.122
        inet6 fe80::8192:569e/10 --> fe80::8192:567a
```

構成後のデバイスの新しいステータスを報告します。

例 11–9 ifconfig による 6to4 トンネルの構成 (長形式)

この 6to4 擬似インタフェース構成例は、サブネット ID として 1 を使用し、ホスト ID を 16 進形式で指定しています。

```
# ifconfig ip.6to4tun0 inet6 plumb
# ifconfig ip.6to4tun0 inet tsrc 129.146.86.187 \
2002:8192:56bb:1::8192:56bb/64 up

# ifconfig ip.6to4tun0 inet6
ip.6to4tun0: flags=2200041<UP,RUNNING,NONUD,IPv6>mtu 1480 index 11
        inet tunnel src 129.146.86.187
        tunnel hop limit 60
        inet6 2002:8192:56bb:1::8192:56bb/64
```

例 11–10 ifconfig による 6to4 トンネルの構成 (短形式)

この例では、6to4 トンネルを構成するための短い形式を示します。

```
# ifconfig ip.6to4tun0 inet6 plumb
# ifconfig ip.6to4tun0 inet tsrc 129.146.86.187 up

# ifconfig ip.6to4tun0 inet6
ip.6to4tun0: flags=2200041<UP,RUNNING,NONUD,IPv6>mtu 1480 index 11
        inet tunnel src 129.146.86.187
        tunnel hop limit 60
```

例 11–10 ifconfig による 6to4 トンネルの構成 (短形式) (続き)

```
        inet6 2002:8192:56bb::1/64
```

IPv6 をサポートするための netstat コマンドの変更

netstat コマンドは、IPv4 ネットワークと IPv6 ネットワークの両方のステータスを表示します。 表示するプロトコル情報を選択するには、/etc/default/inet_type ファイルに DEFAULT_IP 値を設定するか、-f コマンド行オプションを使用します。DEFAULT_IP のパラメータ設定では、netstat に IPv4 情報だけが表示されていることを確認できます。この設定は、-f オプションでオーバーライドできます。inet_type ファイルの詳細については、inet_type(4) のマニュアルページを参照してください。

netstat コマンドの -p オプションは、net-to-media テーブルを表示します。これは、IPv4 の場合は ARP テーブルであり、IPv6 の場合は近傍キャッシュです。詳細は、netstat(1M) のマニュアルページを参照してください。このコマンドを使用する手順については、219 ページの「ソケットのステータスを表示する方法」を参照してください。

IPv6 をサポートするための snoop コマンドの変更

snoop コマンドは、IPv4 パケットと IPv6 パケットの両方を取り込むことができます。IPv6 ヘッダー、IPv6 拡張ヘッダー、ICMPv6 ヘッダー、近傍検索プロトコルデータを表示できます。デフォルトで、snoop コマンドは、IPv4 パケットと IPv6 パケットの両方を表示します。ip または ip6 のプロトコルキーワードを指定した場合、snoop コマンドは IPv4 パケットまたは IPv6 パケットだけを表示します。IPv6 フィルタオプションでは、すべてのパケットをフィルタの対象にでき (IPv4 と IPv6 の両方)、IPv6 パケットだけが表示されます。詳細は、snoop(1M) のマニュアルページを参照してください。snoop コマンドを使用する手順については、231 ページの「IPv6 ネットワークトラフィックを監視する方法」を参照してください。

IPv6 をサポートするための route コマンドの変更

route コマンドは IPv4 ルートと IPv6 ルートの両方で動作します。デフォルトでは、IPv4 ルートで動作します。route コマンドのすぐあとに -inet6 コマンド行オプションを指定した場合、route コマンドは IPv6 ルート上で動作します。詳細は、route(1M) のマニュアルページを参照してください。

IPv6 をサポートするための ping コマンドの変更

ping コマンドは、ターゲットホストを検証するのに、IPv4 プロトコルと IPv6 プロトコルの両方で使用できます。プロトコル選択は、指定のターゲットホストのネームサーバーが戻すアドレスに依存します。デフォルトでネームサーバーによって

ターゲットホストの IPv6 アドレスが返されると、ping コマンドは IPv6 プロトコルを使用します。サーバーが IPv4 アドレスだけを戻すと、ping コマンドは IPv4 プロトコルを使用します。-A コマンド行オプションで使用するプロトコルを指定すれば、この動作をオーバーライドできます。

詳細については、ping(1M) のマニュアルページを参照してください。ping を使用する手順については、222 ページの「ping コマンドによるリモートホストの検証」を参照してください。

IPv6 をサポートするための traceroute コマンドの変更

traceroute コマンドは、指定したホストへの IPv4 ルートと IPv6 ルートの両方で使用できます。使用するプロトコルの選択について、traceroute では、ping と同じアルゴリズムを使用します。選択をオーバーライドするには、-A コマンド行オプションを使用します。マルチホームホストのすべてのアドレスまでの各ルートは -a コマンド行オプションでトレースできます。

詳細については、traceroute(1M) のマニュアルページを参照してください。traceroute を使用する手順については、227 ページの「traceroute コマンドによる経路制御情報の表示」を参照してください。

IPv6 関連のデーモン

このセクションでは、IPv6 関連のデーモンについて説明します。

in.ndpd デーモン、近傍検索用

in.ndpd デーモンは、IPv6 近傍検索プロトコルとルーター発見を実装します。このデーモンは、IPv6 のアドレス自動構成も実装します。次に、in.ndpd でサポートされるオプションを示します。

-d　デバッグを有効にします。

-D　特定のイベントのデバッグを有効にします。

-f　デフォルトの /etc/inet/ndpd.conf ファイル以外で、構成データを読み取るファイルを指定します。

-I　インタフェースごとに関連情報を印刷します。

-n　ルーター広告をループバックしません。

-r　受信パケットを無視します。

-v　冗長モードを指定します (さまざまな種類の診断メッセージを報告する)。

-t　パケット追跡をオンに設定します。

in.ndpd デーモンは、/etc/inet/ndpd.conf 設定ファイルに設定されたパラメータと、/var/inet/ndpd_state.*interface* 起動ファイルの任意の適用可能なパラメータによって制御されます。

/etc/inet/ndpd.conf が存在すると構文解析され、ノードをルーターとして使用するための構成が行われます。表 11–2 に、このファイルに現れる可能性がある有効なキーワードのリストを示します。ホストをブートしても、ルーターがすぐには使用できない場合があります。ルーターによって通知されたパケットがドロップしたり、また、通知されたパケットがホストに届かない場合もあります。

/var/inet/ndpd_state.*interface* ファイルは状態ファイルです。このファイルはノードごとに定期的に更新されます。ノードに障害が発生し再起動した場合、ルーターがなくてもノードはインタフェースを構成できます。このファイルにはインタフェースアドレス、最終更新時間、有効期間などの情報が含まれています。また、先のルーター広告で得られた情報も含まれています。

注 – 状態ファイルの内容を変更する必要はありません。このファイルは、in.ndpd デーモンが自動的に管理します。

構成変数とそれに指定できる値のリストについては、in.ndpd(1M) のマニュアルページと ndpd.conf(4) のマニュアルページを参照してください。

in.ripngd デーモン、IPv6 ルーティング用

in.ripngd デーモンは、RIPng (Routing Information Protocol next-generation for IPv6 routers) を実装します。RIPng は IPv6 における RIP 相当機能を定義します。routeadm コマンドで IPv6 ルーターを構成し、IPv6 経路制御を有効にした場合、in.ripngd デーモンはそのルーターに RIPng を実装します。

次に、RIPng のサポートされるオプションを示します。

-p *n* — *n* は RIPng パケットの送受信に使用する代替ポート番号を指定します。

-q — ルーティング情報を打ち切ります。

-s — デーモンがルーターとして動作しているかどうかのルーティング情報の提供を強制します。

-P — ポイズンリバースを打ち切ります。

-S — in.ripngd がルーターとして機能しない場合、各ルーターにはデフォルトのルートだけが指定されます。

inetd デーモンと IPv6 サービス

IPv6 が有効なサーバーアプリケーションは、IPv4 要求と IPv6 要求の両方、あるいは、IPv6 要求だけを処理できます。IPv6 が有効なサーバーは常に、IPv6 ソケット経由の要求を処理します。さらに、IPv6 が有効なサーバーは、対応するクライアントで使用しているプロトコルと同じプロトコルを使用します。

IPv6 用にサービスを追加または変更するには、Service Management Facility (SMF) から入手できるコマンドを使用します。

- SMF コマンドについては、『Oracle Solaris の管理: 基本管理』の「SMF コマンド行管理ユーティリティー」を参照してください。
- SMF を使用して、SCTP 経由で動作する IPv4 サービスマニフェストを構成するタスクの例については、142 ページの「SCTP プロトコルを使用するサービスを追加する方法」を参照してください。

IPv6 サービスを構成するには、そのサービスの `inetadm` プロファイルにある `proto` フィールド値に、適切な値のリストが含まれていることを確認する必要があります。

- IPv4 要求と IPv6 要求の両方を処理するサービスの場合、proto 値として、`tcp6`、`udp6`、または `sctp` を選択します。`proto` 値として、`tcp6`、`udp6`、または `sctp6` のいずれかを選択した場合、`inetd` は IPv6 が有効なサーバーに IPv6 ソケットを渡します。IPv4 クライアントが要求を持っている場合に備えて、IPv6 が有効なサーバーは IPv4 マップ済みアドレスを含んでいます。
- IPv6 要求だけを処理するサービスの場合、proto 値として、`tcp6only` または `tcp6only` を選択します。これらの値を `proto` に選択した場合、 `inetd` は IPv6 が有効なサーバーに IPv6 ソケットを渡します。

Oracle Solaris コマンドを別の実装で置き換えた場合、そのサービスの実装が IPv6 をサポートすることを確認する必要があります。その実装が IPv6 をサポートしない場合、`proto` 値として、`tcp`、`udp`、または `sctp` のいずれかを指定する必要があります。

次に、IPv4 とIPv6 の両方をサポートし、SCTP で動作する `echo` サービスマニフェストに `inetadm` を実行した結果のプロファイルを示します。

```
# inetadm -l svc:/network/echo:sctp_stream
    SCOPE    NAME=VALUE      name="echo"
             endpoint_type="stream"
             proto="sctp6"
             isrpc=FALSE
             wait=FALSE
             exec="/usr/lib/inet/in.echod -s"
             user="root"
    default  bind_addr=""
    default  bind_fail_max=-1
    default  bind_fail_interval=-1
```

```
default  max_con_rate=-1
default  max_copies=-1
default  con_rate_offline=-1
default  failrate_cnt=40
default  failrate_interval=60
default  inherit_env=TRUE
default  tcp_trace=FALSE
default  tcp_wrappers=FALSE
```

proto フィールドの値を変更するには、次の構文を使用します。

```
# inetadm -m FMRI proto="transport-protocols"
```

Oracle Solaris ソフトウェアが提供されるサーバーはすべて、proto 値として、tcp6、udp6、または sctp6 のいずれかを指定するプロファイルエントリを 1 つだけ必要とします。しかし、リモートシェルサーバー (shell) とリモート実行サーバー (exec) は、現在、単一のサービスインスタンスで設定されており、proto 値として、tcp と tcp6only の両方を含める必要があります。たとえば、shell の proto 値を設定するには、次のコマンドを発行します。

```
# inetadm -m network/shell:default proto="tcp,tcp6only"
```

ソケットを使用する IPv6 対応サーバーの作成方法の詳細については、『プログラミングインタフェースガイド』のソケット API の IPv6 拡張機能を参照してください。

サービスを IPv6 用に構成するときの注意事項

サービスを IPv6 用に追加または変更するときには、次のことに注意しておく必要があります。

- IPv4 接続と IPv6 接続の両方を有効にするには、proto 値として、tcp6、sctp6、または udp6のいずれかを指定する必要があります。proto 値として、 tcp、 sctp、または udp を指定した場合、そのサービスは IPv4 だけを使用します。
- inetd に対して、一対多スタイルの SCTP ソケットを使用するサービスインスタンスも追加できますが、推奨しません。inetd は、一対多スタイルの SCTP ソケットでは機能しません。
- wait-status プロパティーまたは exec プロパティーが異なるため、サービスが 2 つのエントリを必要とする場合、オリジナルのサービスから 2 つのインスタンスまたはサービスを作成する必要があります。

IPv6 近傍検索プロトコル

IPv6 は近傍検索プロトコルを導入します (RFC 2461, Neighbor Discovery for IP Version 6 (IPv6) (http://www.ietf.org/rfc/rfc2461.txt?number=2461) を参照)。近傍検索の主な機能の概要については、83 ページの「IPv6 近傍検索プロトコルの概要」を参照してください。

このセクションでは、近傍検索プロトコルの次の機能について説明します。

近傍検索からの ICMP メッセージ

近傍検索では、次の 5 種類の新しい ICMP (インターネット制御メッセージプロトコル) メッセージを定義します。これらのメッセージの目的は、次のとおりです。

- ルーター要請 – インタフェースが有効になると、ホストはルーター要請メッセージを送信できます。この要請は、次に予定されている時間ではなく、ただちにルーター広告メッセージを送信するようにルーターに要求します。
- ルーター広告 – ルーターは自分の存在、さまざまなリンクパラメータ、およびさまざまなインターネットパラメータを通知します。ルーターは定期的に、あるいはルーター要請メッセージに応じて通知します。ルーター広告には、オンリンク判別またはアドレス構成、あるいはホップ限界数の選択肢などに使用する接頭辞が含まれます。
- 近傍要請 – ノードは近傍要請メッセージを送信して、近傍のリンク層アドレスを判別します。近傍要請メッセージはまた、キャッシュされたリンク層アドレスによって近傍が到達可能であるかを確認するために送信されます。近傍要請は重複アドレス検出にも使用します。
- 近傍通知 – ノードは、近傍要請メッセージへの応答として、近傍通知メッセージを送信します。ノードはまた、非要請近傍通知を送信して、リンク層アドレスの変更を通知できます。
- リダイレクト – ルーターはリダイレクトメッセージを使用して、宛先までのより高速なホップをホストに通知したり、宛先が同じリンク上にあることを通知します。

自動構成プロセス

このセクションでは、自動構成中にインタフェースが実行する一般的な手順の概要について説明します。自動構成が行われるのはマルチキャスト対応リンクだけです。

1. たとえば、ノードの起動中、マルチキャスト対応インタフェースが有効になります。
2. このノードは、そのインタフェースのリンクローカルアドレスを生成することによって、自動構成プロセスを開始します。

   リンクローカルアドレスは、インタフェースの MAC (Media Access Control) アドレスから形成されます。
3. このノードは、仮リンクローカルアドレスをターゲットとする近傍要請メッセージを送信します。

   このメッセージの目的は、仮リンクローカルアドレスが、すでにそのリンク上の別のノードによって使用されているかどうかを確認することです。この確認が終わったら、リンクローカルアドレスをインタフェースに割り当てることができます。

   a. 別のノードがすでにそのアドレスを使用していた場合、その別のノードは近傍通知メッセージを戻して、そのアドレスが使用中であることを伝えます。

   b. 別のノードがそのアドレスを使用しようと試みている場合、そのノードもその宛先に近傍要請を送信します。

      近傍要請送信や再送の数と、連続した要請間の遅延 はリンクによって異なります。これらのパラメータは、必要であれば設定できます。
4. 仮リンクローカルアドレスが一意でないとノードが判断した場合、自動構成は停止します。その時点で、インタフェースのリンクローカルアドレスは手動で構成する必要があります。

   復旧を簡素化するために、デフォルトの識別子をオーバーライドした代替のインタフェース ID を指定できます。これにより、一意であると考えられる新しいインタフェース ID を使用して、自動構成メカニズムを再開できます。
5. この仮リンクローカルアドレスが一意であると判断されると、ノードはインタフェースにそのアドレスを割り当てます。

   このとき、ノードは近傍ノードと IP レベルで接続されます。自動構成手順の残りは、ホストだけで実行されます。

ルーター広告の受信

自動構成の次の段階は、ルーター広告を受信するか、ルーターが存在しないことを判断することです。ルーターがあれば、ホストが実行すべき自動構成の種類を指定したルーター広告が送信されます。

ルーターはルーター広告を定期的に送信します。ただし、連続した送信と送信の間の遅延は、自動構成を実行するホスト側の待機時間より通常は長くなります。通知を迅速に受信するため、すべてのルーターマルチキャストグループに 1 つまたは複数のルーター要請を送信します。

接頭辞構成変数

ルーター広告には、ステートレスアドレス自動構成が接頭辞を生成するときに使用する接頭辞変数とその情報が含まれます。ルーター広告の Stateless Address Autoconfiguration フィールドは個別に処理されます。接頭辞情報オプションフィールドの1 つである Address Autoconfiguration フラグは、オプションがステートレス自動構成にも適用されるかどうかを表します。適用される場合、補助オプションフィールドにサブネット接頭辞と寿命値が含まれます。これらの値は、接頭辞から作成されたアドレスがどれだけの時間優先権を持ち有効であるかを表します。

ルーターは定期的にルーター広告を生成するため、ホストは新しい通知を受信し続けます。IPv6 が有効なホストは、各通知に含まれる情報を処理します。情報を追加します。また、ホストは前の通知で受け取った情報をリフレッシュします。

アドレスの一意性

セキュリティーのため、すべてのアドレスは、インタフェースに割り当てられる前に、その一意性をテストする必要があります。ただし、ステートレス自動構成で作成したアドレスの場合は状況が異なります。アドレスの一意性は、インタフェース ID から生成されるアドレスの一部で主に決まります。したがって、ノードにおいてリンクローカルアドレスの一意性が確認されると、ほかのアドレスの個別の確認は不要になります。これらのアドレスが、同じインタフェース ID から生成されているためです。ただし、手動で得られるアドレスはすべて、個別に一意であることを確認する必要があります。一部のサイトのシステム管理者は、重複アドレス検出を実行するためのオーバーヘッドが大きく、それを実行することで得られる利益が帳消しになると信じています。そのようなサイトでは、インタフェース別設定フラグの設定で重複アドレス検出の使用を無効にできます。

自動構成処理を短時間で終了するために、ルーター広告の待機、リンクローカルアドレスの生成、およびその一意性の確認を、ホストで並列して実行できます。ルーターでは、ルーター要請に対する応答が数秒遅れる可能性があります。そのため、上記 2 つの手順を 1 つずつ実行すると、自動構成を完了するために必要な合計時間が大幅に長くなる可能性があります。

近傍要請と不到達

近傍検索は、「近傍要請」メッセージを使用して、複数のノードに同じユニキャストアドレスが割り当てられているかどうかを判断します。「近傍不到達検出」で

は、近傍エラーや近傍への送信パスのエラーを検出します。近傍不到達検出では、近傍に送信されるパケットがその近傍に実際にアクセスして、パケットがノードの IP 層によって適切に処理されているかどうかを判断します。

近傍不到達検出では、2 つのソースの確認を使用します。 つまり、上位層プロトコルと近傍要請メッセージです。可能な場合、上位層のプロトコルでは、接続が送信を処理中であるという肯定確認を戻します。たとえば、新しい TCP 確認を受信した場合、以前送信されたデータが正しく送信されたことが確認されます。

あるノードが上位層プロトコルから肯定的な確認を受信しない場合、このノードはユニキャスト近傍要請メッセージを送信します。このメッセージは、次のホップからの到達可能確認として近傍通知を要請します。不要なネットワークトラフィックを避けるため、ノードからアクティブにパケットが送信されている近傍にだけ探査メッセージが送信されます。

重複アドレス検出アルゴリズム

すべての構成されたアドレスが特定のリンク上で一意であるかどうかを確認するために、ノードは「重複アドレス検出」アルゴリズムをアドレスに対して実行します。この実行は、インタフェースにアドレスを割り当てる前に行われる必要があります。重複アドレス検出アルゴリズムは、すべてのアドレスを対象として実行されます。

このセクションで指定する自動構成プロセスは、ホストにだけ適用し、ルーターには適用しません。ホストの自動構成では、ルーターが通知した情報を使用するため、ルーターは別の手段で構成する必要があります。ただし、この章で説明したメカニズムを使用して、ルーターによってリンクローカルアドレスが生成される場合があります。また、インタフェースに割り当てられる前に、すべてのアドレスにおいてルーターによる重複アドレス検出アルゴリズムが正常終了していることが望まれます。

プロキシ通知

ターゲットアドレスの代わりにパケットを受信するルーターは、オーバーライドできない近傍通知を発行できます。ルーターは、近傍要請に応答できない宛先アドレスのかわりにパケットを受信します。現在はプロキシの使用方法は指定されていません。ただし、オフリンクになった移動ノードをプロキシ通知で処理できる可能性があります。プロキシは、このプロトコルを実装していないノードを処理する一般的なメカニズムとして使用されることはありません。

インバウンド負荷分散

インタフェースを複製したノードでは、同じリンク上の複数のネットワークインタフェース間の入力パケットの受信の負荷分散が必要になる可能性があります。このようなノードには、同じインタフェースに複数のリンクローカルアドレスが割り当てられます。たとえば、1 つのネットワークドライバで、複数のネットワークインタフェースカードを、複数のリンクローカルアドレスを持つ 1 つの論理インタフェースとして表現できます。

負荷分散は、ルーターがソースリンクローカルアドレスをルーター広告パケットから省略することを可能にすることで処理します。結果として、ルーターのリンクローカルアドレスを確認するために、近傍は近傍要請メッセージを使用する必要があります。返される近傍通知メッセージには、要請元によって異なるリンクローカルアドレスが含まれます。

リンクローカルアドレスの変更

リンクローカルアドレスの変更を認識したノードは、非要請近傍通知パケットをマルチキャストできます。ノードは、すべてのノードにパケットをマルチキャストして、無効になったキャッシュに入っているリンクローカルアドレスを更新できます。非要請通知の送信は、パフォーマンス強化が目的です。近傍不到達検出アルゴリズムにより、すべてのノードが確実に新しいアドレスを探索できますが、遅延が多少伸びる可能性があります。

近傍検索と ARP および関連する IPv4 プロトコルとの比較

IPv6 近傍検索プロトコルの機能は、次のような IPv4 プロトコルの組み合わせのようなものです。 つまり、アドレス解決プロトコル (ARP)、Internet Control Message Protocol (ICMP)、ルーター発見、および ICMP リダイレクトです。IPv4 には近傍不到達検出に全般的に対応できるプロトコルやメカニズムはありませんでした。ただし、ホスト条件ではデッドゲートウェイ検出に対応できるアルゴリズムがいくつか指定されています。デッドゲートウェイ検出は、近傍不到達検出の一部です。

次のリストは、近傍検索プロトコルと関連する IPv4 プロトコルセットを比較します。

- ルーター発見は IPv6 ベースプロトコルセットの一部です。IPv6 ホストは、ルーターを検索するために、ルーティングプロトコルを `snoop` する必要はありません。IPv4 は、ルーターを検索するために、ARP、ICMP ルーター発見、および ICMP リダイレクトを使用します。

- IPv6 ルーター広告はリンクローカルアドレスを伝達します。ルーターのリンクローカルアドレスを解決するために、これ以外のパケットを交換する必要はありません。
- ルーター広告はリンクのサイト接頭辞を伝達します。IPv4 の場合と同様に、ネットマスクを構成するのに別のメカニズムは必要ありません。
- ルーター広告では、アドレス自動構成が使用可能になります。自動構成は IPv4 には実装されません。
- 近傍検索により、IPv6 ルーターはホストの MTU を通知して、リンクで使用できるようにします。したがって、MTU が定義されていないすべてのノードはリンク上の同じ MTU 値を使用します。IPv4 の場合、同じネットワーク上のホストが異なる MTU を持つ場合もあります。
- IPv4 ブロードキャストアドレスとは異なり、IPv6 アドレス解決マルチキャストは 40 億個を超える (2^32) マルチキャストアドレスを持つため、ターゲット以外のノードに対するアドレス解決関係の割り込みを大幅に減らしました。さらに、IPv6 以外のマシンの割り込みをなくしました。
- IPv6 リダイレクトには、新しい最初のホップのリンクローカルアドレスが含まれます。独立したアドレス解決がなくてもリダイレクトを受信できます。
- 同じ IPv6 ネットワークに複数のサイト接頭辞を関連付けられます。デフォルトでは、ホストはローカルサイトのすべての接頭辞をルーター広告を通じて知ります。ただし、ルーター広告にある接頭辞をすべて、あるいは一部省略するようにルーターを構成できます。その場合、ホストは宛先がリモートネットワーク上にあるとみなします。その結果、ホストはルーターにトラフィックを送信します。ルーターは適宜リダイレクトを発行します。
- IPv4 とは異なり、IPv6 リダイレクトの受信者は新しい次のホップがローカルネットワーク上にあるとみなします。IPv4 では、リダイレクトメッセージに指定されている次のホップが (ネットワークマスクによると) ローカルネットワーク上にない場合、ホストはそのリダイレクトメッセージを無視します。IPv6 リダイレクトメカニズムは、IPv4 の XRedirect 機能に似ています。このリダイレクトメカニズムは、非ブロードキャストおよび共有メディアリンク上で便利です。このようなネットワークでは、ノードはローカルリンク宛先のすべての接頭辞を確認できません。
- IPv6 近傍不到達検出は、障害ルーターが存在する場合のパケット伝送能力を改善します。この機能は、部分的に障害があるリンクやパーティション化されたリンクを経由するパケット伝送を改善します。この機能はまた、自分のリンクローカルアドレスを変更するノードを経由するパケット伝送も改善します。たとえば、頻繁に更新される ARP キャッシュのおかげで、移動ノードはローカルネットワークから離れても切断されません。IPv4 には、近傍不到達検出に相当する機能がありません。

- ARP とは異なり、近傍検索では、近傍不到達検出により、ハーフリンクエラーを検出します。近傍検索は、双方向接続がない近傍にトラフィックが送信されるのを防ぎます。
- リンクローカルアドレスでルーターを一意に識別しておけば、ホストでルーター関連付けを維持できます。ルーターを識別する機能は、ルーター広告とリダイレクトメッセージで必要とされます。サイトが新しいグローバル接頭辞を使用しても、ホストはルーター関連付けを維持する必要があります。IPv4 には、ルーター識別に相当する機能がありません。
- 近傍検索メッセージのホップ制限は受信時に 255 なので、プロトコルがオフリンクノードによるスプーフエラーの被害を受けることがありません。逆に、IPv4 オフリンクノードは ICMP リダイレクトメッセージを送信できます。IPv4 オフリンクノードはルーター通知メッセージを送ることもできます。
- ICMP 層にアドレス解決を配置すると、近傍検索が ARP よりもメディアに依存しなくなります。その結果、標準 IP 認証とセキュリティーメカニズムが使用できるようになります。

IPv6 のルーティング

IPv6 におけるルーティングは、Classless Inter-Domain Routing (CIDR) 下における IPv4 のルーティングとほとんど同じです。唯一の違いは、IPv4 では 32 ビットアドレスを使用しますが、IPv6 では 128 ビットアドレスを使用することです。非常に簡単な拡張で、IPv4 のルーティングアルゴリズム (OSPF、RIP、IDRP、IS-IS など) をすべて IPv6 のルーティングに使用できます。

IPv6 には、新たに強力なルーティング機能をサポートした簡単なルーティング拡張機能も組み込まれました。次のリストに、新しいルーティング機能を示します。

- プロバイダ選択 (ポリシー、パフォーマンス、コストなどを基準に)
- ホストの移動性 (現在の場所までのルート)
- アドレスの自動的な再指定 (新しいアドレスへのルート)

新しいルーティング機能を利用するには、IPv6 ルーティングオプションを使用する IPv6 アドレスのシーケンスを作成します。IPv6 の送信元は、ルーティングオプションでを使用して、パケットが宛先に至るまでに経由する複数の中間ノード (またはトポロジカルグループ) をリストします。この中間ノードは、パケットの宛先の途中に通過します。この機能は、IPv4 での緩やかな経路制御と記録オプションによく似ています。

アドレスシーケンスを一般的に使用する場合、通常は、ホストが受信したパケットのルートを逆戻りする必要があります。このパケットは、IPv6 認証ヘッダーを使用して正常に認証される必要があります。パケットを発信者に戻すには、アドレスシーケンスがパケット内に含まれている必要があります。IPv6 ホストの実装で

は、この方式により始点ルートの処理と逆引きをサポートしています。始点ルートの処理と逆引きは、IPv6 の新機能 (プロバイダの選択や拡張アドレスなど) を実装するホストをプロバイダが使用するための鍵です。

ルーター広告

マルチキャスト対応リンクとポイントツーポイントリンクでは、各ルーターは定期的にルーター広告パケットをマルチキャストグループに送信して、ルーターが利用できることを知らせます。ホストはすべてのルーターからルーター広告を受け取り、デフォルトルーターのリストを作成します。ルーターは頻繁にルーター広告を生成するので、ホストは数分でルーターが利用できることを知ることができます。ただし、通知がないからといってルーターエラーであると判断できるほどの頻度ではありません。エラー検出には、近傍到達不能性を判別する別の検出アルゴリズムを利用します。

ルーター広告接頭辞

ルーター広告には、ホストがルーターと同じリンク上にいる (つまり、オンリンクである) かどうかを判断するときに使用するサブネット接頭辞のリストが含まれます。この接頭辞リストは、自動アドレス構成にも使用されます。接頭辞に付属するフラグは特定の接頭辞の使用目的を表します。ホストは通知されたオンリンク接頭辞を使用して、パケットの宛先がオンリンクであるか、あるいはルーターを越えているかを判断するためのリストを作成および管理します。通知されたオンリンク接頭辞になくても宛先がオンリンクの場合があります。この場合、ルーターはリダイレクトを送ることができます。リダイレクトは送信側に、宛先が近傍であることを知らせます。

ルーター広告と接頭辞別のフラグを使用すると、ルーターはステートレスアドレス自動構成を実行する方法をホストに伝えることができます。

ルーター広告メッセージ

ルーター広告メッセージには、ホストが発信するパケットに使用するインターネットパラメータ (ホップの制限など) も含めることができます。また、オプションでリンク MTU などのリンクパラメータも含めることができます。この機能により、重要なパラメータを集中管理できます。パラメータは、ルーターに設定され、関連付けられたすべてのホストに自動的に伝達されます。

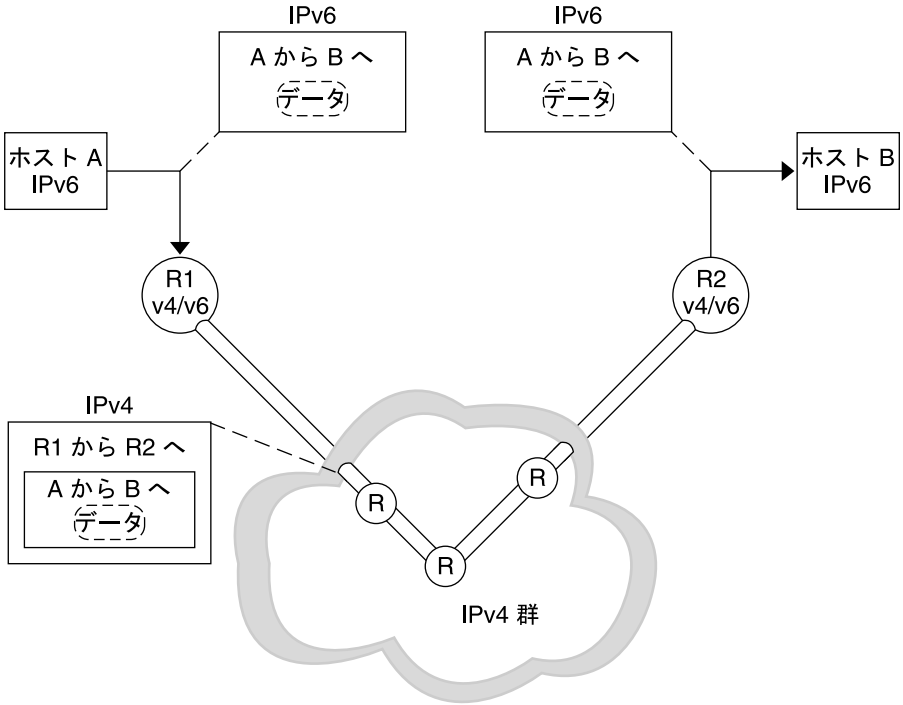
アドレス解決を行うために、ノードは、宛先ノードがリンク層アドレスを戻すように要求する近傍要請をマルチキャストグループに送信します。マルチキャストされた近傍要請メッセージは、宛先アドレスの要請先ノードのマルチキャストアドレスに送信されます。宛先は、そのリンク層アドレスをユニキャスト近傍通知

メッセージで戻します。発信元と宛先の両方に対して 1 つの要求応答パケットペアで互いのリンク層アドレスを処理できます。発信元は、近傍要請に発信元のリンク層アドレスを組み込みます。

IPv6 トンネル

デュアルスタック (IPv4 と IPv6 の両用サイト) における依存を最小限に抑えるため、2 つの IPv6 ノード間のパス中にあるすべてのルーターが IPv6 をサポートする必要はありません。このようなネットワーク構成をサポートするメカニズムのことを「トンネル」と呼びます。基本的に IPv6 パケットは IPv4 パケット内部に組み込まれ、IPv4 ルーター間を転送されます。次の図に、IPv6 ルーター (図中の "R") を通るトンネルメカニズムを示します。

図 11–5　IPv6 トンネルメカニズム



Oracle Solaris IPv6 実装には、2 つの種類のトンネルメカニズムがあります。

- 2 つのルーター間に構成されたトンネル (図 11–5 を参照)
- エンドポイントホストで終了する自動トンネル

作成されたトンネルは現在、インターネット上の、たとえば、MBONE (IPv4 マルチキャストバックボーン) で、ほかの目的に使用されています。構成トンネルの作成手順からいうと、2 つのルーターを構成して、その間に IPv4 ネットワーク経由の仮想ポイントツーポイントリンクを作成します。近い将来、インターネットのさまざまな局面に、この種のトンネルが利用されるでしょう。

自動トンネルには、IPv4 互換アドレスを必要とします。自動トンネルは、IPv6 ルーターが使用できない場合に IPv6 ノードを接続するために使用できます。このようなトンネルは、自動トンネルネットワークインタフェースを構成することによって、デュアルスタックホストまたはデュアルスタックルーターから作成できます。トンネルは常に、デュアルスタックホストで終了します。トンネルのはたらきにより、宛先 IPv4 アドレス (トンネルの終点) が IPv4 互換宛先アドレスから抽出されて動的に指定されます。

トンネルの構成

トンネルインタフェースのフォーマットは次のとおりです。

`ip.tun` *ppa*

ppa は物理的な接続ポイントです。

システムの起動時、トンネルモジュール (`tun`) は `ifconfig` コマンドによって IP の一番上にプッシュされ、仮想インタフェースが作成されます。このプッシュは、`hostname6.*` ファイルを作成することによって行われます。

たとえば、IPv4 ネットワーク経由で IPv6 パケットをカプセル化するためのトンネルを作成するには、次のファイルを作成します。

`/etc/hostname6.ip.tun0`

このファイルの内容は、インタフェースが plumb されたあとで、`ifconfig` に渡されます。ポイントツーポイントトンネルの構成に必要なパラメータになります。

例 11–11　IPv6 over IPv4 トンネルの `hostname6.ip.tun0` ファイル

次に、`hostname6.ip.tun0` ファイルのエントリの例を示します。

```
tsrc 10.10.10.23 tdst 172.16.7.19 up
addif 2001:db8:3b4c:1:5678:5678::2 up
```

この例では、IPv4 ソースアドレスと宛先アドレスは、IPv6 リンクローカルアドレスを自動構成するためのトークンとして使用されます。これらのアドレスは、`ip.tun0` インタフェースのソースと宛先です。次の 2 つのインタフェース `ip.tun0` インタ

フェースと、論理インタフェース ip.tun0:1 が構成されます。論理インタフェースには、addif コマンドによって指定されたソースと宛先 IPv6 アドレスがあります。

システムをマルチユーザーモードで起動すると、これらの構成ファイルの内容は変更されずに ifconfig に渡されます。例 11–11 内のエントリは、次と同等です。

```
# ifconfig ip.tun0 inet6 plumb
# ifconfig ip.tun0 inet6 tsrc 10.0.0.23 tdst 172.16.7.19 up
# ifconfig ip.tun0 inet6 addif 2001:db8:3b4c:1:5678:5678::2 up
```

次に、このトンネルにおける ifconfig -a の出力を示します。

```
ip.tun0: flags=2200850<UP,POINTOPOINT,RUNNING,MULTICAST,
  NONUD,IPv6> mtu 1480 index 6
        inet tunnel src 10.0.0.23  tunnel dst 172.16.7.19
        inet6 fe80::c0a8:6417/10 --> fe80::c0a8:713
ip.tun0:1: flags=2200850<UP,POINTOPOINT,RUNNING,MULTICAST,NONUD,IPv6> mtu 1480
  index 5
        inet6 2001:db8:3b4c:1:5678:5678::2
```

次の構文で構成ファイルに行を追加すれば、さらに論理インタフェースを構成できます。

```
addif IPv6-source IPv6-destination up
```

注 – トンネルのどちらかの端がトンネル経由で 1 つ以上の接頭辞を通知する IPv6 ルーターである場合、トンネル構成ファイルには addif コマンドは必要ありません。ほかのアドレスは自動構成されるため、必要とされる可能性があるのは tsrc と tdst だけです。

場合によっては、特定のトンネルについて、固有のソースリンクローカルアドレスと宛先リンクローカルアドレスを手動で構成する必要があることもあります。その場合、構成ファイルの最初の行を変更して、これらのリンクローカルアドレスを組み込みます。次に例を示します。

```
tsrc 10.0.0.23 tdst 172.16.7.19 fe80::1/10 fe80::2 up
```

ソースのリンクローカルアドレスの接頭辞の長さが 10 であることに注目してください。この例では、ip.tun0 インタフェースは次のようになります。

```
ip.tun0: flags=2200850<UP,POINTOPOINT,RUNNING,MULTICAST,NONUD,IPv6> mtu 1480
index 6
        inet tunnel src 10.0.0.23  tunnel dst 172.16.7.19
        inet6 fe80::1/10 --> fe80::2
```

IPv6 ネットワーク経由で IPv6 パケットをカプセル化するためのトンネル (IPv6 over IPv6 トンネル) を作成するには、次のファイル名を作成します。

```
/etc/hostname6.ip6.tun0
```

例 11–12　IPv6 over IPv6 トンネルの hostname6.ip6.tun0 ファイル

次に、IPv6 ネットワーク経由の IPv6 カプセル化における hostname6.ip6.tun0 ファイルのエントリの例を示します。

```
tsrc 2001:db8:3b4c:114:a00:20ff:fe72:668c
        tdst 2001:db8:15fa:25:a00:20ff:fe9b:a1c3
fe80::4 fe80::61 up
```

IPv6 ネットワーク経由で IPv4 パケットをカプセル化するためのトンネル (IPv4 over IPv6 トンネル) を作成するには、次のファイル名を作成します。

```
/etc/hostname.ip6.tun0
```

例 11–13　IPv4 over IPv6 トンネルの hostname.ip6.tun0 ファイル

次に、IPv6 ネットワーク経由の IPv4 カプセル化における hostname.ip6.tun0 ファイルのエントリの例を示します。

```
tsrc 2001:db8:3b4c:114:a00:20ff:fe72:668c
         tdst 2001:db8:15fa:25:a00:20ff:fe9b:a1c3
10.0.0.4 10.0.0.61 up
```

IPv4 ネットワーク経由で IPv4 パケットをカプセル化するためのトンネル (IPv4 over IPv4 トンネル) を作成するには、次のファイル名を作成します。

```
/etc/hostname.ip.tun0
```

例 11–14　IPv4 over IPv4 トンネルの hostname.ip.tun0 ファイル

次に、IPv4 ネットワーク経由の IPv4 カプセル化における hostname.ip.tun0 ファイルのエントリの例を示します。

```
tsrc 172.16.86.158 tdst 192.168.86.122
10.0.0.4 10.0.0.61 up
```

tun の固有の情報については、tun(7M) のマニュアルページを参照してください。IPv6 への移行期間におけるトンネルの一般的な概念については、85 ページの「IPv6 トンネルの概要」を参照してください。トンネルを構成する手順については、195 ページの「IPv6 サポート用にトンネルを構成するためのタスク (タスクマップ)」を参照してください。

6to4 自動トンネル

Oracle Solaris には、IPv4 から IPv 6 アドレス指定への移行期間における暫定的な優先方法として、6to4 トンネルがあります。6to4 トンネルを使用すると、孤立した IPv6 サイトが、IPv6 をサポートしていない IPv4 ネットワーク上の自動トンネルを通じて通信できるようになります。6to4 トンネルを使用するには、6to4 自動トンネルの片方のエンドポイントとして、境界ルーターを IPv6 ネットワークに構成する必要があります。そのあと、この 6to4 ルーターをほかの 6to4 サイトとの間のトンネルの構成要素として使用することも、あるいは必要に応じて 6to4 以外のネイティブ IPv6 サイトとの間のトンネルで使用することもできます。

この節では、6to4 に関連した次の参考情報を示します。

- 6to4 トンネルのトポロジ
- 6to4 アドレス指定 (通知の書式など)
- 6to4 トンネルを介したパケットフローの説明
- 6to4 ルーターと 6to4 リレールーター間のトンネルのトポロジ
- 6to4 リレールーターサポートを構成する前の考慮事項

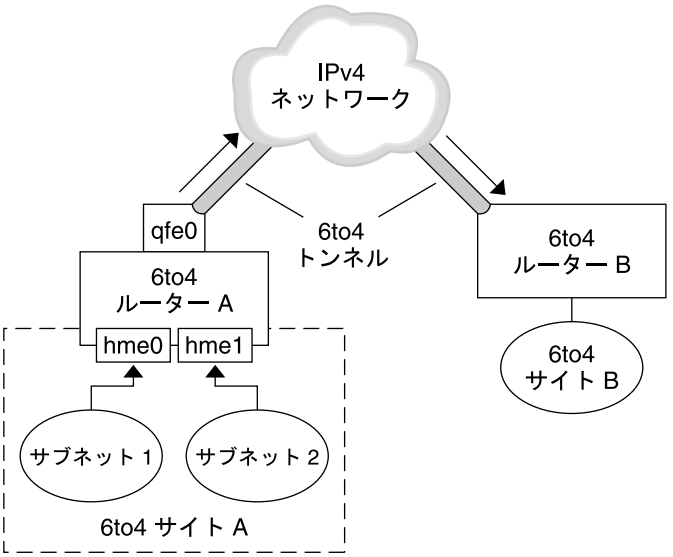
次の表では、6to4 トンネルを構成するための追加タスクについて説明し、有用な追加情報の入手先を示しています。

タスクまたは詳細	参照先
6to4 トンネルの構成タスク	199 ページの「6to4 トンネルを構成する方法」
6to4 関連の RFC	RFC 3056, "Connection of IPv6 Domains via IPv4 Clouds" (`http://www.ietf.org/rfc/rfc3056.txt`)
6to4 リレールーターとの間のトンネルのサポートを有効にする `6to4relay` コマンドの詳細	`6to4relay`(1M)
6to4 のセキュリティー	Security Considerations for 6to4 (`http://www.ietf.org/rfc/rfc3964.txt`)

6to4 トンネルのトポロジ

6to4 トンネルは、あらゆる場所にあるすべての 6to4 サイトに IPv6 接続を提供します。同様に、リレールーターに転送するようにトンネルが構成されている場合、トンネルはネイティブ IPv6 インターネットも含むすべての IPv6 サイトへのリンクとしても機能します。次の図は、6to4 トンネルが 6to4 サイト間にこの接続を提供する仕組みを示しています。

図 11–6　2 つの 6to4 サイト間のトンネル



この図には、孤立した 2 つの 6to4 ネットワーク、サイト A とサイト B が描かれています。各サイトでは、IPv4 ネットワークへの外部接続を備えたルーターが構成されています。IPv4 ネットワークを越える 6to4 トンネルによって、6to4 サイトをリンクする接続が提供されています。

IPv6 サイトを 6to4 サイトにするには、6to4 をサポートできるように 1 つ以上のルーターインタフェースを構成する必要があります。このインタフェースは、IPv4 ネットワークに対する外部接続を提供する必要があります。`qfe0` で構成するアドレスは、一意 (世界で唯一) のものでなければなりません。次の図では、境界ルーター A のインタフェース `qfe0` がサイト A を IPv4 ネットワークに接続しています。`qfe0` を 6to4 擬似インタフェースとして構成するには、IPv4 アドレスを使用してあらかじめインタフェース `qfe0` を構成しておきます。

この図の 6to4 サイト A は、ルーター A のインタフェース `hme0` と `hme1` に接続された 2 つのサブネットから構成されています。サイト A のいずれかのサブネット上の IPv6 ホストはすべて、ルーター A からの広告の受信時に 6to4 派生アドレスで自動的に再構成されます。

サイト B は、もう 1 つの独立した 6to4 サイトです。サイト A からトラフィックを正しく受け取るには、サイト B 側の境界ルーターを 6to4 をサポートするように構成する必要があります。それ以外の場合、ルーターがサイト A から受け取るパケットが認識されずに削除されてしまいます。

6to4 トンネルを介したパケットフロー

このセクションでは、ある 6to4 サイトにあるホストから、リモートの 6to4 サイトにあるホストまでのパケットのフローについて説明します。このシナリオでは、図 11–6 に示すトポロジを使用します。さらにこのシナリオは、6to4 ルーターと 6to4 ホストがすでに構成済みであることを想定しています。

1. 6to4 サイト A のサブネット 1 に存在するホストが伝送を行い、6to4 サイト B 上のホストが宛先として機能します。各パケットヘッダーには、送信元の 6to4 派生アドレスと宛先の 6to4 派生アドレスが含まれます。
2. サイト A のルーターは、IPv4 ヘッダー内で各 6to4 パケットをカプセル化します。このプロセスでルーターは、カプセル化ヘッダーの IPv4 宛先アドレスを、サイト B のルーターアドレスに設定します。トンネルインタフェースを通過する各 IPv6 パケットの IPv6 宛先アドレスには、この IPv4 宛先アドレスも含まれています。したがって、ルーターはカプセル化ヘッダーに設定されている IPv4 宛先アドレスを特定することができます。続いてサイト A のルーターは、標準の IPv4 経路制御手続きを使用し IPv4 ネットワークを介してこのパケットを転送します。
3. パケットが遭遇する IPv4 ルーターが、パケットの IPv4 宛先アドレスを使用して転送を行います。このアドレスはルーター B のインタフェースに使用される一意の (世界に 1 つしかない) IPv4 アドレスであり、6to4 擬似インタフェースとしても機能します。
4. サイト A から送付されたパケットがルーター B に到着します。ルーター B は、IPv4 ヘッダーを削除して IPv6 パケットのカプセル化を解除します。
5. 続いてルーター B は、IPv6 パケット内の宛先アドレスを使用してサイト B の受信ホストにパケットを転送します。

6to4 リレールーターとの間のトンネルについての考慮事項

6to4 リレールーターは、6to4 ではない ネイティブ IPv6 ネットワークと通信を行う必要がある 6to4 ルーターからのトンネルのエンドポイントとして機能します。本来、リレールーターは 6to4 サイトとネイティブ IPv6 サイトとの間のブリッジとして使用されます。この手法は安全ではない場合があるため、Oracle Solaris のデフォルト設定では 6to4 リレールーターのサポートは無効になっています。しかし、サイトでこのようなトンネルが必要な場合には `6to4relay` コマンドを使用して次に示すようなトンネリングを有効にできます。

図 11–7　6to4 サイトと 6to4 リレールーター間のトンネル

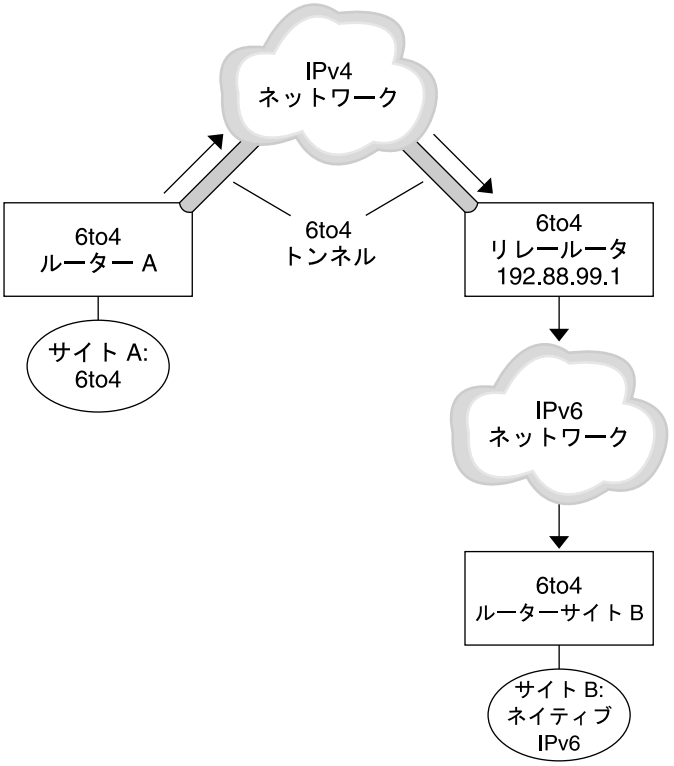


図 11–7 の 6to4 サイト A は、ネイティブ IPv6 サイト B のノードと通信する必要があります。図には、IPv4 ネットワーク経由でサイト A から 6to4 トンネルに向かうトラフィックのパスが示されています。このトンネルは、6to4 ルーター A と 6to4 リレールーターをエンドポイントとして使用しています。6to4 リレールーターより先は IPv6 ネットワークであり、IPv6 サイト B はこのネットワークに接続されています。

6to4 サイトとネイティブ IPv6 サイト間のパケットフロー

このセクションでは、6to4 サイトからネイティブな IPv6 サイトまでのパケットフローについて説明します。このシナリオでは、図 11–7 に示すトポロジを使用します。

1. 6to4 サイト A のホストが、ネイティブ IPv6 サイト B のホストを宛先に指定して伝送を行います。各パケットヘッダーの発信元アドレスには 6to4 派生アドレスが含まれています。宛先アドレスは標準の IPv6 アドレスです。

2. サイト A の 6to4 ルーターは、各パケットを宛先である 6to4 ルーターの Ipv4 アドレスを持つ IPv4 ヘッダー内でカプセル化します。この 6to4 ルーターは、標準の IPv4 経路制御手続きを使用し IPv4 ネットワークを介してこのパケットを転送します。パケットが遭遇する IPv4 ルーターが、6to4 リレールーターにパケットを転送します。
3. サイト A に物理的にもっとも近いエニーキャスト 6to4 リレールーターが、`192.88.99.1` エニーキャストグループ宛てのパケットを検出します。

注 – 6to4 リレールーターエニーキャストグループの一部である 6to4 リレールーターには、`192.88.99.1` という IP アドレスが割り当てられます。このエニーキャストアドレスは、6to4 リレールーターのデフォルトアドレスです。特定の 6to4 リレールーターを使用する必要がある場合は、デフォルトをオーバーライドしてそのルーターの IPv4 アドレスを指定できます。

---

4. このリレールーターは、IPv4 ヘッダーを取り除いて 6to4 パケットのカプセル化を解除し、ネイティブ IPv6 宛先アドレスを明らかにします。
5. 次に、リレールーターが IPv6 のみとなったパケットを IPv6 ネットワークに送信し、そこでサイト B のルーターがそのパケットを最終的に受け取ります。次に、ルーターがそのパケットを宛先の IPv6 ノードに転送します。

Oracle Solaris ネームサービスに対する IPv6 拡張機能

このセクションでは、IPv6 の実装によって導入されたネームサービスの変更について説明します。IPv6 アドレスは、どの Oracle Solaris ネームサービス(NIS、LDAP、DNS、およびファイル) にも格納できます。また、NIS over IPv6 RPC トランスポートを使用すると、NIS データを検出できます。

IPv6 の DNS 拡張機能

IPv6 固有なリソースレコードである AAAA リソースレコードについては、RFC 1886、DNS Extensions to Support IP Version 6を参照してください。この AAAA レコードは、ホスト名を 128 ビット IPv6 アドレスにマップします。PTR レコードは IPv6 でも、IP アドレスをホスト名にマップするときに使用されています。128 ビットアドレスの 32 の 4 ビットニブルは、IPv6 アドレス用に反転されています。各ニブルは対応する 16 進 ASCII 値に変換されます。変換後、`ip6.int` が追加されます。

nsswitch.conf ファイルへの変更

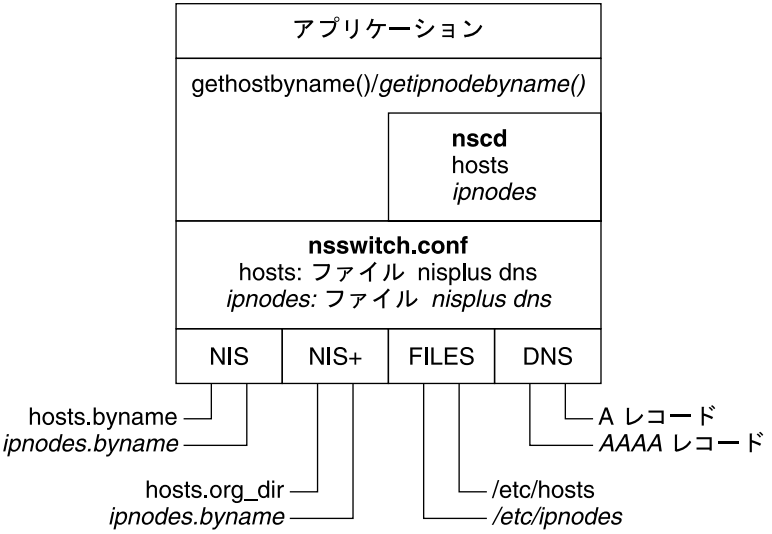
Solaris 10 11/06 以前のリリースでは、/etc/inet/ipnodes で IPv6 アドレスを調べる機能に加え、NIS、LDAP、DNS の各ネームサービスにも IPv6 サポートが追加されています。結果として、nsswitch.conf ファイルは、IPv6 検索をサポートするように変更されました。

```
hosts:  files dns nisplus [NOTFOUND=return]
ipnodes: files dns nisplus [NOTFOUND=return]
```

注 – IPv4 アドレスと IPv6 アドレスでこれらの ipnodes データベースを生成してから、複数のネームサービスで ipnodes を探すように /etc/nsswitch.conf ファイルを変更してください。ホストアドレスの解決時に不要な遅延が発生してしまうからです (ブートタイミングの遅れが発生することもあります)。

次の図に、nsswitch.conf ファイルと、gethostbyname コマンドと getipnodebyname コマンドを使用するアプリケーション用の新しいネームサービスデータベースの間の新しい関係を示します。斜体の項目は新規です。gethostbyname コマンドは、/etc/inet/hosts に保存されている IPv4 アドレスだけを調べます。Solaris 10 11/06 以前のリリースでは、getipnodebyname コマンドは、nsswitch.conf ファイルの ipnodes エントリで指定したデータベースを調べます。検索に失敗すると、nsswitch.conf ファイルの hosts エントリで指定したデータベースを調べます。

図 11–8 nsswitch.conf とネームサービスの関係



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

ネームサービスコマンドの変更

IPv6 をサポートするため、IPv6 アドレスは既存のネームサービスコマンドを使用して検索できます。たとえば、`ypmatch` コマンドは、新しい NIS マップに使用できます。`nslookup` コマンドでは、DNS の新しい AAAA レコードを調べることができます。

NFS と RPC による IPv6 のサポート

NFS ソフトウェアとリモート手続き呼出し (RPC) ソフトウェアは、同じような方法で IPv6 をサポートします。NFS サービスに関連のある既存のコマンドは変更されていません。ほとんどの RPC アプリケーションが、変更なしで IPv6 で実行できます。トランスポート機能のある一部の高度 RPC アプリケーションに更新が必要な場合があります。

IPv6 over ATM のサポート

Oracle Solaris は、IPv6 経由の ATM、固定仮想回路 (PVC)、静的な交換仮想回路 (SVC) をサポートするようになりました。

パート III

DHCP

このパートには、Dynamic Host Configuration Protocol (DHCP) の概念的情報と、DHCP サービスの計画、構成、管理、問題追跡のタスクが含まれています。

◆ ◆ ◆ 第 12 章

DHCP について (概要)

この章では、Dynamic Host Configuration Protocol (DHCP) の概要とそのプロトコルを支える概念について説明します。さらに、DHCP をネットワークで使用することの利点についても述べます。

この章では、次の内容について説明します。

DHCP プロトコルについて

DHCP プロトコルを使用すれば、TCP/IP ネットワーク上のホストシステムを、システムのブート時に、そのネットワークに合わせて自動的に構成できます。DHCP では、クライアント/サーバーメカニズムが使用されます。サーバーは、クライアントの構成情報を格納、管理し、クライアントの要求に応じてその構成情報を提供します。構成情報には、クライアントの IP アドレスと、クライアントが使用可能なネットワークサービス情報が含まれます。

DHCP は、従来の BOOTP プロトコルをベースに機能拡張されたプロトコルです。BOOTP は、TCP/IP ネットワーク経由のブートを可能にすることを目的に設計されました。クライアントとサーバー間のメッセージの形式は、DHCP の場合も BOOTP の場合も同じです。ただし、DHCP メッセージには、BOOTP メッセージとは異なり、クライアント用のネットワーク構成データを組み込むことができます。

DHCP の主な利点は、リースを通して IP アドレス割り当てを管理できることです。「リース」を使用すれば、使用されていない IP アドレスを取り戻すことができます。取り戻された IP アドレスは、ほかのクライアントに割り当てられます。そのため、DHCP を使用する 1 つのサイト用の IP アドレスプールは、すべてのクライアントに常時 IP アドレスを割り当てた場合に比べて、小さくなります。

DHCP を使用することの利点

DHCP は、TCP/IP ネットワークの設定やネットワークの日々の管理に伴う時間のかかるタスクを部分的に軽減します。Oracle Solaris の実装環境では、DHCP は IPv4 でのみ動作します。

DHCP には、次の利点があります。

- **IP** アドレス管理 – DHCP の主な利点は、IP アドレスをより簡単に管理できることです。DHCP を使用しないネットワークでは、IP アドレスを手動で割り当てる必要があります。個々のクライアントに固有の IP アドレスを割り当て、クライアントを個別に構成するためには、慎重な作業が必要です。さらに、クライアントが別のネットワークに移動したら、そのクライアントのために手動で修正を加える必要があります。DHCP が使用可能な場合は、管理者が介在しなくても、DHCP サーバーが IP アドレスを管理し、割り当てます。クライアントは、別のサブネットに移動する際に新しいネットワークに適した新しいクライアント情報を DHCP サーバーから取得するため、手動による再構成は必要ありません。
- 一元的なネットワーククライアントの構成 – 構成は、クライアントまたはクライアントのタイプに合わせてカスタマイズできます。構成情報は、同じ場所 (DHCP データストア) に格納されます。したがって、クライアントの構成を変更するためにクライアントにログインする必要はありません。データストア内の情報を変更するだけで、複数のクライアントに対する変更を実行できます。
- **BOOTP** クライアントのサポート – BOOTP サーバーと DHCP サーバーはどちらも、クライアントからのブロードキャストを待機して、応答します。DHCP サーバーは、DHCP クライアントからの要求だけではなく、BOOTP クライアントからの要求にも応答できます。BOOTP クライアントは、IP アドレスと、ブートに必要な情報をサーバーから受け取ります。
- ローカルおよびリモートクライアントのサポート – BOOTP は、あるネットワークから別のネットワークへのメッセージリレー (中継) 機能を備えています。DHCP は、さまざまな方法で BOOTP リレー機能を使用します。ほとんどのネットワークルーターは、BOOTP リレーエージェントとして機能するように構成できます。そのように構成されたネットワークルーターは、要求側クライアントのネットワーク上に存在しないサーバーに BOOTP 要求を渡します。同じ方法で、DHCP 要求をリレーすることも可能です。これは、ルーターには DHCP 要求と BOOTP 要求の区別がないためです。また、BOOTP リレー機能をサポートするルーターが使用できない場合には、DHCP サーバーを BOOTP リレーエージェントとして動作するように構成することもできます。
- ネットワークブート機能 – クライアントは、DHCP を使用すると、RARP (逆アドレス解決プロトコル) や `bootparams` ファイルを使用しなくても、ネットワーク上のサーバーからブートに必要な情報を取得できます。DHCP サーバーは、IP アドレス、ブートサーバー、ネットワーク構成情報を含む、クライアントが動作するのに必要なすべての情報をクライアントに提供できます。DHCP 要求は、サブネットを越えてリレーできるので、DHCP ネットワークブート機能を使用すれ

ば、ネットワーク内のブートサーバー数を削減できます。RARP でのブートには、サブネットごとにブートサーバーが必要です。

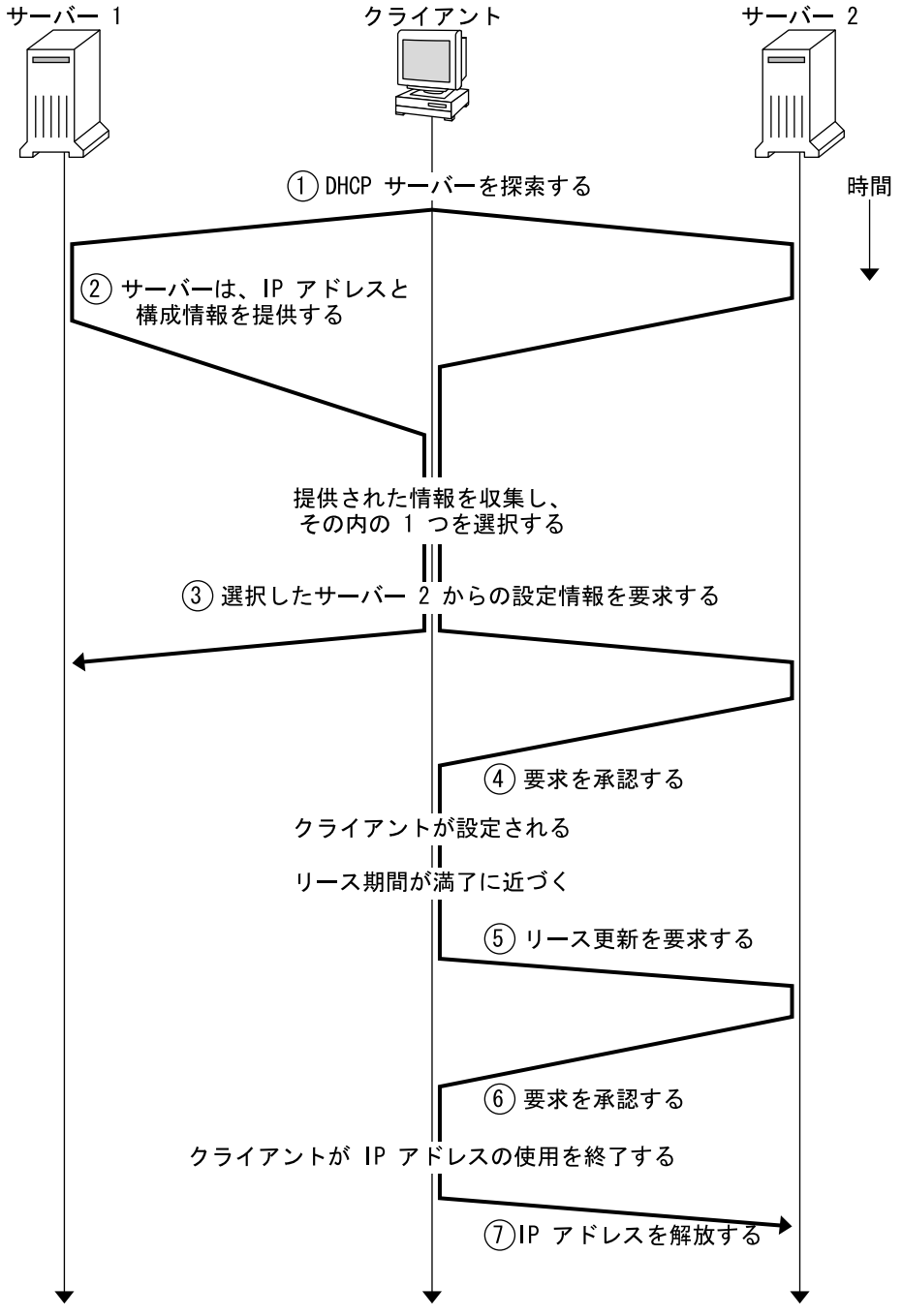
- 大規模ネットワークのサポート – 何百万という DHCP クライアントをもつネットワークでも DHCP を使用できます。DHCP サーバーは、マルチスレッド機能を使って多数のクライアント要求を同時に処理します。さらに、大量データを処理できるように最適化されたデータストアをサポートします。データストアアクセスは、別個の処理モジュールによって行われます。このようなデータストアアプローチでは、データベースが必要になるたびにそのサポートを追加できます。

DHCP の動作

まず始めに、DHCP サーバーのインストールと構成を行う必要があります。構成作業では、クライアントがネットワーク上で機能するために必要なネットワーク情報を指定します。この情報が正しく設定されると、クライアントはネットワーク情報を要求し、受け取ることができます。

次の図は、DHCP サービスにおける一連のイベントを示したものです。丸の中の番号は、図のあとに続く説明の箇条書き番号を示しています。

図 12–1 DHCP サービスにおける一連のイベント



上の図には、次の手順が示されています。

1. クライアントは、ローカルサブネット上で制限付きブロードキャストアドレス (255.255.255.255) に「検索メッセージ」を送信することで、DHCP サーバーを検索します。ルーターが存在し、BOOTP リレーエージェントとして動作するように構成されている場合、要求は異なるサブネット上の別の DHCP サーバーに渡されます。クライアントの「ブロードキャスト」にはクライアント固有の ID が含まれています。この ID は、Oracle Solaris の DHCP 実装環境の場合、クライアントの MAC (Media Access Control) アドレスから派生します。Ethernet ネットワークでは、MAC アドレスは Ethernet アドレスと同じです。

   検索メッセージを受け取った DHCP サーバーは、次の情報からクライアントのネットワークを特定します。

   - この要求がどのネットワークインタフェースから入ってきたか。これによってサーバーは、クライアントが、インタフェースが接続されているネットワーク上にあるのか、あるいはそのネットワークに接続された BOOTP リレーエージェントを使用しているのかがわかります。
   - BOOTP リレーエージェントの IP アドレスが要求に含まれているか。要求がリレーエージェントを通過する際に、リレーエージェントは要求ヘッダーにリレーエージェントのアドレスを挿入します。サーバーが「リレーエージェントのアドレス」を検出すると、サーバーは、そのアドレスのネットワーク部分がクライアントのネットワークアドレスを示していることを認識します。これは、リレーエージェントがクライアントのネットワークに接続されている必要があるからです。
   - クライアントのネットワークは、サブネット化されているか。サーバーは、リレーエージェントのアドレス、または要求を受け取ったネットワークインタフェースのアドレスが示すネットワークのサブネットマスクを `netmasks` テーブルから見つけます。サーバーは、使用されているサブネットマスクを認識すると、ネットワークアドレスのどの部分がホスト部分であるかを特定し、クライアント用の適切な IP アドレスを選択できます。`netmasks` については、`netmasks(4)` のマニュアルページを参照してください。

2. DHCP サーバーは、クライアントのネットワークを特定すると、適切な IP アドレスを選択し、そのアドレスがまだ使用されていないことを確認します。次に DHCP サーバーは、「オファーメッセージ」を送信し、そのクライアントに応答します。オファーメッセージには、選択された IP アドレスと、クライアントの構成に使用できるサービスの情報が含まれています。サーバーは、この IP アドレスを使用するかどうかをクライアントが決めるまで、これを一時的に予約します。
3. クライアントは、オファーされたサービスの数とタイプに基づいて最善のオファーを選択します。そして、最善のオファーとなったサーバーの IP アドレスを求める要求を送信します。この伝送によって、クライアントがサーバーを選択したことを、応答中のすべての DHCP サーバーに知らせることができます。選択されなかったサーバーは、オファーした IP アドレスの予約を取り消します。

4. 選択されたサーバーは、クライアント用の IP アドレスを割り当て、その情報を DHCP データストアに格納します。そして、承認メッセージ (ACK) をクライアントに送信します。「承認メッセージ」には、クライアントのためのネットワーク構成パラメータが含まれています。クライアントは、`ping` ユーティリティーを使って IP アドレスをテストし、ほかのシステムがそれを使っていないか確かめます。そして、ブートを続行しネットワークに参加します。
5. クライアントはリース時間を監視します。設定された時間が経過すると、クライアントは、さきほど選択したサーバーに新しいメッセージを送信してリースを増やそうとします。
6. 要求を受け取った DHCP サーバーは、リース期間と、管理者が規定したローカルリースポリシーとが合っていれば、そのリース期間を延長します。サーバーが 20 秒以内に応答しない場合、クライアントは、ほかの DHCP サーバーのいずれかがリース期間を延長できるように要求をブロードキャストします。
7. クライアントは、その IP アドレスが不要になると、IP アドレスが解放されたことをサーバーに知らせます。この通知は、通常のシャットダウンの際に実行され、また手動で実行することも可能です。

DHCP サーバー

DHCP サーバーは、ホストシステム上で Oracle Solaris のデーモンとして動作します。サーバーは、2 つの基本機能を備えています。

- **IP** アドレスの管理 – DHCP サーバーは、IP アドレスの範囲を制御し、常時または定義した期間、IP アドレスをクライアントに割り当てます。サーバーはリースメカニズムを使って、クライアントが一時的なアドレスを使用できる期間を決めます。アドレスは、不要になるとプールに返され、再割り当てされます。DHCP サーバーは、DHCP ネットワークテーブル内にクライアントへの IP アドレス結合情報を保持し、複数のクライアントが同じアドレスを使用しないようにします。
- クライアントにネットワーク構成情報を提供 – サーバーは、クライアントの IP アドレスを割り当て、ホスト名やブロードキャストアドレス、ネットワークサブネットマスク、デフォルトゲートウェイ、ネームサービスといったネットワーク構成情報をクライアントに提供します。ネットワーク構成情報は、サーバーの `dhcptab` データベースから取得されます。

また、 DHCP サーバーは次の追加機能を実行するように構成することも可能です。

- **BOOTP** クライアント要求への応答 – サーバーは、BOOTP サーバーを検索する BOOTP クライアントからのブロードキャストを待機し、BOOTP クライアントに IP アドレスとブートパラメータを提供します。管理者は、これらの情報をあらかじめ静的に構成しておく必要があります。DHCP サーバーは、同時に BOOTP サーバーとしても DHCP サーバーとしても動作できます。
- 要求のリレー – サーバーは、ほかのサブネット上の適切なサーバーに BOOTP 要求と DHCP 要求をリレーします。DHCP サーバーは BOOTP リレーエージェントとして構成された場合、DHCP サービスや BOOTP サービスを提供できなくなります。
- **DHCP** クライアントにネットワークブート情報を提供 – サーバーは、DHCP クライアントがネットワーク経由でブートするために必要な情報を DHCP クライアントに提供できます。 この情報には、IP アドレスやブートパラメータ、ネットワーク構成情報などがあります。さらにサーバーは、広域ネットワーク (WAN) を介したブートやインストールに必要な情報を DHCP クライアントに提供します。
- ホスト名を指定したクライアントに代わって **DNS** テーブルを更新 – DHCP サービスを求めるクライアントの要求に `Hostname` オプションと値が含まれている場合には、DHCP サーバーが、クライアントに代わって DNS を更新できます。

DHCP サーバーの管理

スーパーユーザーであれば、DHCP マネージャやコマンド行ユーティリティーを使って DHCP サーバーの起動、停止、構成を行うことができます (コマンド行ユーティリティーについては、318 ページの「DHCP コマンド行ユーティリティー」を参照)。通常、DHCP サーバーは、システムのブート時に自動的に起動され、システムのシャットダウン時に自動的に終了するように構成されています。したがって、通常は、サーバーの起動や終了を手動で行う必要はありません。

DHCP データストア

DHCP サーバーが使用するデータはすべてデータストアに保持されます。データストアの内容は、プレーンテキストファイル、NIS+ テーブル、またはバイナリ形式ファイルの場合があります。どの形式のデータストアを使用するかは、DHCP サービスを構成するときに選択されます。データストアのタイプによる違いについては、331 ページの「DHCP データストアの選択」を参照してください。DHCP マネージャや `dhcpconfig` コマンドを使ってデータストアの形式を変換することはできません。

ある DHCP サーバーのデータストアから別のサーバーのデータストアにデータを移動できます。それらのサーバーが、異なるデータストア形式を使用している場合で

も、それらのデータストアを扱うエクスポートやインポートユーティリティーが使用できます。DHCP マネージャか dhcpconfig コマンドを使って、データストアの内容全体またはその一部をエクスポートしたりインポートしたりできます。

注 – DHCP (サーバーツールと管理ツール) とデータベース間のインタフェースになる独自のコードモジュールを開発すれば、データベースやファイルのフォーマットはどのようなものでもDHCP データ領域に使用できます。詳細は、『Solaris DHCP サービス開発ガイド』を参照してください。

DHCP データストア内には、2 つのタイプのテーブルがあります。これらのテーブルの内容を表示したり管理したりするには、DHCP マネージャかコマンド行ユーティリティーを使用します。データテーブルの種類は次のとおりです。

- dhcptab テーブル – クライアントに提供することが可能な構成情報が入っています。
- **DHCP** ネットワークテーブル – テーブル名が示すネットワーク上にある DHCP クライアントや BOOTP クライアントの情報が含まれています。たとえば、ネットワーク 192.168.32.0 のテーブル名には、192_168_32_0 が含まれています。

dhcptab テーブル

dhcptab テーブルには、クライアントが DHCP サーバーから取得できるすべてのデータが含まれています。DHCP サーバーは、起動されるたびに dhcptab テーブルをスキャンします。dhcptab テーブルのファイル名は、使用されるデータストアによって異なります。たとえば、NIS+ データストア SUNWnisplus によって作成された dhcptab テーブルは SUNWnisplus1_dhcptab になります。

DHCP プロトコルは、クライアントに渡すことができる情報の標準的な項目を多数定義しています。これらの項目は、パラメータ、シンボル、またはオプションと呼ばれます。DHCP プロトコルでは、オプションは数値コードとテキストラベルで定義されており、値は与えられていません。例として、一般的に使用される標準オプションの一部を示します。

表 12–1　DHCP 標準オプションの例

コード	ラベル	説明
1	Subnet	サブネットマスク IP アドレス
3	Router	ルーターの IP アドレス
6	DNSserv	DNS サーバーの IP アドレス
12	Hostname	クライアントホスト名を表すテキスト文字列
15	DNSdmain	DNS ドメイン名

オプションの中には、サーバーの構成中に情報が提供されると、自動的に値が割り当てられるものがあります。また、あとで、ほかのオプションに値を明示的に割り当てることもできます。オプションとその値はクライアントに渡され、構成情報を形成します。たとえば、オプションと値のペアである DNSdmain=Georgia.Peach.COM は、クライアントの DNS ドメイン名を Georgia.Peach.COM に設定します。

オプションは、「マクロ」として知られているコンテナ内でほかのオプションと共にグループ化することができ、これによりクライアントへ容易に情報を渡すことができます。マクロの中には、サーバー構成時に自動的に作成され、構成時に値が割り当てられるオプションを含むものがあります。また、マクロにはほかのマクロを含めることもできます。

dhcptab テーブルのフォーマットについては、dhcptab(4) のマニュアルページを参照してください。DHCP マネージャでは、「オプション (Options)」タブや「マクロ (Macros)」タブに示されるすべての情報は dhcptab ファイルから得られます。オプションについては、321 ページの「DHCP オプションについて」を参照してください。マクロについては、322 ページの「DHCP マクロについて」を参照してください。

dhcptab テーブルをテキストエディタで編集しないでください。オプションやマクロの作成、削除、変更には、dhtadm コマンドまたは DHCP マネージャを使用する必要があります。

DHCP ネットワークテーブル

DHCP ネットワークテーブルは、クライアントの識別子を IP アドレスと、各アドレスに関連した構成パラメータに対応付けます。ネットワークテーブルの書式については、dhcp_network(4) のマニュアルページを参照してください。DHCP マネージャでは、「アドレス (Addresses)」タブに示されるすべての情報はネットワークテーブルから得られます。

DHCP マネージャ

DHCP マネージャは、DHCP サービスに関連するすべての管理業務を行うためのグラフィカルユーザーインタフェース (GUI) ツールです。このツールを使用すると、サーバーそのものだけでなく、サーバーが使用するデータも管理できます。DHCP マネージャを実行するためにはスーパーユーザーでなければなりません。

サーバー上では DHCP マネージャを下記の場合に使用できます。

- DHCP サーバーを構成または構成解除する場合
- DHCP サーバーを起動、停止、および再起動する場合
- DHCP サービスを有効または無効にする場合
- DHCP サーバーの設定をカスタマイズする場合

さらに、DHCP マネージャでは、IP アドレスやネットワーク構成マクロ、ネットワーク構成オプションに関して次のことができます。

- DHCP 管理下にあるネットワークの追加や削除
- DHCP 管理下にある IP アドレスの表示、追加、変更、削除、解放
- ネットワーク構成マクロの表示、追加、変更、削除
- 標準以外のネットワーク構成オプションの表示、追加、変更、削除

DHCP マネージャでは、DHCP データストアに関して次のことができます。

- データを新しいデータストアフォーマットに変換する。
- DHCP データをある DHCP サーバーから別のサーバーに移動する。 データを最初のサーバーからエクスポートし、次のサーバーにインポートする必要があります。

DHCP マネージャでは、実行できる手順についての詳細なオンラインヘルプも利用できます。詳細は、356 ページの「DHCP マネージャーについて」を参照してください。

DHCP コマンド行ユーティリティー

すべての DHCP 管理機能は、コマンド行ユーティリティーを使用して実行できます。コマンド行ユーティリティーを実行するには、スーパーユーザーとして、または DHCP 管理プロファイルに割り当てられているユーザーでログインしている必要があります。詳細は、359 ページの「DHCP コマンドへのユーザーアクセスの設定」を参照してください。

次の表に、各ユーティリティーとその使用目的を示します。

表 12–2 DHCP コマンド行ユーティリティー

コマンド	説明と使用目的	マニュアルページへのリンク
`in.dhcpd`	DHCP サービスデーモン。コマンド行引数を使えば、いくつかの実行時オプションを設定できます。	`in.dhcpd(1M)`
`dhcpconfig`	DHCP サーバーの構成や構成解除に使用します。このユーティリティーでは、DHCP マネージャの多くの機能をコマンド行から実行できます。このユーティリティーは主に、一部の構成機能を自動化したいときにスクリプト中で使用します。`dhcpconfig` は、サーバーシステムのネットワークトポロジファイルから情報を収集し、初期構成に必要な情報を作成します。	`dhcpconfig(1M)`

表 12–2 DHCP コマンド行ユーティリティー (続き)

コマンド	説明と使用目的	マニュアルページへのリンク
dhtadm	DHCP クライアント用の構成オプションやマクロを追加、削除、変更するときに使用します。このユーティリティーによって dhcptab テーブルが間接的に編集され、dhcptab テーブルのフォーマットが正しく保たれます。dhcptab テーブルを直接編集してはいけません。	dhtadm(1M)
pntadm	DHCP ネットワークテーブルの管理に使用します。このユーティリティーで実行可能なタスクは、次のとおりです。 ■ DHCP の管理下にある IP アドレスやネットワークを追加、削除する。 ■ 指定する IP アドレスのネットワーク構成を変更する。 ■ DHCP の管理下にある IP アドレスやネットワークの情報を表示する。	pntadm(1M)

役割によるアクセス制御 (RBAC) - DHCP コマンドを使用する場合

dhcpconfig、dhtadm、pntadm コマンドのセキュリティーは、役割によるアクセス制御 (RBAC, Role-Based Access Control) の設定値に基づいて決められます。デフォルトでは、これらのコマンドを実行できるのはスーパーユーザーだけです。別のユーザー名の下でコマンドを使用する場合は、ユーザー名を DHCP 管理プロファイルに割り当てる必要があります (359 ページの「DHCP コマンドへのユーザーアクセスの設定」を参照)。

DHCP サーバーの構成

DHCP サーバーを動作させるシステム上で DHCP マネージャを初めて実行するときに、DHCP サーバーを構成します。

DHCP マネージャのサーバー構成ダイアログボックスに、1 つのネットワーク上で DHCP サーバーを使用可能にして実行するために必要な基本情報を入力するように要求するメッセージが表示されます。既存のシステムファイルからいくつかのデフォルト値を取得できます。そのネットワークに対してシステムを構成していない場合には、デフォルト値はありません。DHCP マネージャは下記の情報を入力するように促します。

- そのサーバーの役割 : DHCP サーバーまたは BOOTP リレーエージェントのいずれか
- データストアの形式 (ファイル、バイナリファイル、NIS+、または独自のもの)

- 選択したデータストアタイプに対するデータストア構成パラメータ
- ホストレコードの更新に使用するネームサービス (使用する場合) (/etc/hosts、NIS+、または DNS)
- リース期間と、クライアントがリース期間を更新できるようにするかどうか
- DNS サーバーの DNS ドメイン名および IP アドレス
- DHCP サービス用に構成する最初のネットワークのネットワークアドレスとサブネットマスク
- ネットワークタイプ (ローカルエリアネットワーク (LAN) かポイントツーポイントネットワーク)
- ルーターの検索、または特定のルーターの IP アドレス
- NIS サーバーの NIS ドメイン名および IP アドレス
- NIS+ サーバーの NIS+ ドメイン名および IP アドレス

DHCP サーバーは dhcpconfig コマンドを使用しても構成できます。このユーティリティーは、既存のシステムファイルから情報を自動的に収集して、使える初期構成を提供します。そのため、dhcpconfig コマンドを実行する前に既存のシステムファイルが正しいことを確認しておく必要があります。dhcpconfig がどのファイルから情報を入手するかについては、dhcpconfig(1M) のマニュアルページを参照してください。

IP アドレスの割り当て

DHCP サーバーは、下記のタイプの IP アドレス割り当てをサポートしています。

- 手動割り当て – DHCP サーバーは、特定の DHCP クライアント用に選択された専用の IP アドレスを割り当てます。このアドレスは変更したりほかのクライアントに割り当てたりすることはできません。
- 自動または常時割り当て – サーバーは有効期限のない IP アドレスを割り当て、その割り当てが変更されるか、あるいは、クライアントがそのアドレスを解放するまで、そのアドレスを永続的にそのクライアントに使用します。
- 動的割り当て – サーバーは IP アドレスを要求しているクライアントに、一定期間このアドレスをリースします (貸し出します)。この期間が過ぎると、サーバーはこのアドレスを回収し、ほかのクライアントに割り当てることができます。このアドレスの使用期間はサーバーに構成されているリース期間によって決まります。

ネットワーク構成情報

どのような情報を DHCP クライアントに提供するかを決める必要があります。DHCP サーバーを構成するときにはネットワークの基本的な情報を指定しますが、あとでクライアントに提供したい情報を追加することもできます。

DHCP サーバーは、オプションと値の対、およびマクロの形で dhcptab テーブルにネットワーク構成情報を保存します。オプションはクライアントに供給するネットワークデータのキーワードです。値はオプションに割り当てられ、DHCP メッセージでクライアントに渡されます。たとえば、NIS サーバーアドレスは、NISservs というオプションで渡されます。NISservs オプションの値は、DHCP サーバーによって割り当てられる一連の IP アドレスと同じものです。マクロは、クライアントに供給したい任意の個数のオプションをグループ化するための便利な方法です。DHCP マネージャを使えば、オプションをグループ化するマクロを作成し、それらのオプションに値を割り当てることができます。コマンド行ツールを使用する場合は、dhtadm (DHCP 構成テーブル管理ユーティリティー) を使ってオプションやマクロを扱うことができます。

DHCP オプションについて

DHCP では、オプションとはクライアントに渡される 1 つのネットワーク情報です。DHCP の資料では、オプションは「シンボル」や「タグ」と呼ばれる場合もあります。オプションは、数値コードやテキストラベルで定義されます。オプションは、DHCP サービスで使用されるときに値を受け取ります。

DHCP プロトコルは、一般的に指定されているネットワークデータに対して多数の標準オプションを定義しています。 それらオプションには、たとえば Subnet、Router、Broadcst、NIS+dom、Hostname、および LeaseTim があります。すべての標準オプションのリストについては、dhcp_inittab(4) のマニュアルページを参照してください。標準オプションのキーワードを変更することは一切できません。ただし、ネットワークに関連するオプションをマクロに組み込む際に、オプションに値を割り当てることができます。

標準オプションで指定できないデータに対しては、新しいオプションを作ることができます。作成するオプションは下記のいずれかのカテゴリに分類されるものでなければなりません。

- 拡張 – この DHCP サーバーにはまだ実装されていないが、標準 DHCP オプションとしてすでに予約されています。使用したい標準オプションがわかっているが、DHCP サーバーをグレードアップしない場合に使用できます。
- サイト – 使用しているサイトに固有なオプションのために予約されています。このようなオプションを作成します。
- ベンダー – ハードウェアまたはベンダープラットフォームなどの特定クラスのクライアントにだけ適用するオプションのために予約されています。DHCP の実装には、Oracle Solaris クライアント用の多数のベンダーオプションが含まれています。たとえばオプション SrootIP4 は、ネットワークからブートされるクライアントがそのルート (/) ファイルシステムとして使用すべきサーバーの IP アドレスを指定します。

DHCP オプションの作成、変更、削除に必要な手順については、第 15 章「DHCP の管理 (タスク)」を参照してください。

DHCP マクロについて

DHCP サービスにおけるマクロとは、ネットワーク構成オプションとユーザーがそのオプションに割り当てた値の集合のことです。マクロは、オプションをグループ化し、特定のクライアントまたはクライアントタイプにオプションをまとめて渡すために作成します。たとえば、特定のサブネット上のすべてのクライアントを対象としたマクロには、サブネットマスク、ルーター IP アドレス、ブロードキャストアドレス、NIS+ ドメイン、およびリース期間のためのオプションと値のペアを含めることができます。

DHCP サーバーによるマクロ処理

DHCP サーバーがマクロを処理するときは、そのマクロに定義されているネットワークオプションと値を、クライアントへの DHCP メッセージに含めます。サーバーは、特定のタイプのクライアントに対し一部のマクロを自動的に処理します。

サーバーでマクロを自動的に処理するためには、マクロの名前が、次の表に示すカテゴリのいずれかに従っている必要があります。

表 12–3　自動処理のための DHCP マクロカテゴリ

マクロのカテゴリ	説明
クライアントクラス	マクロ名は、クライアントマシンタイプ、オペレーティングシステム、またはその両方で表されるクライアントクラスと同じです。たとえば、あるサーバーに `SUNW.Sun-Blade-100` という名前のマクロがあるとします。ハードウェア実装が `SUNW,Sun-Blade-100` であるクライアントは、`SUNW.Sun-Blade-100` マクロの値を自動的に受け取ります。
ネットワークアドレス	マクロ名は、DHCP で管理されている IP アドレスと同じです。たとえば、サーバーのマクロの名前が `10.53.224.0` の場合、`10.53.224.0` ネットワークに接続されているクライアントはいずれも自動的に `10.53.224.0` マクロ内の値を受け取ります。
クライアント ID	マクロ名は、クライアントのある種の固有識別子と同じです。通常、Ethernet または MAC アドレスから得られます。たとえば、`08002011DF32` という名前のマクロがサーバーに存在する場合、(Ethernet アドレス `8:0:20:11:DF:32` から得られる) クライアント ID `08002011DF:32` を持つクライアントは、`08002011DF32` という名前のマクロにある値を自動的に受け取ります。

表 12–3 に記載されたいずれのカテゴリも使用していない名前をもつマクロは、次の条件の 1 つが当てはまるときにだけ処理できます。

- マクロが IP アドレスに割り当てられる場合。
- マクロが、自動的に処理されるほかのマクロに含まれる場合。
- マクロが、IP アドレスに割り当てられているほかのマクロに含まれている場合。

注 – サーバーを構成する場合、デフォルトでは、そのサーバーの名前と一致する名前の付いたマクロが作られます。このサーバーマクロは、自動処理が行われる名称タイプのいずれとも一致しないため、いずれのクライアントに対しても自動的に処理されません。あとでサーバー上で IP アドレスを作成する場合、その IP アドレスは、サーバーのデフォルトのマクロを使用するように割り当てられます。

マクロ処理の順序

DHCP クライアントが DHCP サービスを要求するときは、DHCP サーバーはどのマクロがそのクライアントに一致するかを決定します。サーバーは、マクロを処理する際に、マクロカテゴリを使って処理の順序を決めます。最も一般的なカテゴリが最初に処理され、最も特定されるカテゴリが最後に処理されます。マクロは下記の順序で処理されます。

1. クライアントクラスマクロ – 最も一般的なカテゴリ
2. ネットワークアドレスマクロ – クライアントクラスよりは特定なマクロ
3. IP アドレスに割り当てられたマクロ – ネットワークアドレスよりは特定されたマクロ
4. クライアント ID マクロ – 1 クライアントだけに適用される最も特定されたカテゴリ

ほかのマクロに含まれているマクロはそのマクロの一部として処理されます。

複数のマクロに同じオプションが含まれている場合は、最も特定されたカテゴリのマクロ内のオプションに設定されている値が一番最後に処理されるため、その値が使用されます。たとえば、ネットワークアドレスに、24 時間の値を持つリース期間オプションが含まれていて、クライアント ID マクロに 8 時間の値を持つリース期間オプションが含まれている場合は、そのクライアントは 8 時間のリース期間を受け取ります。

DHCP マクロのサイズ限度

マクロのすべてのオプションに割り当てられている値の長さの合計は、オプションコードと長さ情報を含め 255 バイト以内でなければなりません。この制限は、DHCP プロトコルによるものです。

この制限による影響が最も大きいと思われるマクロは、Oracle Solaris インストールサーバー上のファイルへのパスを渡すためのマクロです。一般に、渡すベンダー情報は、必要最小限に留めるべきです。さらに、パス名を必要とするオプションでは、短いパス名を使用すべきです。長いパスへのシンボリックリンクを作成する場合は、短いこのリンク名を渡すことができます。

DHCP クライアント

「クライアント」という用語は、ネットワーク上でクライアントとしての役割を実行している物理的なマシンについて言及するために使用される場合があります。ただし、このドキュメントで説明している DHCP クライアントはソフトウェアエンティティーです。DHCP クライアントは、そのネットワーク構成を DHCP サーバーから受け取るように構成したシステムの Oracle Solaris で動作するデーモン (`dhcpagent`) です。ほかのベンダーの DHCP クライアントも DHCP サーバーのサービスを使用できます。ただし、このドキュメントでは DHCP クライアントについてのみ説明します。

DHCP クライアントについての詳細は、第 16 章「DHCP クライアントの構成と管理」を参照してください。

◆ ◆ ◆ 第 13 章

DHCP サービスの使用計画 (手順)

DHCP サービスは、既存のネットワークで使用することも、これから構築するネットワークで使用することもできます。ネットワークを設定している場合には、DHCP サービスの設定を試みる前に第 2 章「TCP/IP ネットワークの計画 (手順)」を参照してください。既存のネットワークを使用する場合は、そのままこの章をお読みください。

この章では、ネットワークに DHCP サービスを設定する前に行うべき作業について説明します。この章の説明は、DHCP マネージャを使用することを前提にしていますが、DHCP サービスの設定はコマンド行ユーティリティー `dhcpconfig` を使って行うこともできます。

この章では、次の内容について説明します。

DHCP サービスを使用するためのネットワークの準備 (作業マップ)

DHCP の使用に先立ってネットワークを設定する際には、1 つまたは複数のサーバーを構成する際の判断に役立つ情報を収集する必要があります。次の表の作業マップを使用して、DHCP を使用するためにネットワークを準備する作業を特定します。この表では、必要なタスク、各タスクで実行する内容の説明、および、個別のタスクの実行手順を詳しく説明したセクションを一覧にして示しています。

タスク	説明	手順
ネットワークトポロジをマッピングします。	ネットワークでどのようなサービスが利用できるのかを判別し、その場所を突き止めます。	326 ページの「ネットワークトポロジのマッピング」
必要な DHCP サーバーの数を決めます。	予想される DHCP クライアント数を元にして必要な DHCP サーバーの数を決めます。	327 ページの「DHCP サーバー数の決定」
システムファイルと `netmasks` テーブルを更新します。	ネットワークトポロジを正確に反映します。	328 ページの「システムファイルとネットマスクテーブルの更新」

ネットワークトポロジのマッピング

ネットワークの物理的な構造を示すマップをまだ作成していない場合は、それを作成します。このマップには、ルーターやクライアントの場所と、ネットワークサービスを提供するサーバーの場所を明示してください。ネットワークトポロジを示すこのマップは、どのサーバーから DHCP サービスを提供するかや、どのような構成情報をクライアントに提供するかを決める上で役立ちます。

ネットワーク計画の詳細は、第 2 章「TCP/IP ネットワークの計画 (手順)」を参照してください。

DHCP 構成プロセスは、サーバーのシステムファイルとネットワークファイルから、いくつかのネットワーク情報を検索できます。これらのファイルについては、328 ページの「システムファイルとネットマスクテーブルの更新」を参照してください。ただし、クライアントにほかのサービス情報を提供したい場合には、サーバーのマクロに入力する必要があります。ネットワークトポロジを点検する際には、クライアントが認識する必要があるサーバーの IP アドレスを控えておいてください。たとえば、次のサーバーがネットワークでサービスを提供している場合があります。しかし、DHCP 構成では、これらのサーバーは検出されません。

- タイムサーバー
- ログサーバー
- プリントサーバー
- インストールサーバー
- ブートサーバー
- Web プロキシサーバー
- スワップサーバー
- X Window フォントサーバー
- Trivial File Transfer Protocol (TFTP) サーバー

避けなければならないネットワークトポロジ

IP ネットワーク環境によっては、いくつかのローカルエリアネットワーク (LAN) が同じネットワークハードウェアメディアを共有していることがあります。このようなネットワークでは、複数のネットワークハードウェアインタフェースが使用されていたり、複数の論理インタフェースが使用されていたりします。このような共有メディアネットワークでは、DHCP はうまく動作しません。同じ物理ネットワークで複数の LAN が動作していると、DHCP クライアントの要求はすべてのネットワークハードウェアインタフェースに送信されます。そのため、クライアントは、すべての IP ネットワークに同時に接続されているものとみなされます。

DHCP は、適切な IP アドレスをクライアントに割り当てられるように、クライアントのネットワークアドレスを特定できる必要があります。同じハードウェアメディアに複数のネットワークが存在していると、サーバーは、クライアントのネットワークを特定できないため、IP アドレスを割り当てることができません。

DHCP は、複数のネットワークの 1 つでしか実行できません。1 つのネットワークがユーザーの DHCP ニーズと合わない場合は、すべてのネットワークを再構成する必要があります。次のヒントを参考にしてください。

- サブネットに対して可変長サブネットマスク (VLSM) を使用して、現在の IP アドレス空間をより有効に活用します。これにより、同じ物理ネットワーク上で複数のネットワークを使用する必要がなくなるかもしれません。可変長サブネットの実装については、`netmasks(4)` のマニュアルページを参照してください。Classless Inter-Domain Routing (CIDR) と VLSM の詳細は、`http://www.ietf.org/rfc/rfc1519.txt` を参照してください。
- スイッチ上のポートを構成し、デバイスを別の物理 LAN に割り当てます。これにより、DHCP の要件である、1 つの LAN から 1 つの IP ネットワークへのマッピングが維持されます。ポートの構成については、スイッチに関する技術資料を参照してください。

DHCP サーバー数の決定

DHCP クライアントをサポートするために必要なサーバーの数は、データストアに何を使用するかによって異なります。次の表は、1 つの DHCP サーバーで DHCP と BOOTP クライアントをいくつまでサポートできるかをデータストア別に示したものです。

表 13–1 DHCP サーバーによってサポートされると見込まれる最大のクライアント数

データストア	サポートされる最大のクライアント数
テキストファイル	10,000
NIS+	40,000

表 13–1 DHCP サーバーによってサポートされると見込まれる最大のクライアント数 (続き)

データストア	サポートされる最大のクライアント数
バイナリファイル	100,000

この最大数は一般的な指針であり、絶対的な数ではありません。DHCP サーバーのクライアント容量は、サーバーが 1 秒間にいくつのトランザクションを処理する必要があるかに大きく依存します。一方、トランザクション頻度は、リース期間や使用パターンで大きく変わります。たとえば、リースが 12 時間に設定され、ほとんどのユーザーが夜にシステムを停止するとします。多くのユーザーが朝の同じ時間にシステムを開始すると、多数のクライアントがリースを同時に要求するので、サーバーは、トランザクションのピークを処理できなければなりません。したがって、このような環境では、DHCP サーバーがサポートできるクライアントの数は本来より少なくなります。リース期間がより長い環境や、ケーブルモデムなど常時接続されているデバイスからなる環境では、サポートされるクライアントの数は多くなります。

データストアのタイプによる比較については、331 ページの「DHCP データストアの選択」を参照してください。

システムファイルとネットマスクテーブルの更新

DHCP を構成する間に、DHCP ツールは、サーバー上のさまざまなシステムファイルを走査し、サーバーの構成に使用できる情報を収集します。

DHCP マネージャや `dhcpconfig` を使ってサーバーの構成を行う前に、システムファイルの内容が最新の状態になっていることを確認してください。サーバーの構成を行なったあとにエラーに気が付いた場合は、DHCP マネージャまたは `dhtadm` を使って、サーバー上のマクロを修正する必要があります。

次の表は、DHCP サーバーの構成中に収集されるいくつかの情報と、情報の提供元を示します。サーバーで DHCP を構成する前に、これらの情報が適切に設定されていることを確認してください。サーバーの構成後にシステムファイルを変更する場合は、この変更を反映するためにサービスを再構成する必要があります。

表 13–2　DHCP 構成に使用される情報

情報	送信元	コメント
タイムゾーン	システムの日時、時間帯の設定値	日時と時間帯は Oracle Solaris のインストール時に初期設定されます。日付を変更する場合は、`date` コマンドを使用します。`svc:/system/environment:init` SMF サービスの `timezone/localtime` プロパティーを変更して、`TZ` 環境変数を設定することによってタイムゾーンを変更できます。詳細は、`TIMEZONE(4)` のマニュアルページを参照してください。
DNS パラメータ	`/etc/resolv.conf`	DHCP サーバーは、`/etc/resolv.conf` ファイルから DNS ドメイン名や DNS サーバーアドレスなどの DNS パラメータを取得します。`resolv.conf` の詳細は、『Solaris のシステム管理 (ネーミングとディレクトリサービス : DNS、NIS、LDAP 編)』や `resolv.conf(4)` のマニュアルページを参照してください。
NIS または NIS+ パラメータ	システムのドメイン名、`nsswitch.conf`、NIS、NIS+	DHCP サーバーは、`domainname` コマンドを使ってサーバーシステムのドメイン名を取得します。`nsswitch.conf` ファイルを見てドメインベースの情報をどこから検索するかを決めます。サーバーシステムが NIS または NIS+ クライアントの場合、DHCP サーバーは照会を行なって NIS または NIS+ サーバーの IP アドレスを取得します。詳細は、`nsswitch.conf(4)` のマニュアルページを参照してください。
デフォルトルーター	システムの経路制御テーブル、管理者による入力	DHCP サーバーはネットワーク経路制御テーブルを検索し、ローカルネットワークに接続されているクライアントのデフォルトルーターを見つけます。クライアントが同じネットワーク上にない場合は、DHCP サーバーがこの情報の入力を要求する必要があります。

表 13–2 DHCP 構成に使用される情報 (続き)

情報	送信元	コメント
サブネットマスク	ネットワークインタフェース、netmasks テーブル	DHCP サーバーは、自身のネットワークインタフェースを参照して、ローカルクライアント用のネットマスクとブロードキャストアドレスを特定します。この要求がリレーエージェントからすでに転送されてきている場合には、リレーエージェントのネットワークにある netmasks テーブルからサブネットマスクを取得します。
ブロードキャストアドレス	ネットワークインタフェース、netmasks テーブル	ローカルネットワークの場合には、DHCP サーバーは、ネットワークインタフェースからブロードキャストアドレスを取得します。リモートネットワークでは、サーバーは BOOTP リレーエージェントの IP アドレスとリモートネットワークのネットマスクを使用して、そのネットワーク用のブロードキャストアドレスを計算します。

DHCP サーバーの構成前に必要な選択 (タスクマップ)

この節では、ネットワークに最初の DHCP サーバーを構成する前に決定する必要がある事柄について説明します。次の表では、ネットワークを構成して DHCP を使用するために必要な選択の概要と、各作業の実行手順を説明した節へのリンクを示しています。

タスク	説明	手順
DHCP サーバーを選択します。	DHCP サービスを実行するためのシステム要件をサーバーが満たしているかどうか判断します。	331 ページの「DHCP サービスを実行するホストの選択」
データストアを選択します。	データストアタイプを比較して、サイトに最も適したデータストアを決定します。	331 ページの「DHCP データストアの選択」
リースポリシーを設定します。	サイトに適したリースポリシーを決定するために、IP アドレスのリースについて確認します。	332 ページの「リースポリシーの設定」

タスク	説明	手順
ルーターのアドレスを指定するか、ルーターを検索するかを選択します。	DHCP クライアントでルーターを検索するか、特定のルーターを使用するかを決定します。	334 ページの「DHCP クライアントのためのルーターの決定」

DHCP サービスを実行するホストの選択

ネットワークトポロジを念頭に置き、次のシステム要件に従って、DHCP サーバーを設定するホストを選択します。

ホストの必要条件は次のとおりです。

- Solaris 2.6 リリース以降が動作している。多数のクライアントをサポートする場合は、Solaris 8 7/01 リリース以降のバージョンをインストールする必要があります。
- DHCP を使用する予定のクライアントがあるすべてのネットワークに、直接ネットワーク経由、または BOOTP リレーエージェントを介してホストからアクセスできる。
- 経路制御を使用するように構成されている。
- ホストでは、ネットワークトポロジを反映した `netmasks` テーブルが正しく構成されている。

DHCP データストアの選択

DHCP データは、テキストファイル、バイナリファイル、または NIS+ ディレクトリサービスに保存できます。次の表に、各データストアの特長と、各データストアを使用すべき環境を示します。

表 13–3 DHCP データストアの比較

データストア	パフォーマンス	保守	共有	環境
バイナリファイル	高パフォーマンス、大容量	少ない保守で、データベースサーバーが不要です。内容は、DHCP マネージャ、`dhtadm`、`pntadm` で表示する必要があります。ファイルの定期的なバックアップが必要です。	データストアを DHCP サーバーの間で共有することはできません。	多数のネットワークからなり、ネットワークごとに数千のクライアントがいる中規模から大規模の環境。小規模から中規模の ISP に適しています。

表 13–3 DHCP データストアの比較 (続き)

データストア	パフォーマンス	保守	共有	環境
NIS+	中程度のパフォーマンスと容量。NIS+ サービスのパフォーマンスと容量に依存する	DHCP サーバーシステムが NIS+ クライアントとして構成されていなければなりません。NIS+ サービスの保守が必要です。内容は、DHCP マネージャ、`dhtadm`、`pntadm` で表示する必要があります。`nisbackup` による定期的なバックアップが必要です。	DHCP データは NIS+ に分散されます。複数のサーバーから同じコンテナにアクセスできます。	ネットワーク当たり 5000 クライアントまでの小規模から中規模の環境。
テキストファイル	中程度のパフォーマンス、少ない容量	少ない保守で、データベースサーバーが不要です。ASCII ファイルであるため、DHCP マネージャ、`dhtadm` または `pntadm` を使用しなくても見ることができます。ファイルの定期的なバックアップが必要です。	データストアを DHCP サーバーの間で共有できます。ただし、DHCP データが、NFS マウントポイントを通してエクスポートされる 1 つのファイルシステムに格納されていなければなりません。	ネットワーク当たり数百から 1,000 クライアントで、合計が 10,000 クライアント未満の小規模な環境。

従来の NIS はデータストアオプションとしては推奨されません。NIS が高速な増分更新をサポートしていないためです。ネットワークで NIS が使用されている場合は、データストアとしてテキストファイルまたはバイナリファイルを使用することをお勧めします。

リースポリシーの設定

「リース」とは、DHCP サーバーが特定の IP アドレスの使用を DHCP クライアントに許可する期間のことです。管理者は、サーバーの初期構成時に、サイト全体に適用するリースポリシーを指定する必要があります。「リースポリシー」には、リース期間や、クライアントがこのリースを更新できるかどうかを指定します。サーバーは提供された情報を使用して、構成時に作成するデフォルトマクロ内のオプションの値を設定します。管理者は、作成する構成マクロでオプションを使用することによって、特定のクライアントや特定のクライアントタイプごとに、異なるリースポリシーを設定することもできます。

「リース期間」は、リースが有効な時間数、日数、または週数として指定されます。クライアントに IP アドレスが割り当てられると (あるいは、クライアントが IP アドレスのリースを再度ネゴシエーションすると)、リースの満了日時が計算されます。その際には、クライアントの DHCP 肯定応答のタイムスタンプにリース期間の時間数が加算されます。たとえば、DHCP 肯定応答のタイムスタンプが 2005 年 9 月 16 日 9:15 AM で、リース期間が 24 時間だとすると、リース満了時間は 2005 年 9 月 17 日 9:15 AM になります。リース満了日時はクライアントの DHCP ネットワークレコード中に保存され、DHCP マネージャや `pntadm` ユーティリティーで表示されます。

リース期間には、期限切れのアドレスを速やかに再利用できるように比較的小さな値を設定します。ただし、リース期間の値は、DHCP サービスの中断時間よりも長い期間にする必要があります。クライアントは、DHCP サービスが動作するシステムが修理されている間も動作できる必要があります。一般的な指針としては、サーバーの予想停止時間の 2 倍を指定します。たとえば、故障部品を検出、交換し、システムをリブートするのに 4 時間かかるとすれば、8 時間をリース期間に指定します。

リースネゴシエーションオプションは、リースが満了する前に、クライアントが提供されたリースについてサーバーとネゴシエーションできるかどうかを決めるものです。リースのネゴシエーションが可能な場合、クライアントはリースの残り時間を常に監視します。そして、リース期間の半分が経過すると、リースを元のリース期間の値に戻す要求を DHCP サーバーに送ります。システムの数が IP アドレスの数より多い環境では、リースのネゴシエーションを無効にする必要があります。それによって、IP アドレスの使用時間に限度が与えられます。しかし、IP アドレスの数が十分にある場合は、リースネゴシエーションを有効にして、リース期間の満了時に、ネットワークインタフェースの停止をクライアントに強制することを避ける必要があります。さらに、クライアントに新しいリースを取得させると、NFS や telnet セッションなど、クライアントの TCP 接続が中断されるおそれがあります。リースネゴシエーションは、サーバーの構成時にサイト全体に対して有効にできます。あるいは、構成マクロの `LeaseNeg` オプションを使用すれば、特定のクライアントやクライアントタイプだけにリースネゴシエーションを有効にできます。

注 – ネットワークでサービスを提供するシステムはその IP アドレスを保持する必要があります。このようなシステムは、短期的なリースに依存すべきではありません。このようなシステムで DHCP を使用する場合は、常時リースにより IP アドレスを割り当てるのではなく、予約済みの IP アドレスをシステムに手動で割り当てるべきです。これによって、このシステムの IP アドレスが使用されなくなったときには、それを検出できます。

DHCP クライアントのためのルーターの決定

ホストシステムは、ローカルネットワークの外側にあるネットワークと通信する場合、ルーターを使用します。ホストは、これらのルーターの IP アドレスを知っていなければなりません。

DHCP サーバーを構成する際には、次のどちらかの方法で DHCP クライアントにルーターアドレスを提供する必要があります。 1 つの方法は、ルーターの IP アドレスを指定する方法です。これより望ましい方法としては、クライアントがルーター発見プロトコルを使ってルーターを検索する方法があります。

そのネットワークのクライアントがルーター発見機能を実行できる場合には、ルーターが 1 つしかなくてもルーター発見プロトコルを使用すべきです。ルーター発見プロトコルを使用すると、クライアントはネットワーク内でのルーター変更に容易に対応できます。たとえば、ルーターに故障が発生したために、新しいアドレスを持つルーターに置き換えられる場合でも、クライアントは、新しいアドレスを自動的に検出できます。新しいネットワーク構成を受信して新しいルーターアドレスを取得する必要はありません。

IP アドレスの管理に必要な選択 (タスクマップ)

DHCP サービスの設定の一環として、サーバーが管理する IP アドレスに関する要素を決定します。ネットワークに複数の DHCP サーバーが必要な場合には、いくつかの IP アドレスに対する役割をサーバーごとに割り当てることができます。その際には、アドレス管理をどのように分担させるかを決定する必要があります。次の表の作業マップでは、ネットワークで DHCP を使用するときに IP アドレスを管理するための作業について説明しています。表には、各タスクの実行方法を詳しく説明したセクションへのリンクも含まれています。

タスク	説明	参照先
サーバーが管理するアドレスを指定します。	DHCP サーバーが管理するアドレスの数と範囲を決定します。	335 ページの「IP アドレスの数と範囲」
サーバーがクライアントのホスト名を自動的に生成するかどうかを決定します。	ホスト名を生成すべきかどうかを決定できるように、クライアントホスト名が生成される方法を確認します。	335 ページの「クライアントホスト名の生成」
クライアントに割り当てる構成マクロを決定します。	クライアントに適したマクロを選択できるように、クライアント構成マクロについて確認します。	335 ページの「デフォルトのクライアント構成マクロ」
使用するリースタイプを決定します。	DHCP クライアントに最適なリースタイプを決定するために、リースタイプについて確認します。	336 ページの「動的リースタイプと常時リースタイプ」

IP アドレスの数と範囲

DHCP マネージャを使用すると、サーバーの初期構成時に、総アドレス数とブロックの開始アドレスを指定することにより、そのブロック分の IP アドレス、またはその範囲内の IP アドレスを DHCP の管理下に追加できます。DHCP マネージャは、この情報から連続するアドレスのリストを作成し、追加します。アドレスが連続していない複数のブロックがある場合は、初期構成のあとに DHCP マネージャのアドレスウィザードを再起動してほかのアドレスを追加できます。

IP アドレスの構成を行う前に、アドレスを追加する最初のブロックにあるアドレスの数と、その範囲内の開始のアドレスの IP アドレスを控えておいてください。

クライアントホスト名の生成

DHCP 本来の動的な特性により、IP アドレスはそれを使用するシステムのホスト名に恒久的に関連付けられる訳ではありません。DHCP 管理ツールでは、各 IP アドレスに対応するクライアント名を生成できます。クライアント名には、接頭辞 (ルート名) とダッシュ、それにサーバーから割り当てられる数字が使用されます。たとえば、ルート名が `charlie` なら、クライアント名は `charlie-1`、`charlie-2`、`charlie-3` のようになります。

デフォルトでは、生成されたクライアント名は、それを管理する DHCP サーバーの名前で始まります。この方法は、複数の DHCP サーバーが存在する環境で便利です。特定の DHCP サーバーがどのクライアントを管理しているのかを DHCP ネットワークテーブルから簡単に知ることができるからです。ただし、ルート名は任意の名前に変更できます。

IP アドレスを構成する前に、DHCP 管理ツールを使ってクライアント名を生成するかどうか、生成する場合は、そのクライアント名に使用するルート名を決めてください。

生成されるクライアント名は、DHCP の構成時にホスト名を登録するように指定すれば、`/etc/inet/hosts`、DNS、または NIS+ 内の IP アドレスに対応付けることができます。詳細は、374 ページの「クライアントホスト名の登録」を参照してください。

デフォルトのクライアント構成マクロ

DHCP で、マクロは複数のネットワーク構成オプションとその設定値の集まりです。DHCP サーバーは、マクロを使って、どのようなネットワーク構成情報を DHCP クライアントに送信するかを決めます。

管理ツールは、DHCP サーバーの構成時に、システムファイルから情報を収集するだけでなく、プロンプトやコマンド行オプションを通して管理者から直接情報を収集します。この情報から次のマクロを作成します。

- ネットワークアドレスマクロ - ネットワークアドレスマクロの名前は、クライアントネットワークの IP アドレスと同じになります。たとえば、ネットワークのアドレスが `192.68.0.0` なら、ネットワークアドレスマクロの名前も `192.68.0.0` になります。このマクロには、ネットワークのどのクライアントでも必要になる情報が含まれています。たとえば、サブネットマスク、ネットワークブロードキャストアドレス、デフォルトルーター、またはルーター発見トークン、さらに、サーバーで NIS/NIS+ を使用する場合には、NIS/NIS+ のドメインとサーバーなどです。ネットワークに適用可能なその他のオプションも含まれることがあります。ネットワークアドレスマクロは、そのネットワークにあるすべてのクライアントに対して自動的に処理されます (323 ページの「マクロ処理の順序」を参照)。
- ロケールマクロ — ロケールマクロの名前は `Locale` です。このマクロには、時間帯を指定するための協定世界時 (UTC) からの時間差 (秒単位) が含まれています。ロケールマクロは自動的には処理されません。これは、サーバーマクロに含まれています。
- サーバーマクロ — サーバーマクロの名前はサーバーのホスト名と同じになります。たとえば、サーバーの名前が `pineola` なら、サーバーマクロの名前も `pineola` になります。このサーバーマクロには、リースポリシー、時間サーバー、DNS ドメイン、DNS サーバーに関する情報のほかに、構成プログラムがシステムファイルから入手したその他の情報が含まれていることがあります。サーバーマクロにはロケールマクロが含まれているため、DHCP は、ロケールマクロをサーバーマクロの一部として処理します。

  最初のネットワークの IP アドレスを構成する際に、これらのアドレスを使用するすべての DHCP クライアントに対して使用するクライアント構成マクロを選択する必要があります。選択したマクロは、その IP アドレスにマップされます。デフォルトではサーバーマクロが選択されます。このサーバーマクロには、このサーバーを使用するすべてのクライアントに必要な情報が含まれています。

クライアントは、IP アドレスにマップされているマクロのオプションより先に、ネットワークアドレスマクロに含まれているオプションを受け取ります。そのため、サーバーマクロのオプションとネットワークアドレスマクロのオプションが矛盾する場合は、サーバーマクロのオプションが優先します。マクロが処理される順序については、323 ページの「マクロ処理の順序」を参照してください。

動的リースタイプと常時リースタイプ

構成しようとするアドレスにリースポリシーが適用されるかどうかは、「リースタイプ」で決まります。DHCP マネージャでは、最初のサーバーの構成時に、追加す

るアドレスに動的リースを使用するか、常時リースを使用するかを選択できます。dhcpconfig コマンドによる DHCP サーバーの構成では、動的リースが使用されます。

IP アドレスのリースが「動的リース」なら、DHCP サーバーはこのアドレスを管理できます。DHCP サーバーは、IP アドレスをクライアントに割り当て、リース期間を延長し、さらに、そのアドレスが使用されなくなったら、それを検出し、再利用できます。IP アドレスのリースが「常時リース」の場合、DHCP サーバーはアドレスを割り当てることしかできません。その場合、クライアントは、そのアドレスを、明示的に解放するまで所有します。アドレスが解放されると、サーバーはアドレスをほかのクライアントに割り当てることができます。そのアドレスは、常時リースとして構成されている限り、リースポリシーの対象となることはありません。

IP アドレスの範囲を構成した場合、選択したリースタイプはその範囲内のすべてのアドレスに適用されます。DHCP の利点を最大限に活かすためには、大部分のアドレスに対して動的リースを使用する必要があります。必要なら、あとで個々のアドレスを変更して常時リースを指定できます。ただし、常時リースの数は最小限に抑えるべきです。

予約済み IP アドレスとリースタイプ

IP アドレスは、特定のクライアントに手動で割り当てることにより予約できます。予約済みアドレスは、常時リースとも動的リースとも関連付けることもできます。予約済みアドレスに常時リースが割り当てられている場合には、次の条件が適用されます。

- そのアドレスに結合されているクライアント以外のクライアントにそのアドレスを割り当てることはできません。
- DHCP サーバーがこのアドレスを別のクライアントに割り当てることはできません。
- DHCP サーバーがこのアドレスを再利用することはできません。

予約済みアドレスに動的リースが割り当てられている場合には、このアドレスに結合されているクライアントだけにこのアドレスを割り当てることができます。ただし、クライアントは、アドレスが予約されていないかのように、リース期間を管理し、リースの延長をネゴシエートする必要があります。この方法を使用すると、ネットワークテーブルを参照するだけで、クライアントがそのアドレスを使用しているかどうかを監視できます。

初期構成時にすべての IP アドレスに対して予約済みアドレスを生成することはできません。予約済みアドレスは、個々のアドレスに対しての使用は制限すべきものだからです。

複数の DHCP サーバーを使用するための計画

複数の DHCP サーバーを構成して IP アドレスを管理する場合には、次のガイドラインに従ってください。

- 各サーバーがそれぞれのアドレス範囲を受け持ち、またアドレス範囲が重複しないように、IP アドレスのプールを分割します。
- 可能であれば、データストアとして NIS+ を選択します。そうでない場合は、テキストファイルを選択し、データストアへの絶対パスとして共有ディレクトリを指定します。バイナリファイルのデータストアを共有することはできません。
- アドレスの所有権が正しく割り当てられるように、またサーバーベースのマクロが自動的に作成されるように、個々のサーバーを個別に構成します。
- 指定された時間間隔で `dhcptab` テーブルのオプションとマクロを走査するようにサーバーを設定します。これによって、すべてのサーバーが最新の情報を使用します。DHCP マネージャでは、`dhcptab` の自動的な読み取りをスケジュールできます (375 ページの「DHCP サーバー用のパフォーマンスオプションのカスタマイズ」を参照)。
- すべてのクライアントからすべての DHCP サーバーにアクセスできるようにします。これによって、個々のサーバーがそれぞれを相互にサポートできます。たとえば、有効な IP アドレスリースを持つクライアントが、そのアドレスを所有するサーバーにアクセスできないため、構成の検証またはリースの延長を行おうとしているとします。クライアントがプライマリサーバーに 20 秒間アクセスを試みても応答がない場合は、別のサーバーがクライアントに応答できます。さらに、あるクライアントが特定の IP アドレスを要求しても、そのアドレスを所有するサーバーが応答しない場合にも、ほかのいずれかのサーバーが要求を処理します。ただし、この場合、クライアントが受信するのは、要求したアドレスではありません。クライアントは、応答した DHCP サーバーが所有している IP アドレスを受け取ります。

リモートネットワークの DHCP 構成の計画

DHCP の初期構成が完了すると、リモートネットワーク内の IP アドレスを DHCP の管理下に置くことができます。ただし、システムファイルはサーバー内にないため、DHCP マネージャや `dhcpconfig` はデフォルト値を提供するための情報を検索できません。したがって、管理者が情報提供する必要があります。リモートネットワークの構成を行う前に、次の情報を用意してください。

- リモートネットワークの IP アドレス。
- リモートネットワークのサブネットマスク。この情報は、ネームサービスの `netmasks` テーブルから取得できます。ネットワークがローカルファイルを使用する場合は、そのネットワーク内のシステム上にある `/etc/netmasks` を参照してください。ネットワークが NIS+ を使用する場合は、`niscat netmasks.org_dir` コマンドを使用します。ネットワークが NIS を使用する場合には、`ypcat -k netmasks.byaddr` コマンドを使用します。`netmasks` テーブルが、管理対象としたいすべてのサブネットに関するトポロジ情報をすべて含んでいることを確認してください。
- ネットワークタイプ。クライアントは、ローカルエリアネットワーク (LAN) 接続か、ポイントツーポイントプロトコル (PPP) を通してネットワークに接続します。
- 経路制御情報。クライアントはルーター発見機能を使用できますか。使用できない場合は、クライアントが使用するルーターの IP アドレスを指定する必要があります。
- NIS ドメインと NIS サーバー (使用する場合)。
- NIS+ ドメインと NIS+ サーバー (使用する場合)。

DHCP ネットワークの追加手順については、381 ページの「DHCP ネットワークの追加」を参照してください。

DHCP を構成するためのツールの選択

情報の収集や DHCP サービスの計画が終わったら、DHCP サーバーを構成します。サーバーの構成には、DHCP マネージャかコマンド行ユーティリティー `dhcpconfig` を使用します。DHCP マネージャでオプションを選択し、データを指定すると、そのデータから DHCP サーバーが使用する `dhcptab` テーブルとネットワークテーブルが作成されます。`dhcpconfig` ユーティリティーでは、コマンド行オプションを使ってデータを入力する必要があります。

DHCP マネージャの機能

Java™ 技術ベースの GUI ツールである DHCP マネージャには、DHCP 構成ウィザードがあります。DHCP サーバーとして構成されていないシステムで DHCP マ

ネージャを初めて実行すると、DHCP 構成ウィザードが自動的に起動されます。DHCP 構成ウィザードの一連のダイアログボックスでは、サーバーの構成に不可欠な次の情報を入力する必要があります。 データストア形式、リースポリシー、DNS/NIS/NIS+ サーバーとドメイン、ルーターのアドレスなど。ただし、この情報の一部はウィザードがシステムファイルから入手します。したがって、情報が正しいかどうかを確認したり、正しくない場合は訂正します。

いくつかのダイアログボックスを表示し、情報が正しいことを確認すると、DHCP サーバーデーモンがサーバーシステムで起動されます。そして、追加アドレスウィザードを起動し、ネットワーク用の IP アドレスを構成するように求められます。最初は、サーバーのネットワークだけが DHCP 用に構成され、その他のサーバーオプションにはデフォルト値が与えられます。初期構成が完了したあとで DHCP マネージャを再起動すると、ネットワークを追加したり、ほかのサーバーオプションを変更したりできます。

DHCP 構成ウィザードについては、343 ページの「DHCP サーバーの構成と構成解除 (DHCP マネージャ)」を参照してください。DHCP マネージャの詳細については、356 ページの「DHCP マネージャーについて」を参照してください。

dhcpconfig 機能

`dhcpconfig` ユーティリティーは、DHCP サーバーの構成や構成解除だけでなく、新しいデータストアへの変換やほかの DHCP サーバーとのデータのインポート/エクスポートを行うことができます。`dhcpconfig` ユーティリティーを使って DHCP サーバーを構成しようとすると、このユーティリティーは、システムファイルから情報を取得します (328 ページの「システムファイルとネットマスクテーブルの更新」を参照)。DHCP マネージャの場合とは異なり、システムファイルから取得した情報の表示や確認を行うことはできません。したがって、`dhcpconfig` を使用する場合は、システムファイルを事前に更新しておくことが重要になります。コマンド行オプションを使用すると、`dhcpconfig` がデフォルトでシステムファイルから得る値を無効にできます。`dhcpconfig` ユーティリティーは、スクリプト中で使用できます。詳細は、`dhcpconfig(1M)` のマニュアルページを参照してください。

DHCP マネージャと dhcpconfig の比較

下の表に、2 つのサーバー構成ツールの相違点を示します。

表 13–4　DHCP マネージャと dhcpconfig コマンドの比較

機能	DHCP マネージャ	`dhcpconfig` (オプションの指定)
システムから収集されたネットワーク情報。	システムファイルから収集された情報を表示し、必要な場合は変更できます。	コマンド行オプションを使ってネットワーク情報を指定できます。

表 13–4　DHCP マネージャと dhcpconfig コマンドの比較　　(続き)

機能	DHCP マネージャ	dhcpconfig (オプションの指定)
構成処理の速さ。	必須ではないサーバーオプションのプロンプトを省略することによって、構成処理を高速化できます。代わりにデフォルト値を使用します。必須ではないオプションは、初期構成後に変更できます。	構成処理はもっとも速いですが、多くのオプションを使って値を指定しなければならない場合があります。

DHCP マネージャまたは dhcpconfig ユーティリティーを使ってサーバーを構成する手順については、第 14 章「DHCP サービスの構成 (手順)」を参照してください。

◆ ◆ ◆ 第 14 章

DHCP サービスの構成 (手順)

ネットワーク上の DHCP サービスを構成する際には、最初の DHCP サーバーを構成し起動します。ほかの DHCP サーバーはあとで追加できます。データストアが共有データをサポートする場合、共有された場所から同じデータにアクセスできます。この章では、DHCP サーバーを構成し、ネットワークと関連する IP アドレスを DHCP の管理下に置く作業について説明します。さらに、DHCP サーバーを構成解除する方法についても説明します。

タスクごとに、同じタスクを DHCP マネージャで処理する手順と、 `dhcpconfig` ユーティリティーで処理する手順を説明します。この章では、次の内容について説明します。

- 343 ページの「DHCP サーバーの構成と構成解除 (DHCP マネージャ)」
- 351 ページの「DHCP サーバーの構成と構成解除 (`dhcpconfig` コマンド)」

DHCP サービスの構成作業で問題がある場合は、第 17 章「DHCP のトラブルシューティング (リファレンス)」を参照してください。

DHCP サービスの構成が終了したら、DHCP サービスの管理に関する説明が記載されている第 15 章「DHCP の管理 (タスク)」を参照してください。

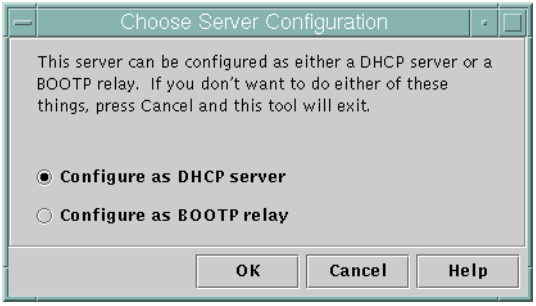
DHCP サーバーの構成と構成解除 (DHCP マネージャ)

この節では、DHCP マネージャを使用して DHCP サーバーを構成および構成解除する手順について説明します。なお、DHCP マネージャを使用するには、CDE や GNOME などの X Window System が動作している必要があります。

`/usr/sadm/admin/bin/dhcpmgr` コマンドで DHCP マネージャを実行するためには、スーパーユーザーになる必要があります。このユーティリティーの一般的な説明については、356 ページの「DHCP マネージャーについて」を参照してください。DHCP マネージャを実行する方法についての詳細は、362 ページの「DHCP サービスを起動および停止する方法 (DHCP マネージャ)」を参照してください。

DHCP として構成されていないサーバーで DHCP マネージャを実行すると、次の画面が表示されます。DHCP サーバーを構成するのか、BOOTP リレーエージェントを構成するのかを指定できます。

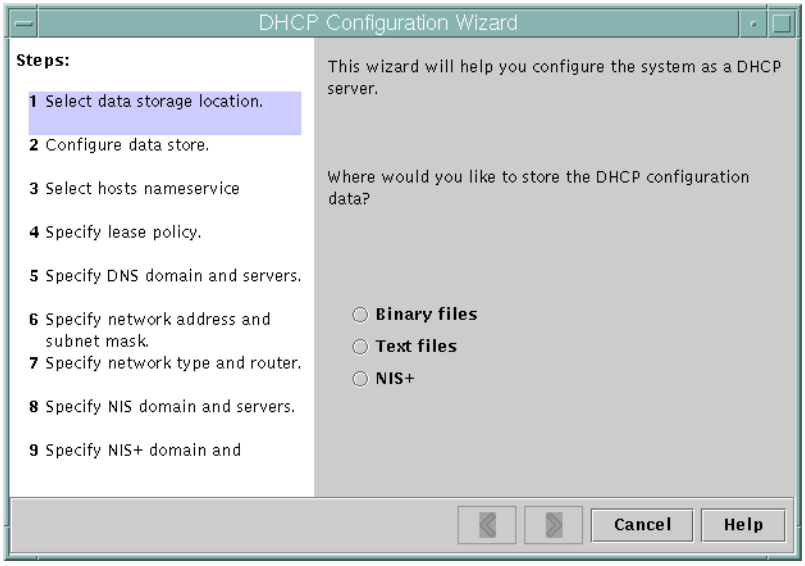
図 14–1 DHCP マネージャの「サーバー構成の選択 (Choose Server Configuration)」ダイアログボックス



DHCP サーバーの構成

DHCP サーバーを構成する場合は、DHCP マネージャによって DHCP 構成ウィザードが起動され、サーバーの構成に必要な情報の入力を求められます。次の図に示すような、ウィザードの初期画面が表示されます。

図 14–2　DHCP 構成ウィザードの初期画面



ウィザードの質問に答えると、DHCP マネージャは、次のテーブルに示されている項目を作成します。

表 14–1　DHCP サーバーの構成時に作成される項目

項目	説明	内容
サーバー構成ファイル。`/etc/inet/dhcpsvc.conf`	サーバー構成オプションのキーワードおよび値を記録します。	データストア形式とその場所、システムのブート時に DHCP デーモンを起動するために `in.dhcpd` に指定するオプション。このファイルを手動で編集してはいけません。DHCP 構成情報を変更するには、`dhcpmgr` または `dhcpconfig` を使用する必要があります。
`dhcptab` テーブル	DHCP マネージャは、`dhcptab` テーブルが存在しない場合はこれを生成します。	値が割り当てられたマクロとオプション。
ロケールマクロ (任意)。名前は `Locale`	ユニバーサル時間 (UTC) とローカルな時間帯との時間差 (秒単位) が含まれます。	秒数が割り当てられた `UTCoffst` オプション。

表 14–1 DHCP サーバーの構成時に作成される項目 (続き)

項目	説明	内容
サーバーマクロ。名前はサーバーのノード名に一致する	DHCP サーバーを構成した管理者の入力によって決定される値を持つオプションが含まれます。オプションは、サーバーが所有するアドレスを使用するすべてのクライアントに適用されます。	Locale マクロと次のオプション。 ■ Timeserv。サーバーのプライマリ IP アドレスを指し示すように設定されています。 ■ LeaseTim。リースの時間 (秒数) が設定されます。 ■ LeaseNeg (ネゴシエーション可能なリースを選択した場合)。 ■ DNSdmain および DNSserv (DNS が構成されている場合)。 ■ Hostname。このオプションに値を設定してはいけません。このオプションが存在すると、ホスト名はネームサービスから取得される必要があることを意味します。
ネットワークアドレスマクロ。名前はクライアントネットワークのネットワークアドレスと同じ	DHCP サーバーを構成した管理者の入力によって決定される値を持つオプションが含まれます。オプションは、マクロ名で指定されたネットワーク上に存在するすべてのクライアントに適用されます。	次のオプション。 ■ Subnet。ローカルサブネットのサブネットマスクが設定されます ■ Router。ルーターの IP アドレスが設定されます。RDiscvyF。クライアントはルーターディスカバリを使用します ■ Broadcst。ブロードキャスト IP アドレスが設定されます。このオプションは、ネットワークがポイントツーポイントネットワークではない場合だけ存在します。 ■ MTU。最大転送ユニット ■ NISdmain および NISservs (NIS が構成されている場合) ■ NIS+dom および NIS+serv (NIS+ が構成されている場合)
ネットワークのネットワークテーブル	ネットワークの IP アドレスが作成されるまで、空のテーブルとして作成されます。	IP アドレスを追加するまで、内容はありません。

▼ DHCP サーバーを構成する方法 (DHCP マネージャ)

始める前に　DHCP サーバーを構成する前に第 13 章「DHCP サービスの使用計画 (手順)」を必ずお読みください。特に、次のタスクを行う際には、330 ページの「DHCP サーバーの構成前に必要な選択 (タスクマップ)」 のガイドラインが役立ちます。

- DHCP サーバーとして使用するシステムを選択します。
- データストア、リースポリシー、ルーター情報について決定します。

1. サーバーシステム上でスーパーユーザーになります。
2. **DHCP** マネージャを起動します。

   ```
   #/usr/sadm/admin/bin/dhcpmgr &
   ```

3. 「**DHCP** サーバーとして構成 **(Configure as DHCP Server)**」オプションを選択します。

   サーバーの構成を支援する「DHCP 構成ウィザード (DHCP Configuration Wizard)」が表示されます。
4. 計画作成段階で決めた事項に基づいて、オプションを選択するか、要求された情報を入力します。

   わからないことがある場合は、ウィザードウィンドウ内の「ヘルプ (Help)」をクリックして Web ブラウザを開き、DHCP 構成ウィザードのヘルプを表示します。
5. 要求された情報の指定が終了したら、「完了 **(Finish)**」をクリックしてサーバー構成を完了します。
6. アドレス起動ウィザードのプロンプトで「はい **(Yes)**」をクリックし、サーバーの **IP** アドレスを構成します。

   「ネットワークへアドレスの追加 (Add Addresses to Network)」ウィザードでは、どのアドレスを DHCP の制御下に置くかを指定できます。
7. 計画作成段階で決めた事柄に従って質問に答えます。

   詳細は、334 ページの「IP アドレスの管理に必要な選択 (タスクマップ)」を参照してください。わからないことがある場合は、ウィザードウィンドウ内の「ヘルプ (Help)」をクリックして Web ブラウザを開き、「ネットワークへアドレスの追加 (Add Addresses to Network)」ウィザードのヘルプを表示します。
8. 選択した項目を確認し、「完了 **(Finish)**」をクリックしてネットワークテーブルに **IP** アドレスを追加します。

   指定した範囲内にある各アドレスのレコードで、ネットワークテーブルが更新されます。

参照　「ネットワークウィザード (Network Wizard)」を使えば、ほかのネットワークを DHCP サーバーに追加できます (381 ページの「DHCP ネットワークの追加」を参照)。

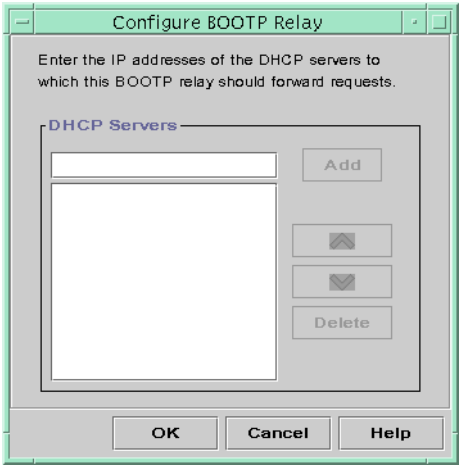
BOOTP リレーエージェントの構成

BOOTP リレーエージェントを構成する場合は、DHCP マネージャが次の動作を行います。

- 要求をリレーすべき 1 つまたは複数の DHCP サーバーの IP アドレスを入力するように求める
- BOOTP リレーサービスに必要な設定を格納する

次に、BOOTP リレーエージェントの構成を選択した場合に表示される画面を示します。

図 14–3　DHCP マネージャの「BOOTP リレーの構成 (Configure BOOTP Relay)」



▼ BOOTP リレーエージェントを構成する方法 (DHCP マネージャ)

始める前に　BOOTP リレーエージェントを構成する前に第 13 章「DHCP サービスの使用計画 (手順)」を必ずお読みください。とりわけ、使用するシステムの選択には、331 ページの「DHCP サービスを実行するホストの選択」が役立ちます。

1　サーバーシステム上でスーパーユーザーになります。

2 **DHCP マネージャを起動します。**

```
#/usr/sadm/admin/bin/dhcpmgr &
```

システムが DHCP サーバーか BOOTP リレーエージェントとして構成されていないと、「DHCP 構成ウィザード (DHCP Configuration Wizard)」が起動されます。システムがすでに DHCP サーバーとして構成されている場合は、まずサーバーを構成解除する必要があります。詳細は、349 ページの「DHCP サーバーと BOOTP リレーエージェントの構成解除」を参照してください。

3 **「BOOTP リレーとして構成 (Configure as BOOTP Relay)」を選択します。**

「BOOTP リレーの構成 (Configure BOOTP Relay)」ダイアログボックスが表示されます。

4 **1 つまたは複数の DHCP サーバーの IP アドレスかホスト名を入力し、「追加 (Add)」をクリックします。**

指定する DHCP サーバーは、この BOOTP リレーエージェントによって受信される BOOTP 要求や DHCP 要求を処理できるように構成されていなければなりません。

5 **「了解 (OK)」をクリックして、ダイアログボックスを終了します。**

DHCP マネージャはアプリケーションを終了するための「ファイル (File)」メニューと、サーバーを管理するための「サービス (Service)」メニューだけを表示します。無効になっているメニューオプションは DHCP サーバーだけで使用されます。

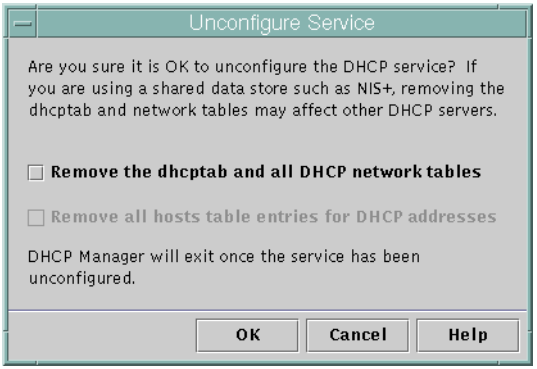
DHCP サーバーと BOOTP リレーエージェントの構成解除

DHCP サーバーまたは BOOTP リレーエージェントを構成解除するとき、DHCP マネージャは次の動作を行います。

- DHCP デーモン (`in.dhcpcd`) プロセスを停止する
- デーモンの起動に関する情報とデータストアの場所を記録している `/etc/inet/dhcpsvc.conf` ファイルを削除する

次に、DHCP サーバーの構成解除を選択した場合の画面を示します。

図 14–4 DHCP マネージャの「サービスの解除 (Unconfigure Service)」ダイアログボックス



構成解除したサーバー上の DHCP データ

DHCP サーバーを構成解除するときには、dhcptab テーブルと DHCP ネットワークテーブルをどのように処理するかを決定する必要があります。そのデータがサーバー間で共有されている場合は、dhcptab とDHCP ネットワークテーブルを削除しないでください。これらのテーブルを削除すると、DHCP がネットワーク全体で使用できなくなります。データの共有は、NIS+ またはエクスポートしたローカルファイルシステムを使用して行うことができます。/etc/inet/dhcpsvc.conf ファイルには、使用されるデータストアとその場所が記録されています。

データを削除するためのいずれのオプションも選択しなければ、データをそのままの形で残し、DHCP サーバーを構成解除できます。サーバーを構成解除し、データをそのままの形で残す場合は、DHCP サーバーを無効にします。

IP アドレスの所有権を別の DHCP サーバーに移したい場合は、DHCP データをその DHCP サーバーに移動する必要があります。その場合には、現在のサーバーを構成解除する前にデータを移動してください。詳細は、434 ページの「DHCP サーバー間での構成データの移動 (タスクマップ)」を参照してください。

データを削除したい場合は、dhcptab およびネットワークテーブルを削除するためのオプションを選択します。DHCP アドレス用のクライアント名がすでに作成されている場合には、このようなエントリを hosts テーブルから削除することもできます。クライアント名のエントリは、DNS、/etc/inet/hosts 、または NIS+ から削除できます。

BOOTP リレーエージェントを構成解除する前に、DHCP サーバーへ要求を転送するために、このエージェントを使用しているクライアントが存在しないことを確認してください。

▼ DHCP サーバーまたは BOOTP リレーエージェントを構成解除する方法 (DHCP マネージャ)

1. スーパーユーザーになります。
2. DHCP マネージャを起動します。

   ```
   #/usr/sadm/admin/bin/dhcpmgr &
   ```

3. 「サービス (Service)」メニューから、「構成解除 (Unconfigure)」を選択します。

   「サービスの解除 (Unconfigure Service)」ダイアログボックスが表示されます。サーバーが BOOTP リレーエージェントの場合、このダイアログボックスでリレーエージェントを構成解除することを確認できます。サーバーが DHCP サーバーの場合、DHCP データをどうするかを決定し、このダイアログボックスで選択する必要があります。図 14–4 を参照してください。

4. (省略可能) データを削除するためのオプションを選択します。

   サーバーが共有データ (NIS+ 経由で共有されるデータ、または NFS 経由で共有されるファイル) を使用する場合、データを削除するオプションは選択しないでください。サーバーが共有データを使用しない場合、データを削除するオプションの 1 つまたは両方を選択します。

   データの削除については、350 ページの「構成解除したサーバー上の DHCP データ」を参照してください。

5. 「了解 (OK)」をクリックしてサーバーを構成解除します。

   「サービスの解除 (Unconfigure Service)」ダイアログボックスと DHCP マネージャが終了します。

DHCP サーバーの構成と構成解除 (dhcpconfig コマンド)

この節では、dhcpconfig とコマンド行オプションを使用して、DHCP サーバーまたは BOOTP リレーエージェントを構成または構成解除する手順について説明します。

▼ DHCP サーバーを構成する方法 (dhcpconfig -D)

始める前に DHCP サーバーを構成する前に第 13 章「DHCP サービスの使用計画 (手順)」を必ずお読みください。特に、次のタスクを行う際には、330 ページの「DHCP サーバーの構成前に必要な選択 (タスクマップ)」のガイドラインが役立ちます。

- DHCP サーバーとして使用するシステムを選択します。
- データストア、リースポリシー、ルーター情報について決定します。

1 **DHCP** サーバーの構成を行うシステムにログインします。

2 スーパーユーザーになるか、**DHCP** 管理プロファイルに割り当てられている役割またはユーザー名になります。

DHCP 管理プロファイルの詳細については、359 ページの「DHCP コマンドへのユーザーアクセスの設定」を参照してください。

役割には、認証と特権コマンドが含まれます。役割の詳細については、『Solaris のシステム管理: セキュリティーサービス』の「RBAC の構成 (タスクマップ)」を参照してください。

3 次の書式のコマンドを入力して **DHCP** サーバーを構成します。

#/usr/sbin/dhcpconfig -D -r *datastore* **-p** *location*

datastore は次のどれかです。 `SUNWfiles` 、`SUNWbinfiles`、または `SUNWnisplus`。

location には、(データストアによって異なる) DHCP データを保存したい場所を指定します。`SUNWfiles` および `SUNWbinfiles` の場合、この場所は絶対パス名で指定する必要があります。`SUNWnisplus` の場合、この場所は完全指定の NIS+ ディレクトリで指定する必要があります。

たとえば、次のようなコマンドを入力します。

```
dhcpconfig -D -r SUNWbinfiles -p /var/dhcp
```

`dhcpconfig` ユーティリティーは、ホストのシステムファイルとネットワークファイルを使って、DHCP サーバーの構成に使用する値を決定します。デフォルトの値を変更できる `dhcpconfig` コマンドのその他のオプションについては、`dhcpconfig(1M)` のマニュアルページを参照してください。

4 1 つまたは複数のネットワークを **DHCP** サービスに追加します。

ネットワークを追加する手順については、383 ページの「DHCP ネットワークを追加する方法 (dhcpconfig)」を参照してください。

▼ BOOTP リレーエージェントを構成する方法 (dhcpconfig -R)

始める前に 331 ページの「DHCP サービスを実行するホストの選択」に記載されている要件に従って、BOOTP リレーエージェントとして使用するシステムを選択します。

1 **BOOTP** リレーエージェントとして構成したいサーバーにログインします。

2 スーパーユーザーになるか、**DHCP** 管理プロファイルに割り当てられている役割またはユーザー名になります。

DHCP 管理プロファイルの詳細については、359 ページの「DHCP コマンドへのユーザーアクセスの設定」を参照してください。

役割には、認証と特権コマンドが含まれます。役割の詳細については、『Solaris のシステム管理: セキュリティーサービス』の「RBAC の構成 (タスクマップ)」を参照してください。

3 次の書式のコマンドを入力して **BOOTP** リレーエージェントを構成します。

```
# /usr/sbin/dhcpconfig -R server-addresses
```

要求を転送する DHCP サーバーの 1 つまたは複数の IP アドレスを指定します。複数のアドレスを指定する場合には、アドレス間をコンマで区切ります。

たとえば、次のようなコマンドを入力します。

```
/usr/sbin/dhcpconfig -R 192.168.1.18,192.168.42.132
```

▼ DHCP サーバーまたは BOOTP リレーエージェントを構成解除する方法 (dhcpconfig -U)

1 構成解除したい **DHCP** サーバーまたは **BOOTP** リレーエージェントシステムにログインします。

2 スーパーユーザーになるか、**DHCP** 管理プロファイルに割り当てられている役割またはユーザー名になります。

DHCP 管理プロファイルの詳細については、359 ページの「DHCP コマンドへのユーザーアクセスの設定」を参照してください。

役割には、認証と特権コマンドが含まれます。役割の詳細については、『Solaris のシステム管理: セキュリティーサービス』の「RBAC の構成 (タスクマップ)」を参照してください。

3 **DHCP** サーバーまたは **BOOTP** リレーエージェントを構成解除します。

```
# /usr/sbin/dhcpconfig -U
```

サーバーが共有データを使用していない場合は、`-x` オプションを使って、構成解除と同時に `dhcptab` テーブルとネットワークテーブルを削除できます。サーバーが共有データを使用する場合、`-x` オプションは使用しないでください。`-h` オプションを使用すると、ホスト名をホストテーブルから削除できます。`dhcpconfig` のオプションについては、`dhcpconfig(1M)` のマニュアルページを参照してください。

データの削除については、350 ページの「構成解除したサーバー上の DHCP データ」を参照してください。

◆ ◆ ◆ 第 15 章

DHCP の管理 (タスク)

この章では、DHCP サービスを管理するときに便利なタスクについて説明します。この章では、サーバー、BOOTP リレーエージェント、およびクライアントに関するタスクを説明します。各タスクごとに、DHCP マネージャを使用する手順と DHCP コマンド行ユーティリティーを使用する手順を説明します。DHCP コマンド行ユーティリティーについての詳細は、マニュアルページを参照してください。

この章に進む前に、DHCP サービスとネットワークの初期構成を済ませておく必要があります。DHCP の構成については、第 14 章「DHCP サービスの構成 (手順)」を参照してください。

この章では、次の内容について説明します。

DHCP マネージャーについて

DHCP マネージャは、DHCP サービスで管理タスクを実行するために使用するグラフィカルユーザーインタフェース (GUI) ツールです。

DHCP マネージャーウィンドウ

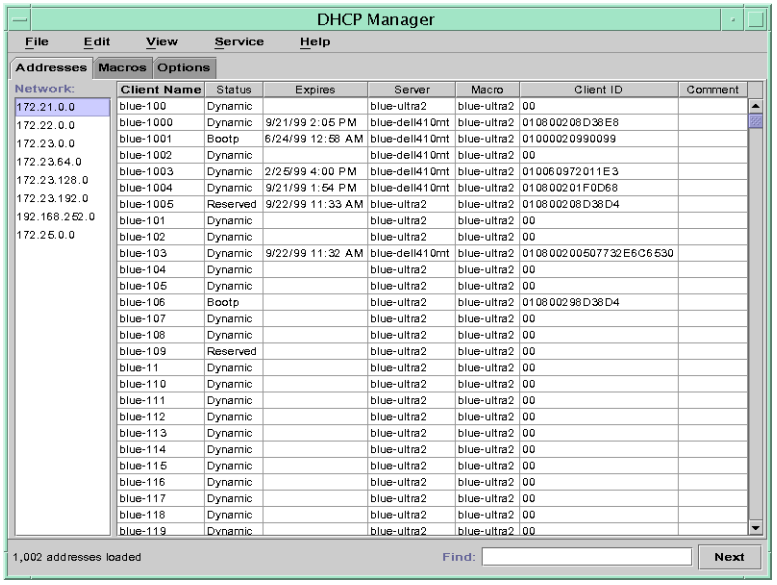
DHCP マネージャウィンドウの外観は、DHCP マネージャが動作するシステムで DHCP サーバーがどのように構成されているかによって異なります。

システムが DHCP サーバーとして構成されている場合、DHCP マネージャはタブ形式のウィンドウを使用します。作業に応じて適切なタブを選択してください。DHCP マネージャには次のタブがあります。

- 「アドレス (Addresses)」タブ – DHCP が管理しているすべてのネットワークと IP アドレスをリストします。「アドレス (Addresses)」タブでは、ネットワークや IP アドレスを操作できます。追加や削除は、1 つずつ行うこともブロック単位で行うこともできます。また、各ネットワークや IP アドレスの属性を変更したり、アドレスをまとめて同時に同じ属性に変更したりできます。DHCP マネージャを起動すると、「アドレス (Addresses)」タブが最初に開かれます。
- 「マクロ (Macros)」タブ – DHCP 構成テーブル (`dhcptab`) で利用できるすべてのマクロと、それらのマクロに含まれるオプションをリストします。「マクロ (Macros)」タブでは、マクロの作成や削除ができます。さらに、オプションを追加し、その値を設定することによってマクロを変更できます。
- 「オプション (Options)」タブ – この DHCP サーバーについて定義されたすべてのオプションをリストします。このタブで表示されるオプションは、DHCP プロトコルで定義された標準的なオプションではありません。「拡張 (Extended)」、「ベンダー (Vendor)」、または「サイト (Site)」のクラスを持つ、標準オプションを拡張したものです。標準オプションは変更できないため、このタブには表示されません。

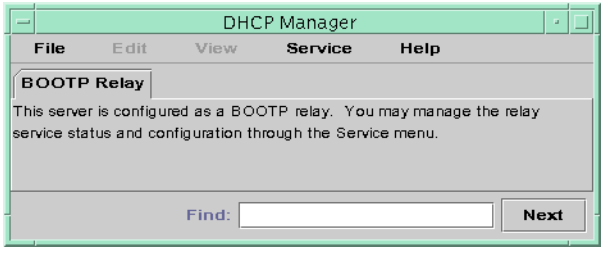
DHCP サーバーで DHCP マネージャを起動すると、次のような DHCP マネージャのウィンドウが表示されます。

図 15–1　DHCP サーバーシステム上の DHCP マネージャ



サーバーが BOOTP リレーエージェントとして構成されている場合、DHCP マネージャウィンドウにはこれらのタブは表示されません。BOOTP リレーエージェントでは、これらのタブの情報は必要ないからです。BOOTP リレーエージェントの属性を変更し、DHCP マネージャを使用して DHCP デーモンを停止または起動することだけが可能です。次の図は、BOOTP リレーエージェントとして構成されたシステム上でブートした場合の DHCP マネージャウィンドウです。

図 15–2　BOOTP リレーエージェント上の DHCP マネージャ



DHCP マネージャーのメニュー

DHCP マネージャのメニューには、次の項目が含まれます。

- 「ファイル **(File)**」 – DHCP マネージャを終了します。
- 「編集 **(Edit)**」 – ネットワーク、アドレス、マクロ、およびオプションについて管理作業を実行します。
- 「表示 **(View)**」 – 現在選択されているタブの表示を変更します。
- 「サービス **(Service)**」 – DHCP デーモンとデータストアを管理します。
- 「ヘルプ **(Help)**」 – Web ブラウザを開いて、DHCP マネージャのヘルプを表示します。

DHCP マネージャが BOOTP リレーエージェントで実行されている場合、「編集 (Edit)」メニューと「表示 (View)」メニューは使用できません。

すべての DHCP 管理機能は、「編集 (Edit)」メニューと「サービス (Service)」メニューで実行されます。

「編集 (Edit)」メニューにあるコマンドを使用して、選択されているタブの項目を作成、削除、変更できます。この項目には、ネットワーク、アドレス、マクロ、オプションがあります。また、「アドレス (Addresses)」タブが選択されている場合、「編集 (Edit)」メニューはウィザードも表示します。このウィザードは、ネットワークと複数の IP アドレスを容易に作成できるダイアログのセットです。

「サービス (Service)」メニューは、DHCP デーモンを管理するためのコマンドを表示します。「サービス (Service)」メニューでは、次のタスクを実行できます。

- DHCP デーモンを起動および停止します。
- DHCP デーモンを有効化および無効化します。
- サーバー構成を変更します。
- サーバー構成を解除します。
- データストアを変換します。
- サーバー上のデータをエクスポートおよびインポートします。

DHCP マネージャの起動と停止

DHCP マネージャを DHCP サーバーシステムで実行する場合、スーパーユーザーとして実行する必要があります。DHCP マネージャをリモートから実行する必要がある場合には、X Window リモート表示機能を使用して、表示画面を使用しているシステムに送信できます。

▼ DHCP マネージャを起動および停止する方法

1 **DHCP** サーバーシステムでスーパーユーザーになります。

2 (省略可能) リモートで DHCP サーバーシステムにログインしている場合、次の手順でローカルのシステムに DHCP マネージャを表示できます。

a. ローカルシステムで次のように入力します。

```
# xhost +server-name
```

b. リモートの DHCP サーバーシステムで次のように入力します。

```
# DISPLAY=local-hostname;export DISPLAY
```

3 DHCP マネージャを起動します。

```
# /usr/sadm/admin/bin/dhcpmgr &
```

DHCP マネージャウィンドウが開きます。サーバーが DHCP サーバーとして構成されている場合には、「アドレス (Addresses)」タブが表示されます。サーバーが BOOTP リレーエージェントとして構成されている場合には、タブは表示されません。

4 DHCP マネージャを停止するには、「ファイル (File)」メニューから「終了 (Exit)」を選択します。

DHCP マネージャウィンドウが閉じます。

DHCP コマンドへのユーザーアクセスの設定

デフォルトでは、`dhcpconfig`、`dhtadm`、および `pntadm` コマンドを実行できるのは、`root` ユーザーだけです。これらのコマンドを `root` ユーザー以外で使用する場合は、これらのコマンドに対して、役割によるアクセス制御 (RBAC) を設定する必要があります。

役割には、認証と特権コマンドが含まれます。役割の詳細については、『Solaris のシステム管理: セキュリティーサービス』の「RBAC の構成 (タスクマップ)」を参照してください。

さらに、次のマニュアルページも役立ちます。`rbac(5)`、`exec_attr(4)`、および `user_attr(4)` です。

次の手順では、ユーザーが DHCP コマンドを実行できるようになる DHCP 管理プロファイルを割り当てる方法について説明します。

▼ ユーザーに DHCP コマンドへのアクセス権を付与する方法

1 **DHCP** サーバーシステム上でスーパーユーザーになります。

2 ユーザーまたは役割を **/etc/user_attr** ファイルに追加します。

`/etc/user_attr` ファイルを編集してエントリを次の書式で追加します。DHCP サーバーを管理するユーザーまたは役割ごとに 1 つのエントリを追加します。

username`::::type=normal;profiles=DHCP Management`

たとえば、ユーザー `ram` には、次のエントリを追加します。

```
ram::::type=normal;profiles=DHCP Management
```

DHCP サーバーのタスク

▼ ISC DHCP サーバーを構成する方法

ISC DHCP サーバーの初期構成には次の手順を使用できます。

1 スーパーユーザーになるか、**DHCP** 管理プロファイルに割り当てられている役割またはユーザー名になります。

役割には、認証と特権コマンドが含まれます。役割の詳細については、『Solaris のシステム管理: セキュリティーサービス』の「RBAC の構成 (タスクマップ)」を参照してください。

2 **DHCP** 構成ファイルを編集します。

`/etc/dhcp/dhcpd4.conf` または `/etc/dhcp/dhcpd6.conf` ファイルのいずれかを作成します。詳細は、`dhcpd.conf`(5) のマニュアルページを参照してください。

3 必要なサービスを有効にします。

```
# svcadm enable service
```

service は次のいずれかの値にすることができます。

`svc:/network/dhcp/server:ipv4`	IPv4 クライアントからの DHCP および BOOTP 要求を提供します
`svc:/network/dhcp/server:ipv6`	IPv6 クライアントからの DHCP および BOOTP 要求を提供します

`svc:/network/dhcp/relay:ipv4`	IPv4 クライアントからの DHCP および BOOTP 要求を、DHCP サーバーのあるネットワークに中継します
`svc:/network/dhcp/relay:ipv6`	IPv6 クライアントからの DHCP および BOOTP 要求を、DHCP サーバーのあるネットワークに中継します

▼ DHCP サービスの構成を変更する方法

1. スーパーユーザーになるか、**DHCP** 管理プロファイルに割り当てられている役割またはユーザー名になります。

   役割には、認証と特権コマンドが含まれます。役割の詳細については、『Solaris のシステム管理: セキュリティーサービス』の「RBAC の構成 (タスクマップ)」を参照してください。

2. **DHCP** 構成ファイルを編集します。

   `/etc/dhcp/dhcpd4.conf` ファイルまたは `/etc/dhcp/dhcpd6.conf` ファイルのいずれかを編集します。詳細は、`dhcpd.conf`(5) のマニュアルページを参照してください。

3. **SMF** データをリフレッシュします。

   ```
   # svcadm refresh service
   ```

DHCP サービスの起動と停止

この節では、DHCP マネージャや `dhcpconfig` コマンドを使用した DHCP サービスの起動や停止を説明します。DHCP サービスの起動や停止は、サービス管理機能 (SMF) コマンドでもできます。DHCP サービスで SMF コマンドを使用する方法については、364 ページの「DHCP サービスとサービス管理機能」を参照してください。

DHCP サービスの起動と停止には、DHCP デーモンの動作に影響する可能性がある処理をいくつか実行する必要があります。正しい手順を選択して希望する結果を得るためには、それぞれの処理の意味を理解しておく必要があります。これらの処理の意味は次のとおりです。

- 「起動、停止、再起動コマンド」は、現在のセッションのデーモンだけに影響します。つまり、DHCP サービスを停止すると現在実行中のデーモンは終了しますが、システムをリブートすると終了したデーモンは再び起動します。サービスを停止しても、DHCP データテーブルは影響されません。DHCP マネージャや SMF コマンドを使って、サービスを有効にも無効にもすることなく、DHCP サービスを一時的に起動したり停止したりできます。
- 「有効コマンドと無効コマンド」は、現在のセッションと将来のセッションのデーモンに影響します。DHCP サービスを無効にすると、現在実行中のデーモンは終了し、サーバーをリブートしても終了したデーモンは起動しません。DHCP デーモンがシステム起動時に自動的に起動するように設定しておく必要があります。DHCP データテーブルは影響されません。DHCP サービスを有効または無効にするには、DHCP マネージャ、`dhcpconfig` コマンド、または SMF コマンドを使用できます。
- 「構成解除コマンド」は、デーモンをシャットダウンし、システムのリブート時にデーモンが起動されないようにし、DHCP データテーブルを削除できるようにします。DHCP サービスを構成解除する操作には、DHCP マネージャや `dhcpconfig` コマンドが使用できます。構成解除については、第 14 章「DHCP サービスの構成 (手順)」を参照してください。

注 – サーバーに複数のネットワークインタフェースがあり、すべてのネットワークでは DHCP サービスを提供したくない場合、378 ページの「DHCP で監視するネットワークインタフェースの指定」を参照してください。

次に、DHCP サービスを起動および停止、有効および無効にする手順を説明します。

▼ DHCP サービスを起動および停止する方法 (DHCP マネージャ)

1. **DHCP サーバーシステム上でスーパーユーザーになります。**
2. **DHCP マネージャを起動します。**

```
# /usr/sadm/admin/bin/dhcpmgr &
```

3 次のいずれかの手順に従います。

- 「サービス (Service)」メニューから「起動 (Start)」を選択して、DHCP サービスを起動します。
- 「サービス (Service)」メニューから「停止 (Stop)」を選択して、DHCP サービスを停止します。
  DHCP デーモンは、再開されるかシステムがリブートするまで停止します。
- 「サービス (Service)」メニューから「再開 (Restart)」を選択して、DHCP サービスを停止しすぐに再起動します。

▼ DHCP サービスを有効または無効にする方法 (DHCP マネージャ)

- DHCP マネージャの中で次の 1 つを選択します。
  - 「サービス (Service)」メニューの「有効 (Enable)」を選択して、システム起動時に DHCP デーモンが自動的に起動するようにします。
    有効を選択すると、DHCP サービスは直ちに起動します。
  - 「サービス (Service)」メニューの「無効 (Disable)」を選択して、システム起動時に DHCP デーモンが自動的に起動しないようにします。
    無効を選択すると、DHCP サービスは直ちに停止します。

▼ DHCP サービスを有効または無効にする方法 (dhcpconfig -S)

1 DHCP サーバーシステムにログインします。

2 スーパーユーザーになるか、DHCP 管理プロファイルに割り当てられている役割またはユーザー名になります。
  DHCP 管理プロファイルの詳細については、359 ページの「DHCP コマンドへのユーザーアクセスの設定」を参照してください。
  役割には、認証と特権コマンドが含まれます。役割の詳細については、『Solaris のシステム管理: セキュリティーサービス』の「RBAC の構成 (タスクマップ)」を参照してください。

3 次のいずれかを選択してください。

- **DHCP** サービスを有効にするには、次のコマンドを入力します。

  ```
  # /usr/sbin/dhcpconfig -S -e
  ```

- **DHCP** サービスを無効にするには、次のコマンドを入力します。

  ```
  # /usr/sbin/dhcpconfig -S -d
  ```

DHCP サービスとサービス管理機能

サービス管理機能 (SMF) については、『Oracle Solaris の管理: 基本管理』の第 18 章「サービスの管理 (概要)」を参照してください。SMF の `svcadm` コマンドを使用すると、DHCP サーバーを有効にして起動したり、DHCP サーバーを無効にして停止したりできます。ただし、DHCP ツールで設定できる DHCP サービスオプションを SMF コマンドで変更することはできません。つまり、`/etc/dhcp/dhcpsvc.conf` ファイルに格納されているサービスオプションを SMF ツールで設定することはできません。

次の表は、DHCP コマンドと、それと同等の SMF コマンドを対比したものです。

表 15–1 DHCP サーバーのタスクに使用する SMF コマンド

タスク	**DHCP** のコマンド	**SMF** コマンド
DHCP サービスを有効にします	`dhcpconfig -S -e`	`svcadm enable svc:/network/dhcp-server`
DHCP サービスを無効にします	`dhcpconfig -S -d`	`svcadm disable svc:/network/dhcp-server`
DHCP サービスを起動します (現在のセッションのみ)	なし	`svcadm enable -t svc:/network/dhcp-server`
DHCP サービスを停止します (現在のセッション)	なし	`svcadm disable -t svc:/network/dhcp-server`
DHCP サービスを再開します	`dhcpconfig -S -r`	`svcadm restart svc:/network/dhcp-server`

DHCP サービスオプションの変更 (タスクマップ)

DHCP サービスの機能には、DHCP マネージャを使用した初期構成の際にはできない場合でも、あとから値を変更できるものがあります。サービスオプションを変更する場合には、DHCP マネージャの「サービスオプションの変更 (Modify Service Options)」ダイアログが使用できます。あるいは、`dhcpconfig` コマンドでオプションを指定することもできます。

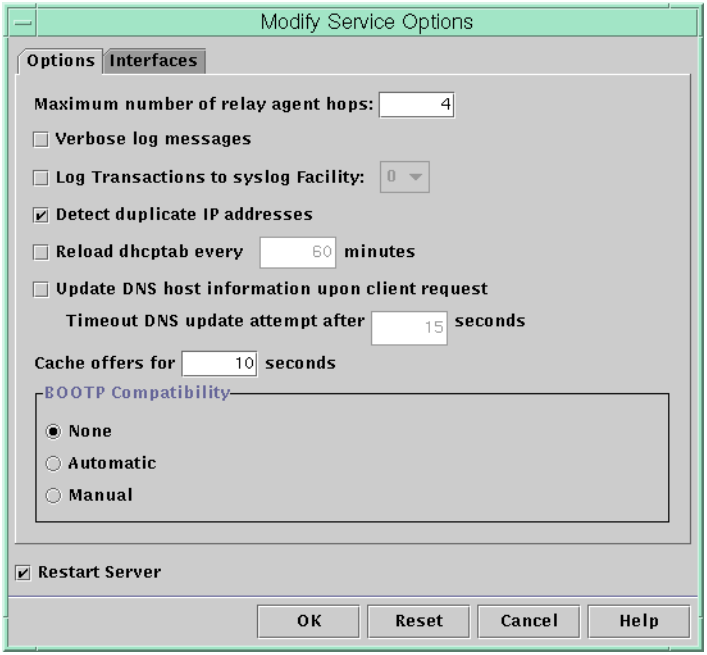
次の表のマップでは、DHCP サービスオプションを変更するためのタスクについて説明しています。この表には、各タスクを完了するための手順へのリンクも含まれています。

タスク	説明	参照先
ロギングオプションを変更します。	ログを有効または無効にし、さらに DHCP トランザクションのログに使用する `syslog` 機能を選択します。	368 ページの「詳細 DHCP ログメッセージを生成する方法 (DHCP マネージャー)」 369 ページの「詳細 DHCP ログメッセージを生成する方法 (コマンド行)」 369 ページの「DHCP トランザクションログを有効または無効にする方法 (DHCP マネージャ)」 370 ページの「DHCP トランザクションログを有効または無効にする方法 (コマンド行)」 371 ページの「DHCP トランザクションを別の `syslog` ファイルに記録する方法」
DNS 更新オプションを変更します。	クライアント用の DNS エントリ (ホスト名を指定する) を動的に追加するサーバーの機能を有効または無効にします。DNS を更新する際にサーバーが使用できる最大の時間を決めます。	373 ページの「DHCP クライアント用に動的 DNS 更新を有効にする方法」
重複 IP アドレス検出を有効または無効にします。	DHCP サーバーが IP アドレスをクライアントに提供する前に IP アドレスがまだ使用されていないことを確認できるようにするかどうかを指定します。	376 ページの「DHCP パフォーマンスオプションをカスタマイズする方法 (DHCP マネージャ)」 377 ページの「DHCP パフォーマンスオプションをカスタマイズする方法 (コマンド行)」
DHCP サーバーの構成情報の読み込みに関するオプションを変更します。	指定された間隔での `dhcptab` の自動読み込みを有効または無効にします。また、読み込み間隔を変更します。	376 ページの「DHCP パフォーマンスオプションをカスタマイズする方法 (DHCP マネージャ)」 377 ページの「DHCP パフォーマンスオプションをカスタマイズする方法 (コマンド行)」

タスク	説明	参照先
リレーエージェントホップ数を変更します。	DHCP デーモンでドロップされる前に要求をやり取りできるネットワーク数を増減します。	376 ページの「DHCP パフォーマンスオプションをカスタマイズする方法 (DHCP マネージャ)」 377 ページの「DHCP パフォーマンスオプションをカスタマイズする方法 (コマンド行)」
提供される IP アドレスがキャッシュされている時間を変更します。	新しいクライアントに IP アドレスを提供する前に DHCP サービスが提供された IP アドレスを予約する秒数を増減します。	376 ページの「DHCP パフォーマンスオプションをカスタマイズする方法 (DHCP マネージャ)」 377 ページの「DHCP パフォーマンスオプションをカスタマイズする方法 (コマンド行)」

次に、DHCP マネージャの「サービスオプションの変更 (Modify Service Options)」ダイアログボックスを示します。

図 15–3 DHCP マネージャの「サービスオプションの変更 (Modify Service Options)」ダイアログボックス



DHCP ログオプションの変更

DHCP サービスは、DHCP サービスメッセージと DHCP トランザクションを syslog に記録できます。syslog についての詳細は、syslogd(1M) および syslog.conf(4) のマニュアルページを参照してください。

syslog に記録される DHCP サービスメッセージには、次のものがあります。

- エラーメッセージ。DHCP サービスがクライアントまたは管理者の要求を完了するのを妨げる条件を、管理者に通知します。
- 警告と通知。DHCP サービス完了を妨げはしないが、正常終了しなかった状態を管理者に通知します。

DHCP デーモンの詳細オプションを使用して、報告される情報を増やすことができます。詳細メッセージ出力は、DHCP に関する問題の障害追跡に役立つ場合があります。詳細は、368 ページの「詳細 DHCP ログメッセージを生成する方法 (DHCP マネージャー)」を参照してください。

もう 1 つの有用な障害追跡方法は、トランザクションの記録です。トランザクションは、DHCP サーバーや BOOTP リレーとクライアントとの間のすべての交換に関する情報を提供します。DHCP トランザクションのメッセージには、次のタイプがあります。

- ASSIGN – IP アドレスの割り当て
- ACK – サーバーは、クライアントが提供された IP アドレスを受け入れることを認め、構成パラメータを送る
- EXTEND – リース期間の延長
- RELEASE – IP アドレスの解放
- DECLINE – クライアントはアドレス割り当てを拒否している
- INFORM – クライアントはネットワーク構成パラメータを要求しているが IP アドレスは要求していない
- NAK – サーバーは、クライアントに対して、すでに使用された IP アドレスの使用要求を認めない
- ICMP_ECHO – サーバーは、可能性のある IP アドレスがほかのホストですでに使用中であることを検出する

BOOTP リレートランザクションのメッセージには、次のタイプがあります。

- RELAY-CLNT – メッセージは DHCP クライアントから DHCP サーバーへリレーされる
- RELAY-SRVR – メッセージは DHCP サーバーから DHCP クライアントへリレーされる

DHCP トランザクションのログは、デフォルトでは使用不可になっています。DHCP トランザクションのログが使用可能になっていると、デフォルトで `syslog` の `local0` 機能が使用されます。DHCP トランザクションメッセージは、`syslog` 重大度レベル *notice* (通知) で生成されます。セキュリティーレベルがこのレベルにあると、DHCP トランザクションのログは、ほかのシステム通知のログと同じファイルに書き込まれます。しかし、`local` 機能を使用しているため、DHCP トランザクションメッセージのログはほかのシステム通知とは別のファイルに書き込むことができます。トランザクションメッセージのログを別のファイルに書き込むには、`syslog.conf` ファイルを編集して、別のログファイルを指定する必要があります。`syslog.conf` ファイルについての詳細は、`syslog.conf(4)` のマニュアルページを参照してください。

DHCP トランザクションログは有効または無効にできます。さらに、異なる `syslog` 機能 (`local0` から `local7` まで) を指定できます。詳細は、369 ページの「DHCP トランザクションログを有効または無効にする方法 (DHCP マネージャ)」を参照してください。サーバーシステムの `syslog.conf` ファイルを構成すると、`syslogd` が DHCP トランザクションメッセージを格納するファイルを変更できます。詳細は、371 ページの「DHCP トランザクションを別の `syslog` ファイルに記録する方法」を参照してください。

▼ 詳細 DHCP ログメッセージを生成する方法 (DHCP マネージャー)

1. **DHCP マネージャでは、「サービス (Service)」メニューから「変更 (Modify)」を選択する**

   358 ページの「DHCP マネージャを起動および停止する方法」を参照してください。

   「サービスオプションの変更 (Modify Service Options)」ダイアログボックスが開かれ、そこに「オプション (Options)」タブが表示されます。図 15–3 を参照してください。

2. **「詳細ログメッセージ (Verbose Log Messages)」を選択します。**

3. **「サーバーの再起動」を選択します。**

   「サーバーの再起動 (Restart Server)」オプションはダイアログボックス下部にあります。

4. **「了解」をクリックします。**

   このセッション以降、このオプションを再設定するまで、デーモンは詳細モードで動作します。メッセージを表示するのに時間がかかるため、詳細モードではデーモンの効率が低下する場合があります。

▼ 詳細 DHCP ログメッセージを生成する方法 (コマンド行)

1 スーパーユーザーになるか、**DHCP** 管理プロファイルに割り当てられている役割またはユーザー名になります。

DHCP 管理プロファイルの詳細については、359 ページの「DHCP コマンドへのユーザーアクセスの設定」を参照してください。

役割には、認証と特権コマンドが含まれます。役割の詳細については、『Solaris のシステム管理: セキュリティーサービス』の「RBAC の構成 (タスクマップ)」を参照してください。

2 次のコマンドを入力して詳細モードにします。

```
# /usr/sbin/dhcpconfig -P VERBOSE=true
```

次回の DHCP サーバーの起動からサーバーは詳細モードで実行されます (詳細モードがオフにされるまで)。

詳細モードをオフに設定する場合は、次のコマンドを入力します。

```
# /usr/sbin/dhcpconfig -P VERBOSE=
```

このコマンドは、VERBOSE キーワードを値なしに設定します。すると、このキーワードはサーバーの構成ファイルから削除されます。

メッセージを表示するのに時間がかかるため、詳細モードではデーモンの効率が低下する場合があります。

▼ DHCP トランザクションログを有効または無効にする方法 (DHCP マネージャ)

この手順では、以後すべての DHCP サーバーセッションに関するトランザクションログを有効または無効にします。

1 **DHCP** マネージャでは、「サービス (**Service**)」メニューから「変更 (**Modify**)」を選択する

358 ページの「DHCP マネージャを起動および停止する方法」を参照してください。

2 「**syslog** へのログトランザクション (**Log Transactions to Syslog Facility**)」を選択します。

トランザクションログを無効にするには、このオプションの選択を解除します。

3 (省略可能) ローカル機能を 0 から 7 まで選択して、DHCP トランザクションログに使用します。

デフォルトでは、DHCP トランザクションはシステム通知が書き込まれる場所に書き込まれます。システム通知が書き込まれる場所は syslogd で構成できます。DHCP トランザクションをほかのシステム通知とは別の場所に記録する場合は、371 ページの「DHCP トランザクションを別の syslog ファイルに記録する方法」を参照してください。

トランザクションログを有効にすると、メッセージファイルのサイズは急速に大きくなります。

4 「サーバーの再起動」を選択します。

5 「了解」をクリックします。

このセッション以降、ログを無効にするまで、デーモンは選択された syslog 機能にトランザクションを記録します。

▼ DHCP トランザクションログを有効または無効にする方法 (コマンド行)

1 スーパーユーザーになるか、DHCP 管理プロファイルに割り当てられている役割またはユーザー名になります。

DHCP 管理プロファイルの詳細については、359 ページの「DHCP コマンドへのユーザーアクセスの設定」を参照してください。

役割には、認証と特権コマンドが含まれます。役割の詳細については、『Solaris のシステム管理: セキュリティーサービス』の「RBAC の構成 (タスクマップ)」を参照してください。

2 次の手順から 1 つを選択します。

- DHCP トランザクションログを使用可能にする場合は、次のコマンドを入力します。

  ```
  # /usr/sbin/dhcpconfig -P LOGGING_FACILITY=syslog-local-facility
  ```

  syslog-local-facility は 0 - 7 の数字です。このオプションを省略すると、0 が使用されます。

  デフォルトでは、DHCP トランザクションはシステム通知が書き込まれる場所に書き込まれます。システム通知が書き込まれる場所は syslogd で構成できます。DHCP トランザクションをほかのシステム通知とは別の場所に記録する場合は、371 ページの「DHCP トランザクションを別の syslog ファイルに記録する方法」を参照してください。

トランザクションログを有効にすると、メッセージファイルのサイズは急速に大きくなります。

- **DHCP トランザクションログを使用不可にする場合は、次のコマンドを入力します。**

  ```
  # /usr/sbin/dhcpconfig -P LOGGING_FACILITY=
  ```

  パラメータの値を指定しません。

▼ DHCP トランザクションを別の syslog ファイルに記録する方法

1. **DHCP サーバーシステムでスーパーユーザーになるか、同等の役割を引き受けます。**

   役割には、認証と特権コマンドが含まれます。役割の詳細については、『Solaris のシステム管理: セキュリティーサービス』の「RBAC の構成 (タスクマップ)」を参照してください。

   DHCP 管理プロファイルに割り当てられている役割が、このタスクの実施に十分であるとは限りません。この役割には、`syslog` ファイルを編集する権限が必要です。

2. **サーバーシステムの `/etc/syslog.conf` ファイルを編集し、次の書式の行を追加します。**

   `local`*n*`.notice` *path-to-logfile*

   n にはトランザクションログ用に指定した `syslog` 機能番号を指定します。*path-to-logfile* には、トランザクションを記録するファイルへの絶対パスを指定します。

   たとえば、次のような行を追加できます。

   ```
   local0.notice /var/log/dhcpsrvc
   ```

   `syslog.conf` ファイルについての詳細は、`syslog.conf(4)` のマニュアルページを参照してください。

DHCP サーバーによる動的 DNS 更新の有効化

DNS はインターネット用にネームサービスを提供します。DNS マッピングが行われていれば、ホスト名からでも IP アドレスからでもシステムにアクセスできます。さらに、このシステムにはドメインの外からでもアクセスできます。

DHCP サービスでは、DNS を 2 つの方法で使用できます。

- DHCP サーバーは、クライアントに割り当てようとする IP アドレスにマップされているホスト名を検索できます。サーバーは、クライアントのホスト名とクライアントのそのほかの構成情報を返します。
- DHCP サーバーは、クライアントに代わって DNS マッピングを行うことができます。ただし、DHCP サーバーの構成で DNS の更新が可能になっていなければなりません。クライアントは、DHCP サービスを要求する際に独自のホスト名を指定できます。DNS の更新が可能に構成されていると、DHCP サーバーは指定されたホスト名を使って DNS を更新します。DNS 更新が正常なら、DHCP サーバーは要求されたホスト名をクライアントに返します。DNS 更新が正常でないなら、DHCP サーバーは異なるホスト名をクライアントに返します。

自身のホスト名を供給する DHCP クライアントのために DNS サービスを更新するように DHCP サービスを構成できます。DNS 更新機能が正しく働くためには、DNS サーバー、DHCP サーバー、DHCP クライアントがどれも正しく設定されていなければなりません。さらに、要求されたホスト名が、ドメインのほかのシステムによって使用されているものではなりません。

DHCP サーバーの DNS 更新機能が動作するのは、次の条件がすべて真であるときです。

- DNS サーバーが RFC 2136 をサポートする。
- DNS ソフトウェアが BIND v8.2.2 パッチレベル 5 以降のものである。DNS ソフトウェアが DHCP サーバーシステムで動作するか、DNS サーバーシステムで動作するかを問いません。
- DNS サーバーがDHCP サーバーからの動的 DNS 更新を受け入れるように構成されている。
- DHCP サーバーが動的 DNS 更新を行うように構成されている。
- DNS サポートが、DHCP サーバー上の DHCP クライアントのネットワーク用に構成されている。
- DHCP クライアントが、その DHCP 要求メッセージで要求されたホスト名を供給するように構成されている。
- 要求されたホスト名が、DHCP 所有のアドレスに対応している。そのホスト名に対応するアドレスがない場合もあります。

▼ DHCP クライアント用に動的 DNS 更新を有効にする方法

注 – 動的 DNS 更新は「セキュリティー上のリスク」であることに注意してください。

デフォルトでは、Oracle Solaris DNS デーモン (in.named) は動的更新を許可しません。動的 DNS 更新を行う権限は、DNS サーバーシステムの named.conf 構成ファイルで与えられます。ほかのセキュリティーは提供されません。動的 DNS 更新を有効にするときには、この機能のユーザーに対する便利さとセキュリティーリスクのバランスを注意深く考慮する必要があります。

1 **DNS サーバーで、スーパーユーザーとして /etc/named.conf ファイルを編集します。**

2 **named.conf ファイルで、適切なドメインの zone セクションを見つけます。**

3 **DHCP サーバーの IP アドレスを allow-update キーワードに追加します。**

allow-update キーワードが存在しない場合は、このキーワードを挿入します。

たとえば、DHCP サーバーのアドレスが 10.0.0.1 と 10.0.0.2 である場合、dhcp.domain.com ゾーンの named.conf ファイルを次のように変更します。

```
zone "dhcp.domain.com" in {
             type master;
             file "db.dhcp";
             allow-update { 10.0.0.1; 10.0.0.2; };
};

zone "10.IN-ADDR.ARPA" in {
             type master;
             file "db.10";
             allow-update { 10.0.0.1; 10.0.0.2; };
};
```

DHCP サーバーが A と PTR の両方のレコードを DNS サーバー上で更新できるように、両方のゾーンの allow-update を有効にする必要があります。

4 **DHCP サーバー上で、DHCP マネージャを起動します。**

```
# /usr/sadm/admin/bin/dhcpmgr &
```

詳細は、358 ページの「DHCP マネージャを起動および停止する方法」を参照してください。

5 **「サービス (Service)」メニューから「変更 (Modify)」を選択します。**

「サービスオプションの変更 (Modify Service Options)」ダイアログボックスが開きます。

6 「クライアント要求により **DNS** ホスト情報を更新 **(Update DNS Host Information Upon Client Request)**」を選択します。

7 **DNS** サーバーからの応答を待ち、時間切れになるまでの秒数を指定し、「了解 **(OK)**」をクリックします。

通常は、デフォルト値として 15 秒を指定すれば十分です。時間切れに関する問題が発生した場合は、あとでこの値を増やすことも可能です。

8 「マクロ **(Macros)**」タブをクリックして、正しい **DNS** ドメインが指定されていることを確認します。

`DNSdmain` オプションを渡すには、動的 DNS 更新のサポートを期待するクライアントへの正しいドメイン名と共に指定する必要があります。デフォルトでは、`DNSdmain` がサーバーマクロ中に指定されています。この値は、各 IP アドレス構成マクロとして使用されます。

9 **DHCP** サービスを要求するときにそのホスト名を指定するように **DHCP** クライアントを設定します。

DHCP クライアントを使用する場合は、458 ページの「DHCPv4 クライアントが特定のホスト名を要求できるようにする方法」を参照してください。DHCP クライアント以外のクライアントを使用する場合は、そのクライアントのドキュメントでホスト名の指定方法を確認してください。

クライアントホスト名の登録

DHCP サービスで使用する IP アドレスのホスト名を DHCP サーバーが生成するようにすると、DHCP サーバーがこれらのホスト名を NIS+、`/etc/inet/hosts`、または DNS ネームサービスに登録できます。ホスト名の登録を NIS で行うことはできません。NIS には、NIS マップの更新や伝達をプログラムで行うためのプロトコルが備わっていないからです。

注 – DNS サーバーと DHCP サーバーが同じシステムで動作している場合のみ、DHCP サーバーは、生成したホスト名を DNS に登録できます。

DHCP クライアントがそのホスト名を指定し、DHCP サーバーが動的に更新できるよう DNS サーバーが構成されている場合には、DHCP サーバーがクライアントに代わってDNS を更新できます。動的更新は、DNS サーバーとDHCP サーバーが異なるシステムで動作している場合でも行われます。この機能を有効にする方法については、371 ページの「DHCP サーバーによる動的 DNS 更新の有効化」を参照してください。

次の表に、さまざまなネームサービスによる DHCP クライアントシステムのホスト名の登録を要約します。

表 15–2 ネームサービスへのクライアントホスト名の登録

	ホスト名を登録する人または物	
ネームサービス	DHCP が生成したホスト名	DHCP クライアントが指定したホスト名
NIS	NIS 管理者	NIS 管理者
NIS+	DHCP ツール	DHCP ツール
`/etc/hosts`	DHCP ツール	DHCP ツール
DNS	DHCP ツール (DNS サーバーが DHCP サーバーと同じシステムで動作している場合) DNS 管理者 (DNS サーバーが異なるシステムで動作している場合)	DHCP サーバー (動的 DNS 更新が可能として構成されている場合) DNS 管理者 (動的 DNS 更新が可能として DHCP サーバーが構成されていない場合)

458 ページの「DHCPv4 クライアントが特定のホスト名を要求できるようにする方法」の説明通りに DHCP クライアントが DHCP 要求で特定のホスト名を指定できるように構成されていれば、その指定が可能となります。ほかの DHCP クライアントでこの機能がサポートされているかどうかについては、それぞれのベンダーのマニュアルを参照してください。

DHCP サーバー用のパフォーマンスオプションのカスタマイズ

DHCP サーバーのパフォーマンスに影響するオプションを変更できます。これらのオプションについて、次の表で説明します。

表 15–3 DHCP サーバーのパフォーマンスに影響するオプション

サーバーオプション	説明	キーワード
BOOTP リレーエージェントホップの最大数	一定数以上の BOOTP リレーエージェントを通過すると、その要求はドロップされます。デフォルトのリレーエージェントホップ数の最大は 4 です。ほとんどのネットワークは、この数で十分なはずです。DHCP 要求が DHCP サーバーに到達するまでにいくつかの BOOTP リレーエージェントを通過する場合は、ネットワークで 4 つ以上のホップが必要になることがあります。	`RELAY_HOPS=`*integer*

表 15–3 DHCP サーバーのパフォーマンスに影響するオプション (続き)

サーバーオプション	説明	キーワード
重複したアドレスを検出する	デフォルトでは、サーバーは IP アドレスをクライアントに提供する前にこのアドレスに ping します。ping に対する応答がなければ、このアドレスはまだ使用されていません。この機能を無効にすることによって、サーバーがクライアントにアドレスを提供するまでの時間を短縮できます。ただし、この機能を無効にすると、重複した IP アドレスを使用する危険があります。	`ICMP_VERIFY=TRUE/FALSE`
指定された間隔で `dhcptab` を自動的に再読み込む	サーバーは、指定された間隔 (分単位) で `dhcptab` を自動的に読み込むように設定できます。ネットワークの構成情報を頻繁に変更せず、複数の DHCP サーバーを持っていない場合は、`dhcptab` を自動的に再読み込みする必要はありません。また、DHCP マネージャには、データ変更後にサーバーに `dhcptab` を再読み込みさせるようにするオプションもあります。	`RESCAN_INTERVAL=`*min*
指定された時間、提供された IP アドレスをキャッシュする	サーバーが IP アドレスをクライアントに提供すると、その IP アドレスはキャッシュされます。提供内容がキャッシュされている間は、サーバーはそのアドレスを再び提供することはしません。提供内容がキャッシュに書き込まれている秒数を変更できます。デフォルトは 10 秒です。低速のネットワークでは、このキャッシュ時間を延長する必要があります。	`OFFER_CACHE_TIMEOUT=`*sec*

次の手順では、これらのオプションを変更する方法を説明します。

▼ DHCP パフォーマンスオプションをカスタマイズする方法 (DHCP マネージャ)

1. **DHCP マネージャでは、「サービス (Service)」メニューから「変更 (Modify)」を選択する**
   358 ページの「DHCP マネージャを起動および停止する方法」を参照してください。
2. **必要なオプションを変更します。**
   オプションについては、表 15–3 を参照してください。
3. **「サーバーの再起動」を選択します。**
4. **「了解」をクリックします。**

▼ DHCP パフォーマンスオプションをカスタマイズする方法 (コマンド行)

この手順で変更したオプションは、DHCP サーバーを再起動するまで有効にはなりません。

1 スーパーユーザーになるか、**DHCP 管理プロファイル**に割り当てられている役割またはユーザー名になります。

DHCP 管理プロファイルの詳細については、359 ページの「DHCP コマンドへのユーザーアクセスの設定」を参照してください。

役割には、認証と特権コマンドが含まれます。役割の詳細については、『Solaris のシステム管理: セキュリティーサービス』の「RBAC の構成 (タスクマップ)」を参照してください。

2 1 つまたは複数のパフォーマンスオプションを変更します。

```
# /usr/sbin/dhcpconfig -P keyword=value,keyword=value...
```

keyword=value には、次のキーワードが使用できます。

`RELAY_HOPS=`*integer*	デーモンが DHCP または BOOTP のデータグラムをドロップする前に発生することができるリレーエージェントホップの最大数を指定します。
`ICMP_VERIFY=TRUE/FALSE`	重複 IP アドレスの自動検出を有効または無効にします。このキーワードに `FALSE` を設定することはお勧めできません。
`RESCAN_INTERVAL=`*minutes*	DHCP サーバーが `dhcptab` 情報を自動的に読み込み直す間隔を分で指定します。
`OFFER_CACHE_TIMEOUT=`*seconds*	DHCP サーバーが DHCP クライアントを検索するために提供した内容をキャッシュに書き込んでおく秒数を指定します。デフォルトは 10 秒です。

例 15–1 DHCP パフォーマンスオプションの設定

次に、すべてのコマンドオプションを指定する方法の例を示します。

```
# dhcpconfig -P RELAY_HOPS=2,ICMP_VERIFY=TRUE,\
RESCAN_INTERVAL=30,OFFER_CACHE_TIMEOUT=20
```

DHCP ネットワークの追加、変更、削除 (作業マップ)

DHCP サーバーを構成する際に、DHCP サービスを使用するために少なくとも 1 つのネットワークを構成する必要があります。いつでもネットワークを追加できます。

次の表のマップでは、初期構成を完了した DHCP ネットワークを管理するときに実行できる追加の作業について説明しています。このタスクマップには、タスクの実施に必要な手順へのリンクが含まれています。

タスク	説明	参照先
サーバーネットワークインタフェースでの DHCP サービスを有効または無効にします	デフォルトの動作では、DHCP 要求に対するすべてのネットワークインタフェースを監視します。DHCP 要求をすべてのインタフェースで受け付けない場合は、監視するインタフェースのリストからインタフェースを削除できます。	380 ページの「DHCP 監視用のネットワークインタフェースを指定する方法 (DHCP マネージャ)」
DHCP サービスに新しいネットワークを追加します。	ネットワーク上の IP アドレスを管理するため、ネットワークを DHCP の管理下に置きます。	382 ページの「DHCP ネットワークを追加する方法 (DHCP マネージャ)」 383 ページの「DHCP ネットワークを追加する方法 (dhcpconfig)」
DHCP に管理されたネットワークのパラメータを変更します。	特定のネットワークのクライアントに渡される情報を変更します。	384 ページの「DHCP ネットワークの構成を変更する方法 (DHCP マネージャ)」 385 ページの「DHCP ネットワークの構成を変更する方法 (dhtadm)」
DHCP サービスからネットワークを削除します。	これ以降、ネットワーク上の IP アドレスが DHCP によって管理されないようにネットワークを削除します。	387 ページの「DHCP ネットワークを削除する方法 (DHCP マネージャ)」 387 ページの「DHCP ネットワークを削除する方法 (pntadm)」

DHCP で監視するネットワークインタフェースの指定

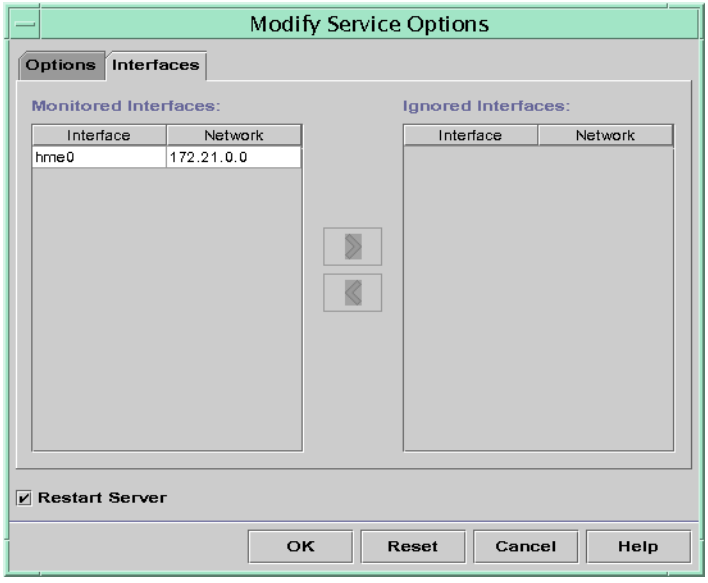
デフォルトでは、`dhcpconfig` も DHCP マネージャの構成ウィザードも、サーバーシステムのすべてのネットワークインタフェースを監視するように DHCP サーバーを構成します。新しいネットワークインタフェースをサーバーシステムに追加した場合、システムをブートすると、DHCP サーバーがこの新しいネットワークインタフェースを自動的に監視します。そのため、どのネットワークを追加してもそのネットワークインタフェースを通して監視できます。

ただし、どのネットワークインタフェースを監視し、どのインタフェースを無視すべきかを指定することもできます。特定のネットワーク上で DHCP サービスを提供したくない場合、インタフェースを無視すると便利なことがあります。

どのインタフェースも無視するように設定したあとに新しいインタフェースをインストールすると、DHCP サーバーは新しいインタフェースを無視します。その場合には、サーバーの監視インタフェーリストに新しいインタフェースを追加する必要があります。インタフェースを指定するには、DHCP マネージャまたは `dhcpconfig` ユーティリティーを使用できます。

この節では、DHCP が監視または無視するネットワークインタフェースを指定できるようにするための手順についても説明します。この DHCP マネージャの手順では、DHCP マネージャの「サービスオプションの変更 (Modify Service Options)」ダイアログボックスの「インタフェース (Interfaces)」タブを使用します (次図を参照)。

図 15–4 DHCP マネージャの「サービスオプションの変更 (Modify Service Options)」ダイアログボックスの「インタフェース (Interfaces)」タブ



▼ DHCP 監視用のネットワークインタフェースを指定する方法 (DHCP マネージャ)

1. DHCP マネージャでは、「サービス (Service)」メニューから「変更 (Modify)」を選択する

   「サービスオプションの変更 (Modify Service Options)」ダイアログボックスが表示されます。

   358 ページの「DHCP マネージャを起動および停止する方法」を参照してください。

2. 「インタフェース (Interfaces)」タブを選択します。

3. 適切なネットワークインタフェースを選択します。

4. 矢印ボタンをクリックして、インタフェースを適切なリストに移動します。

   たとえば、インタフェースを無視する場合は、「監視中のインタフェース (Monitored Interfaces)」リストからそのインタフェースを選択し、右矢印ボタンをクリックします。インタフェースは、「削除するインタフェース (Ignored Interfaces)」リストに表示されます。

5. 「サーバーの再起動 (Restart Server)」を選択し、「了解 (OK)」をクリックします。

   これらの変更は、リブートが行われたあとも有効です。

▼ DHCP 監視用のネットワークインタフェースを指定する方法 (dhcpconfig)

1. スーパーユーザーになるか、DHCP 管理プロファイルに割り当てられている役割またはユーザー名になります。

   DHCP 管理プロファイルの詳細については、359 ページの「DHCP コマンドへのユーザーアクセスの設定」を参照してください。

   役割には、認証と特権コマンドが含まれます。役割の詳細については、『Solaris のシステム管理: セキュリティーサービス』の「RBAC の構成 (タスクマップ)」を参照してください。

2. DHCP サーバーシステム上で次のコマンドを入力します。

   ```
   # /usr/sbin/dhcpconfig -P INTERFACES=int,int,...
   ```

   int, int,... は、監視するインタフェースのリストです。インタフェース名をコンマで区切って指定します。

たとえば、ge0 と ge1 だけを監視する場合は、次のコマンドを使用します。

```
#/usr/sbin/dhcpconfig -P INTERFACES=ge0,ge1
```

無視するインタフェースは、dhcpconfig コマンド行から除外します。

このコマンドによる変更はリブート後も引き続き有効です。

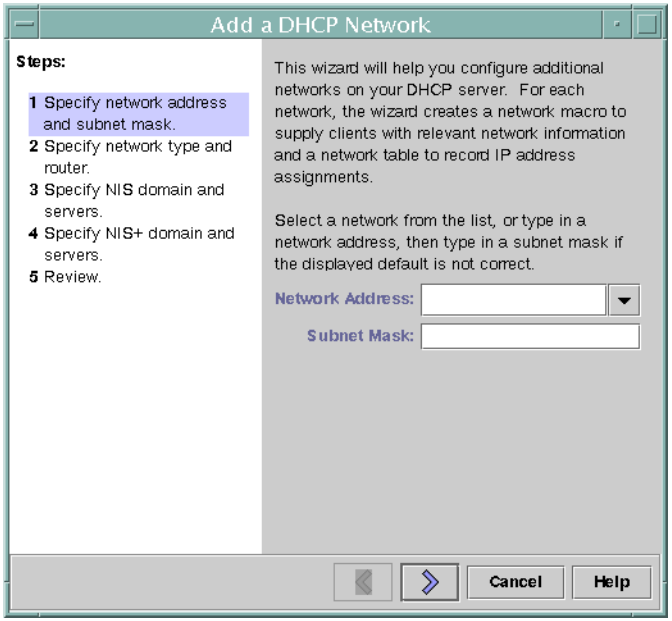
DHCP ネットワークの追加

DHCP マネージャを使用してサーバーを構成すると、最初のネットワークも同時に構成されます。最初のネットワークは通常、サーバーシステムの主インタフェースにあるローカルネットワークです。さらにほかのネットワークを構成したい場合は、DHCP マネージャの DHCP ネットワークウィザードを使用します。

dhcpconfig -D コマンドを使用してサーバーを構成する場合は、DHCP サービスを使用するすべてのネットワークを別個に構成する必要があります。詳細は、383 ページの「DHCP ネットワークを追加する方法 (dhcpconfig)」を参照してください。

次の図は、DHCP マネージャの DHCP ネットワークウィザード用の最初のダイアログボックスを示しています。

図 15–5　DHCP マネージャのネットワークウィザード



新しいネットワークを構成すると、DHCP マネージャが次の内容を作成します。

- データストアにネットワークテーブルを作成します。新しいネットワークは、DHCP マネージャの「アドレス (Addresses)」タブにあるネットワークリストに表示されます。
- このネットワークに常駐するクライアントで必要とする情報を含むネットワークマクロを作成します。このネットワークマクロの名前はネットワークの IP アドレスと一致します。ネットワークマクロはデータストア内の `dhcptab` テーブルに追加されます。

▼ DHCP ネットワークを追加する方法 (DHCP マネージャ)

1. **DHCP マネージャの「アドレス (Addresses)」タブをクリックします。**
   すでに DHCP サービス用に構成されているネットワークがリストされます。
   358 ページの「DHCP マネージャを起動および停止する方法」を参照してください。

2. **「編集 (Edit)」メニューから「ネットワークウィザード (Network Wizard)」を選択します。**

3. **オプションを選択するか、要求された情報を入力します。計画フェーズで行なった決定に基づいて、指定すべき情報を決めます。**
   計画フェーズについては、339 ページの「リモートネットワークの DHCP 構成の計画」を参照してください。

   このウィザードでわからないことがある場合は、ウィザードウィンドウの「ヘルプ (Help)」をクリックすると、DHCP ネットワークウィザードのヘルプが表示されます。

4. **必要な情報を入力し終えたあと、「完了 (Finish)」をクリックしてネットワークの構成を終了します。**
   ネットワークウィザードは、空のネットワークテーブルを作成します。このテーブルはウィンドウの左側のペインに表示されます。

   さらに、ネットワークウィザードは、そのネットワークの IP アドレスと一致する名前のネットワークマクロを作成します。

5. **(省略可能) 「マクロ (Macros)」タブを選択し、作成したネットワークマクロを選択して、マクロの内容を表示します。**
   ウィザードで指定した情報が、ネットワークマクロのオプションの値として挿入されているか確認します。

参照　このネットワークのアドレスを追加してから、そのネットワークの IP アドレスを DHCP で管理する必要があります。詳細は、395 ページの「DHCP サービスへの IP アドレスの追加」を参照してください。

ネットワークテーブルが空であっても、DHCP サーバーは、構成情報をクライアントに提供できます。詳細は、431 ページの「情報だけを受け取るように DHCP クライアントを設定 (タスクマップ)」を参照してください。

▼ DHCP ネットワークを追加する方法 (dhcpconfig)

1. スーパーユーザーになるか、**DHCP** 管理プロファイルに割り当てられている役割またはユーザー名になります。

   DHCP 管理プロファイルの詳細については、359 ページの「DHCP コマンドへのユーザーアクセスの設定」を参照してください。

   役割には、認証と特権コマンドが含まれます。役割の詳細については、『Solaris のシステム管理: セキュリティーサービス』の「RBAC の構成 (タスクマップ)」を参照してください。

2. **DHCP** サーバーシステム上で次のコマンドを入力します。

   ```
   # /usr/sbin/dhcpconfig -N network-address
   ```

   network-address には、DHCP サービスに追加するネットワークの IP アドレスを指定します。`-N` オプションで使用できるサブオプションについては、`dhcpconfig(1M)` のマニュアルページを参照してください。

   サブオプションを使用しない場合、`dhcpconfig` はネットワークファイルを使用して、ネットワークについての情報を取得します。

参照　このネットワークのアドレスを追加してから、そのネットワークの IP アドレスを DHCP で管理する必要があります。詳細は、395 ページの「DHCP サービスへの IP アドレスの追加」を参照してください。

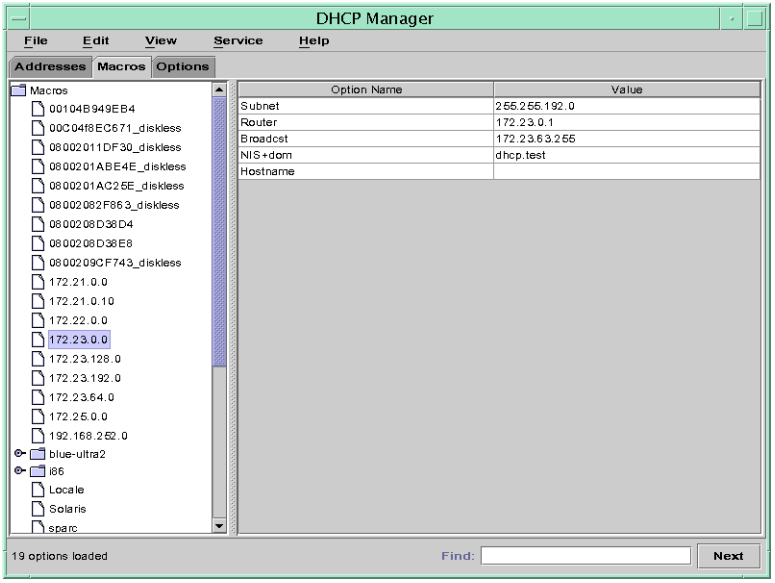
ネットワークテーブルが空であっても、DHCP サーバーは、構成情報をクライアントに提供できます。詳細は、431 ページの「情報だけを受け取るように DHCP クライアントを設定 (タスクマップ)」を参照してください。

DHCP ネットワークの構成の変更

ネットワークを DHCP サービスに追加したら、そのあとで、最初に入力した構成情報を変更できます。構成情報は、ネットワーク上のクライアントに情報を渡すためのネットワークマクロに格納されています。ネットワーク構成を変更するためには、ネットワークマクロを変更する必要があります。

次に、DHCP マネージャの「マクロ (Macros)」タブを示します。

図 15–6　DHCP マネージャの「マクロ (Macros)」タブ



▼ DHCP ネットワークの構成を変更する方法 (DHCP マネージャ)

1. **DHCP マネージャの「マクロ (Macros)」タブを選択します。**
   この DHCP サーバーについて定義されたすべてのマクロが左側のペインにリストされます。

   358 ページの「DHCP マネージャを起動および停止する方法」を参照してください。

2. **変更するネットワーク構成と名前が一致するネットワークマクロを選択します。**
   ネットワークマクロ名は、そのネットワークの IP アドレスです。

3. **「編集 (Edit)」メニューから「属性 (Properties)」を選択します。**
   「マクロの属性 (Macro Properties)」ダイアログボックスに、マクロに含まれるオプションが示されます。

4 変更するオプションを選択します。
オプションの名前とその値は、ダイアログボックス上部のテキストフィールドに表示されます。

5 (省略可能) オプション名を変更するか、「選択 (Select)」ボタンをクリックしてオプション名のリストを表示します。
「(Select Option)」ダイアログボックスに、DHCP のすべての標準オプションが表示されます。各オプションには簡単な説明がついています。

6 (省略可能) 「(Select Option)」ダイアログボックスから 1 つのオプション名を選択し、「了解 (OK)」をクリックします。
新しいオプション名が 「(Option Name)」フィールドに表示されます。

7 そのオプションの新しい値を入力して、「変更 (Modify)」をクリックします。

8 (省略可能) さらに、ダイアログボックスの「選択 (Select)」を使用することによって、ネットワークマクロにオプションを追加できます。
マクロの変更に関するより一般的な情報については、410 ページの「DHCP マクロの変更」を参照してください。

9 「DHCP サーバーに変更を通知する (Notify DHCP Server of Change)」を選択して、「了解 (OK)」をクリックします。
この選択によって、DHCP サーバーは `dhcptab` テーブルを再読み込みし、「了解 (OK)」をクリックすると直ちに変更が適用されます。

▼ DHCP ネットワークの構成を変更する方法 (dhtadm)

1 スーパーユーザーになるか、DHCP 管理プロファイルに割り当てられている役割またはユーザー名になります。
DHCP 管理プロファイルの詳細については、359 ページの「DHCP コマンドへのユーザーアクセスの設定」を参照してください。

役割には、認証と特権コマンドが含まれます。役割の詳細については、『Solaris のシステム管理: セキュリティーサービス』の「RBAC の構成 (タスクマップ)」を参照してください。

2 ネットワークのすべてのクライアントに関する情報を含むマクロを特定します。
ネットワークマクロの名前は、ネットワークの IP アドレスと一致します。

この情報がどのマクロに含まれているがわからない場合、`dhtadm -P` コマンドを使用すると、`dhcptab` テーブルを表示して、すべてのマクロを表示できます。

3 次の書式でコマンドを入力して、変更したいオプションの値を変更します。

```
# dhtadm -M -m macro-name -e 'symbol=value' -g
```

dhtadm コマンド行オプションについては、dhtadm(1M) のマニュアルページを参照してください。

例 15–2 dhtadm コマンドによる DHCP マクロの変更

たとえば、10.25.62.0 のマクロのリース期間を 57600 秒に変更し、NIS ドメインを sem.example.com に変更するには、次のコマンドを入力します。

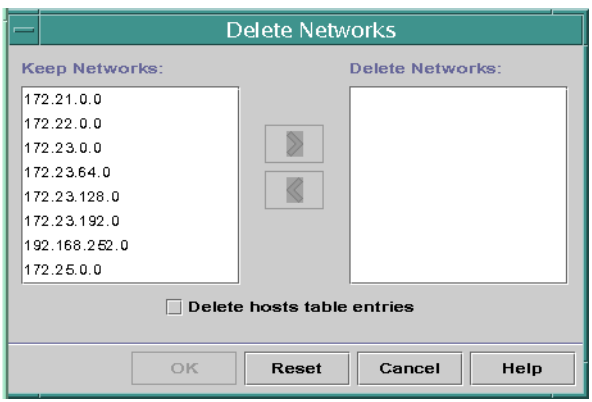
```
# dhtadm -M -m 10.25.62.0 -e 'LeaseTim=57600' -g

# dhtadm -M -m 10.25.62.0 -e 'NISdmain=sem.example.com' -g
```

-g オプションを指定すると、DHCP デーモンは dhcptab テーブルを再読み込みして、変更を有効にします。

DHCP ネットワークの削除

DHCP マネージャを使用すると、複数のネットワークを同時に削除できます。削除するネットワークにある DHCP に管理された IP アドレスに関連するホストテーブルのエントリを自動的に削除するオプションもあります。次に、DHCP マネージャの「ネットワークの削除 (Delete Networks)」ダイアログボックスを示します。

図 15–7 DHCP マネージャの「ネットワークの削除 (Delete Networks)」ダイアログボックス



pntadm コマンドを使用する場合、ネットワークからそれぞれの IP アドレスのエントリを削除してからそのネットワークを削除する必要があります。一度に 1 つのネットワークだけを削除できます。

▼ DHCP ネットワークを削除する方法 (DHCP マネージャ)

1. DHCP マネージャの「アドレス (Addresses)」タブをクリックします。
   358 ページの「DHCP マネージャを起動および停止する方法」を参照してください。
2. 「編集 (Edit)」メニューから「ネットワークの削除 (Delete Networks)」を選択します。
   「ネットワークの削除 (Delete Networks)」ダイアログボックスが開きます。
3. 「保持するネットワーク (Keep Networks)」リストで、削除したいネットワークを選択します。
   Control キーを押しながらマウスをクリックすれば、複数のネットワークを選択できます。Shift キーを押しながらクリックすれば、ある範囲のネットワークを選択できます。
4. 右矢印ボタンをクリックして、選択したネットワークを「ネットワークの削除 (Delete Networks)」リストに移動します。
5. このネットワークの DHCP アドレスに対するホストテーブルエントリを削除する場合は、「ホストテーブルエントリも削除 (Delete Host Table Entries)」を選択します。
   ホストテーブルエントリを削除しただけでは、これらのアドレスに関する DNS サーバー上のホスト登録は削除されません。エントリは、ローカルネームサービスからのみ削除されます。
6. 「了解」をクリックします。

▼ DHCP ネットワークを削除する方法 (pntadm)

この手順では、ネットワークを削除する前に DHCP ネットワークテーブルからネットワークの IP アドレスを削除します。これらのアドレスを削除することによって、そのホスト名が `hosts` ファイルまたはデータベースから確実に削除されます。

1. スーパーユーザーになるか、DHCP 管理プロファイルに割り当てられている役割またはユーザー名になります。
   DHCP 管理プロファイルの詳細については、359 ページの「DHCP コマンドへのユーザーアクセスの設定」を参照してください。

   役割には、認証と特権コマンドが含まれます。役割の詳細については、『Solaris のシステム管理: セキュリティーサービス』の「RBAC の構成 (タスクマップ)」を参照してください。

2 次の書式でコマンドを入力して、ネームサービスから **IP** アドレスとそのホスト名を削除します。

```
# pntadm -D -y IP-address
```

たとえば、IP アドレス `10.25.52.1` を削除するには、次のコマンドを入力します。

```
# pntadm -D -y 10.25.52.1
```

この `-y` オプションは、ホスト名の削除を指定します。

3 ネットワークのアドレスごとに **pntadm -D -y** コマンドを繰り返し入力します。
多数のアドレスを削除する場合には、`pntadm` コマンドを実行するスクリプトを作成することもできます。

4 すべてのアドレスを削除してから、次のコマンドを入力して、**DHCP** サービスからネットワークを削除します。

```
# pntadm -R network-IP-address
```

たとえば、ネットワーク `10.25.52.0` を削除するには、次のコマンドを入力します。

```
# pntadm -R 10.25.52.0
```

`pntadm` ユーティリティーの使用についての詳細は、`pntadm(1M)` のマニュアルページを参照してください。

DHCP サービスによる BOOTP クライアントのサポート (タスクマップ)

DHCP サーバー上で BOOTP クライアントをサポートするには、DHCP サーバーを BOOTP 互換に設定する必要があります。この DHCP を使用できる BOOTP クライアントを指定したい場合は、BOOTP クライアントを DHCP サーバーのネットワークテーブルに登録できます。あるいは、その代わりに、BOOTP クライアントに自動的に割り当てるいくつかの IP アドレスを予約しておくこともできます。

注 – BOOTP アドレスは常時割り当てられます。アドレスに常時リースを明示的に割り当てたかどうかは関係ありません。

次の表では、BOOTP クライアントをサポートするために実行する必要があるタスクについて説明します。タスクマップには、タスクの実行に必要な手順へのリンクが含まれています。

タスク	説明	参照先
自動 BOOTP サポートを設定します。	DHCP に管理されたネットワークや、リレーエージェントによって DHCP に管理されたネットワークに接続されたネットワークにあるすべての BOOTP クライアントに IP アドレスを提供します。 そのため、BOOTP クライアントでアドレスを排他的に使用するためにアドレスのプールを予約する必要があります。このオプションは、サーバーが多くの BOOTP クライアントをサポートする必要がある場合に便利です。	389 ページの「すべての BOOTP クライアントのサポートを設定する方法 (DHCP マネージャ)」
手動 BOOTP サポートを設定します。	DHCP サービスを使用して手動で登録された BOOTP クライアントだけに IP アドレスを提供します。 このオプションでは、BOOTP クライアント用に指定された特定の IP アドレスにクライアントの ID を結びつける必要があります。このオプションは、BOOTP クライアントが少数の場合や、DHCP サーバーを使用できる BOOTP クライアントの数を制限する場合に便利です。	390 ページの「登録された BOOTP クライアントのサポートを設定する方法 (DHCP マネージャ)」

▼ すべての **BOOTP** クライアントのサポートを設定する方法 **(DHCP** マネージャ**)**

1. **DHCP** マネージャでは、「サービス **(Service)**」メニューから「変更 **(Modify)**」を選択する

   「サービスオプションの変更 (Modify Service Options)」ダイアログボックスが開きます。

   358 ページの「DHCP マネージャを起動および停止する方法」を参照してください。

2. このダイアログボックスの「**BOOTP** 互換 **(BOOTP Compatibility)**」セクションで、「自動 **(Automatic)**」を選択します。

3. 「サーバーの再起動 **(Restart Server)**」を選択し、「了解 **(OK)**」をクリックします。

4. 「アドレス **(Addresses)**」タブを選択します。

5 **BOOTP** クライアント用に予約したいアドレスを選択します。
最初のアドレスをクリックし、Shift キーを押しながら最後のアドレスをクリックして、一定範囲のアドレスを選択します。Control キーを押しながら各アドレスをクリックして、重複していない複数のアドレスを選択します。

6 「編集 **(Edit)**」メニューから「属性 **(Properties)**」を選択します。
「複数アドレスの変更 (Modify Multiple Addresses)」ダイアログボックスが開きます。

7 「**BootP**」セクションで、「**BootP** クライアントだけにすべてのアドレスを割り当てる **(Assign All Addresses Only to BOOTP Clients)**」を選択します。
残りのオプションは「現在の設定を維持 (Keep Current Settings)」に設定しておきます。

8 「了解」をクリックします。
これで、すべての BOOTP クライアントがこの DHCP サーバーからアドレスを取得できるようになりました。

▼ 登録された **BOOTP** クライアントのサポートを設定する方法 **(DHCP** マネージャ**)**

1 **DHCP** マネージャでは、「サービス **(Service)**」メニューから「変更 **(Modify)**」を選択する
「サービスオプションの変更 (Modify Service Options)」ダイアログボックスが開きます。

358 ページの「DHCP マネージャを起動および停止する方法」を参照してください。

2 このダイアログボックスの「**BOOTP** 互換 **(BOOTP Compatibility)**」セクションで、「手動 **(Manual)**」を選択します。

3 「サーバーの再起動 **(Restart Server)**」を選択し、「了解 **(OK)**」をクリックします。

4 「アドレス **(Addresses)**」タブを選択します。

5 特定の **BOOTP** クライアントに割り当てるアドレスを選択します。

6 「編集 **(Edit)**」メニューから「属性 **(Properties)**」を選択します。
「アドレスの属性 (Address Properties)」ダイアログボックスが開きます。

7 「アドレス属性 **(Address Properties)**」ダイアログボックスの「リース **(Lease)**」タブを選択します。

8 「クライアント ID (Client ID)」フィールドでクライアントの ID を入力します。

Ethernet ネットワークで動作する BOOTP Oracle Solaris クライアントの場合、クライアント ID は、クライアントの 16 進数 Ethernet アドレスから派生した文字列になります。クライアント ID には、Ethernet 用の Address Resolution Protocol (ARP) タイプ (01) を示す接頭辞が含まれています。たとえば、Ethernet アドレス `8:0:20:94:12:1e` を持つ BOOTP クライアントは、`0108002094121E` というクライアント ID を使用します。

ヒント – Oracle Solaris クライアントシステム上のスーパーユーザーとして次のコマンドを入力すると、そのインタフェースに関する Ethernet アドレスを取得できます。

```
# ifconfig -a
```

---

9 「予約 (Reserved)」を選択して、このクライアント用に IP アドレスを予約します。

10 「BOOTP クライアントのみに割り当てる (Assign Only to BOOTP Clients)」を選択し、「了解 (OK)」をクリックします。

「アドレス (Addresses)」タブでは、BOOTP は「ステータス (Status)」フィールドに表示され、指定したクライアント ID は「クライアント ID (Client ID)」フィールドに表示されます。

DHCP サービスで IP アドレスを使用して作業する (作業マップ)

IP アドレスの追加、アドレス属性の変更、DHCP サービスからのアドレスの削除を実行するには、DHCP マネージャまたは `pntadm` コマンドを使用できます。IP アドレスを使用して作業する前に、表 15–4 を参照して、IP アドレスの属性をよく理解してください。この表を使用して、DHCP マネージャと `pntadm` を使用するための情報を知ることができます。

注 – 表 15–4 には、`pntadm` を使って IP アドレスの追加や変更をしながら IP アドレスの属性を指定する例が含まれています。`pntadm` の詳細については、`pntadm(1M)` のマニュアルページも参照してください。

次の作業マップに、IP アドレスを追加、変更、または削除するときに必要な作業を示します。さらに、タスクマップには、タスクの実行に必要な手順へのリンクが含まれています。

タスク	説明	参照先
単一または複数の IP アドレスを DHCP サービスに追加します。	DHCP マネージャを使用して DHCP サービスですでに管理されているネットワークに IP アドレスを追加します。	397 ページの「単一の IP アドレスを追加する方法 (DHCP マネージャ)」 398 ページの「既存の IP アドレスを複製する方法 (DHCP マネージャ)」 398 ページの「複数の IP アドレスを追加する方法 (DHCP マネージャ)」 399 ページの「IP アドレスを追加する方法 (pntadm)」
IP アドレスの属性を変更します。	表 15–4 に示す IP アドレスの属性を変更します。	401 ページの「IPアドレスの属性を変更する方法 (DHCP マネージャ)」 401 ページの「IP アドレスの属性を変更する方法 (pntadm)」
DHCP サービスから IP アドレスを削除します。	指定された IP アドレスを DHCP から使用できないように設定します。	402 ページの「IP アドレスを使用不可に指定する方法 (DHCP マネージャ)」 403 ページの「IP アドレスを使用不可に指定する方法 (pntadm)」 404 ページの「DHCP サービスから IP アドレスを削除する方法 (DHCP マネージャ)」 405 ページの「DHCP サービスから IP アドレスを削除する方法 (pntadm)」
固定 IP アドレスを DHCP クライアントに割り当てます。	クライアントがその構成を要求するたびに同じ IP アドレスを受け取るようにクライアントを設定します。	406 ページの「固定 IP アドレスを DHCP クライアントに割り当てる方法 (DHCP マネージャ)」 407 ページの「固定 IP アドレスを DHCP クライアントに割り当てる方法 (pntadm)」

次の表に、IP アドレスの属性の一覧および説明を示します。

表 15–4 IP アドレスの属性

属性	説明	pntadm コマンドで指定する方法
ネットワークアドレス	作業の際に使用する IP アドレスを含むネットワークのアドレス。 このネットワークアドレスは、DHCP マネージャのアドレスタブにあるネットワークリストに表示されます。	ネットワークアドレスは、IP アドレスを作成、変更、または削除するために使用する pntadm コマンド行の最後の引数にする必要があります。 たとえば、IP アドレスをネットワーク 10.21.0.0 に追加する場合には、次のように入力します。 **pntadm -A** *ip-address options* **10.21.0.0**
IP アドレス	作成、変更、または削除する、作業中のアドレス。 この IP アドレスは、DHCP マネージャのアドレスタブの最初の列に表示されます。	IP アドレスを操作する場合、pntadm コマンドに必ず -A、-M、または -D オプションを付けます。 たとえば、IP アドレス 10.21.5.12 を変更する場合には、次のように入力します。 **pntadm -M 10.21.5.12** *options* **10.21.0.0**
クライアント名	ホストテーブルで IP アドレスに割り当てられるホスト名。この名前は、アドレスが作成されるときに、DHCP マネージャによって自動的に生成できます。単一のアドレスを作成する場合、その名前を入力できます。	-h オプションを使用してクライアント名を指定します。 たとえば、10.21.5.12 に対してクライアント名 carrot12 を指定する場合には、次のように入力します。 **pntadm -M 10.21.5.12 -h carrot12 10.21.0.0**
サーバーによる所有	IP アドレスを管理し、DHCP クライアントの IP アドレス割り当て要求に応答する DHCP サーバー。	-s オプションを使用して所有サーバー名を指定します。 たとえば、サーバー blue2 が 10.21.5.12 を所有することを指定するには、次のように入力します。 **pntadm -M 10.21.5.12 -s blue2 10.21.0.0**
構成マクロ	dhcptab テーブルからネットワーク構成オプションを取得するために DHCP サーバーが使用するマクロ。サーバーを構成したり、ネットワークを追加したりすると、いくつかのマクロが自動的に作成されます。マクロについては、322 ページの「DHCP マクロについて」を参照してください。アドレスを作成すると、サーバーマクロが同時に作成されます。サーバーマクロは、各アドレスの構成マクロとして割り当てられます。	-m オプションを使用してマクロ名を指定します。 たとえば、サーバーマクロ blue2 をアドレス 10.21.5.12 に割り当てる場合には、次のように入力します。 **pntadm -M 10.21.5.12 -m blue2 10.21.0.0**

表 15–4 IP アドレスの属性 (続き)

属性	説明	pntadm コマンドで指定する方法
クライアント ID	DHCP サービス内で一意のテキスト文字列。 クライアント ID が 00 の場合、アドレスはどのクライアントにも割り当てられていません。IP アドレスの属性を変更する際にクライアント ID を指定すると、そのアドレスはそのクライアントに排他的に結合されます。 クライアント ID は、DHCP クライアントのベンダーによって決定されます。DHCP クライアント以外のクライアントを使用している場合、そのクライアントのドキュメントで詳細を確認してください。 DHCP クライアントの場合、クライアント ID は、クライアントの 16 進数ハードウェアアドレスから派生したものになります。クライアント ID には、ネットワークタイプの ARP コード (たとえば、Ethernet の場合 01) を表す接頭辞が含まれます。ARP コードは、Internet Assigned Numbers Authority (IANA) (Assigned Numbers 標準の ARP Parameters セクション) によって割り当てられます (`http://www.iana.com/numbers.html` を参照)。 たとえば、16 進 Ethernet アドレス `8:0:20:94:12:1e` を持つ Oracle Solaris クライアントは、クライアント ID `0108002094121E` を使用します。クライアントがアドレスを使用している場合、このクライアント ID は DHCP マネージャと `pntadm` で示されます。 ヒント: Oracle Solaris クライアントシステム上のスーパーユーザーとして次のコマンドを入力すると、そのインタフェースの Ethernet アドレスを取得できます。`ifconfig -a`	`-i` オプションを使用してクライアント ID を指定します。 たとえば、クライアント ID `08002094121E` をアドレス `10.21.5.12` に割り当てる場合には、次のように入力します。 **`pntadm -M 10.21.5.12 -i 0108002094121E 10.21.0.0`**

表 15–4 IP アドレスの属性 (続き)

属性	説明	pntadm コマンドで指定する方法
予約済み	このアドレスを指定した設定は、クライアント ID で示されるクライアントのために排他的に予約されます。DHCP サーバーがこのアドレスを取り戻すことはできません。このオプションを選択した場合、アドレスはクライアントに手動で割り当てます。	-f オプションを使用して、アドレスの予約または手動を指定します。 たとえば、あるクライアントについて IP アドレス 10.21.5.12 の予約を指定するには、次のように入力します。 **pntadm -M 10.21.5.12 -f MANUAL 10.21.0.0**
リースのタイプとポリシー	クライアントによる IP アドレスの使用を DHCP が管理する方法を示す設定。リースは、動的または常時です。詳細な説明は、336 ページの「動的リースタイプと常時リースタイプ」を参照してください。	-f オプションを使用して、アドレスが常時割り当てされるように指定します。デフォルトではアドレスは動的にリースされます。 たとえば、IP アドレス 10.21.5.12 に常時リースを割り当てるには、次のように入力します。 **pntadm -M 10.21.5.12 -f PERMANENT 10.21.0.0**
リースの有効期限	リースが期限切れになる日付。動的リースが指定された場合のみ利用できます。日付は *mm/dd/yyyy* 書式で指定します。	リースの有効期限を -e オプションで指定します。 たとえば、有効期限として 2006 年 1 月 1 日を指定する場合には、次のように入力します。 **pntadm -M 10.21.5.12 -e 01/01/2006 10.21.0.0**
BOOTP 設定	BOOTP クライアントに対してアドレスが予約されていることを指定します。BOOTP クライアントのサポートについては、388 ページの「DHCP サービスによる BOOTP クライアントのサポート (タスクマップ)」を参照してください。	-f を使用して BOOTP クライアント用のアドレスを予約します。 たとえば、IP アドレス 10.21.5.12 を BOOTP クライアント用に予約する場合には、次のように入力します。 **pntadm -M 10.21.5.12 -f BOOTP 10.21.0.0**
使用不可設定	どのクライアントにも割り当てられないようにアドレスに使用不可のマークを付ける設定。	-f オプションを使ってアドレスを使用不可にします。 たとえば、IP アドレス 10.21.5.12 を使用不可にするには、次のように入力します。 **pntadm -M 10.21.5.12 -f UNUSABLE 10.21.0.0**

DHCP サービスへの IP アドレスの追加

IP アドレスを追加する前に、それらのアドレスを所有するネットワークを DHCP サービスに追加する必要があります。ネットワークの追加については、381 ページの「DHCP ネットワークの追加」を参照してください。

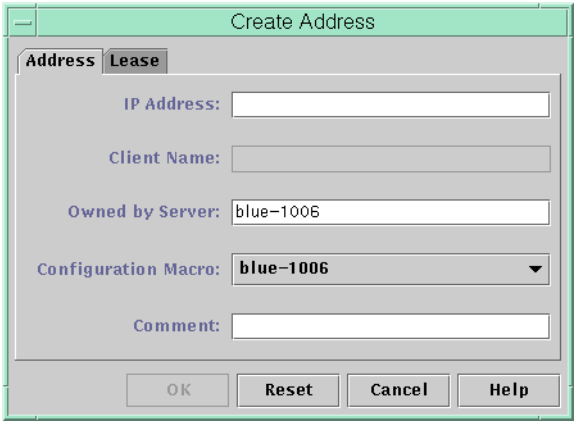
アドレスの追加は、DHCP マネージャまたは pntadm コマンドで行うことができます。

すでに DHCP サービスによって管理されているネットワーク上では、DHCP マネージャを使用すると、次のような複数の方法でアドレスを追加できます。

- 単一の **IP** アドレスの追加 – 単一の新しい IP アドレスを DHCP の管理下に置きます。
- 既存の **IP** アドレスの複製 – DHCP が管理する既存の IP アドレスの属性をコピーし、新しい IP アドレスとクライアント名を与えます。
- 一定範囲の複数の **IP** アドレスの追加 – アドレスウィザードを使用して、一連の IP アドレスを DHCP の管理下に置きます。

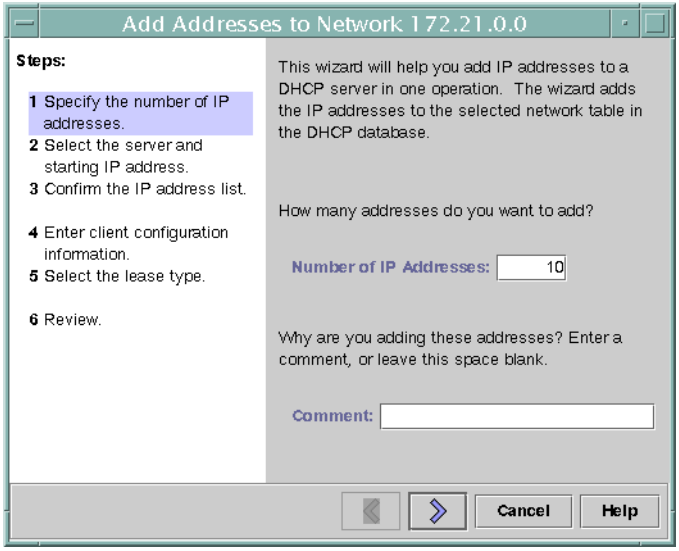
次に、「アドレスの作成 (Create Address)」ダイアログボックスを示します。「アドレスの複製 (Duplicate Address)」ダイアログボックスは、テキストフィールドに既存のアドレスの値が表示されていることを除いて「アドレスの作成 (Create Address)」ダイアログボックスと同じです。

図 15–8 DHCP マネージャの「アドレスの作成 (Create Address)」ダイアログボックス



次の図に、一定範囲の IP アドレスの追加に使用する「ネットワークへアドレスの追加 (Add Addresses to Network)」ウィザードの最初のダイアログを示します。

図 15–9　DHCP マネージャの「ネットワークへアドレスの追加 (Add Addresses to Network)」ウィザード



▼ 単一の IP アドレスを追加する方法 (DHCP マネージャ)

1. **DHCP** マネージャの「アドレス **(Addresses)**」タブをクリックします。
   358 ページの「DHCP マネージャを起動および停止する方法」を参照してください。
2. 新しい **IP** アドレスを追加するネットワークを選択します。
3. 「編集 **(Edit)**」メニューから「作成 **(Create)**」を選択します。
   「アドレスの作成 (Create Address)」ダイアログボックスが開きます。
4. 「アドレス **(Address)**」と「リース **(Lease)**」タブで、値を選択または入力します。
   「ヘルプ (Help)」ボタンを選択して Web ブラウザを開き、ダイアログボックスのヘルプを表示します。また、設定については、表 15–4 を参照してください。
5. 「了解」をクリックします。

▼ 既存の IP アドレスを複製する方法 (DHCP マネージャ)

1. **DHCP** マネージャの「アドレス **(Addresses)**」タブをクリックします。
   358 ページの「DHCP マネージャを起動および停止する方法」を参照してください。
2. 新しい **IP** アドレスを配置するネットワークを選択します。
3. 複製したい属性をもつアドレスを選択します。
4. 「編集 **(Edit)**」メニューから「複製 **(Duplicate)**」を選択します。
5. 「**IP** アドレス **(IP Address)**」フィールドに新しい **IP** アドレスを指定します。
6. **(省略可能)** そのアドレスの新しいクライアント名を指定します。
   複製するアドレスと同じ名前のクライアント名を使用することはできません。
7. **(省略可能)** 必要に応じて、ほかのオプションの値を変更します。
   ほとんどのオプションの値は変更の必要がないはずです。
8. 「了解」をクリックします。

▼ 複数の IP アドレスを追加する方法 (DHCP マネージャ)

1. **DHCP** マネージャの「アドレス **(Addresses)**」タブをクリックします。
   358 ページの「DHCP マネージャを起動および停止する方法」を参照してください。
2. 新しい **IP** アドレスを追加するネットワークを選択します。
3. 「編集 **(Edit)**」メニューから「アドレスウィザード **(Address Wizard)**」を選択します。
   「ネットワークへアドレスの追加 (Add Addresses to Network)」ダイアログボックスに IP アドレス属性の値を指定する必要があります。属性については、表 15–4 を参照するか、ダイアログボックスの「ヘルプ」ボタンをクリックしてください。334 ページの「IP アドレスの管理に必要な選択 (タスクマップ)」 には、さらに詳しい情報が記載されています。
4. 情報を入力し終わったら、画面ごとに右矢印ボタンをクリックし、最後の画面で「完了 **(Finish)**」をクリックします。
   「アドレス (Addresses)」 タブに新規アドレスが更新されます。

▼ IP アドレスを追加する方法 (pntadm)

1 スーパーユーザーになるか、**DHCP** 管理プロファイルに割り当てられている役割またはユーザー名になります。

DHCP 管理プロファイルの詳細については、359 ページの「DHCP コマンドへのユーザーアクセスの設定」を参照してください。

役割には、認証と特権コマンドが含まれます。役割の詳細については、『Solaris のシステム管理: セキュリティーサービス』の「RBAC の構成 (タスクマップ)」を参照してください。

2 次の書式のコマンドを入力して **IP** アドレスを追加します。

```
# pntadm -A ip-address options network-address
```

`pntadm -A` で使用できるオプションの一覧については、`pntadm(1M)` のマニュアルページを参照してください。また、表 15–4 には `pntadm` コマンドの例が記載されています。

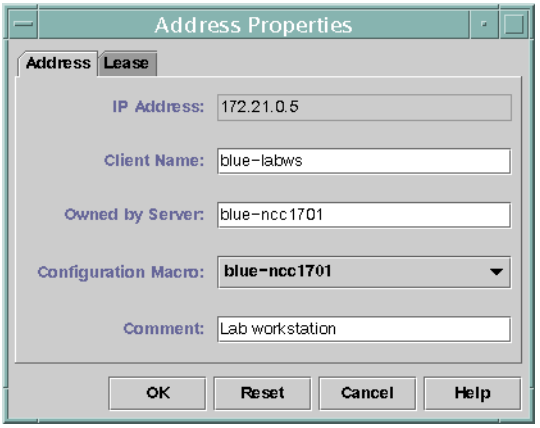
注 – `pntadm` を利用して複数のアドレスを追加するような、スクリプトを作成することもできます。例 18–1 の例を参照してください。

DHCP サービスでの IP アドレスの変更

DHCP マネージャーまたは `pntadm -M` コマンドを使用すると表 15–4 に記載されているアドレス属性を変更できます。`pntadm -M` の詳細は、`pntadm(1M)` のマニュアルページを参照してください。

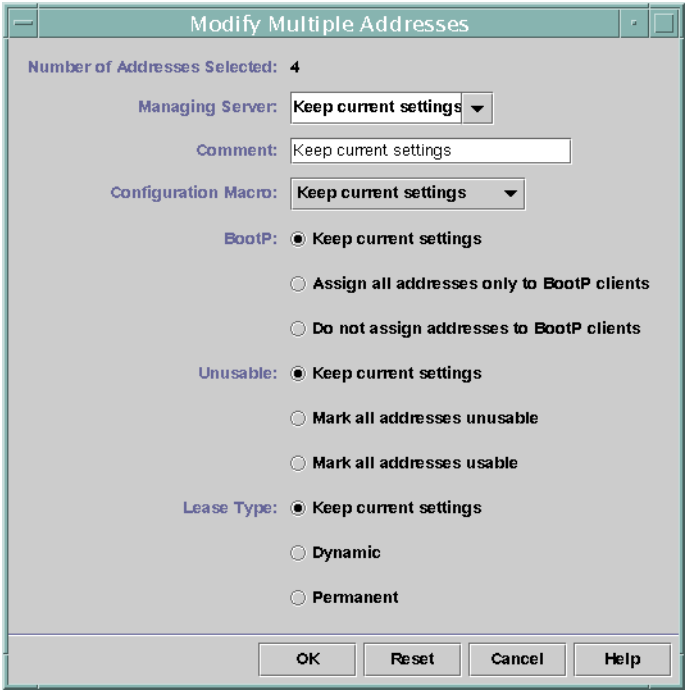
次に、IP アドレスの属性を変更するときに使用する「アドレスの属性 (Address Properties)」ダイアログボックスを示します。

図 15–10　DHCP マネージャの「アドレスの属性 (Address Properties)」ダイアログボックス



次に、複数の IP アドレスを変更するために使用する「複数アドレスの変更 (Modify Multiple Addresses)」ダイアログボックスを示します。

図 15–11　DHCP マネージャの「複数アドレスの変更 (Modify Multiple Addresses)」ダイアログボックス



▼ IPアドレスの属性を変更する方法 (DHCP マネージャ)

1. DHCP マネージャの「アドレス (Addresses)」タブをクリックします。
   358 ページの「DHCP マネージャを起動および停止する方法」を参照してください。
2. その IP アドレスのネットワークを選択します。
3. 変更する IP アドレスを 1 つまたは複数選択します。
   複数のアドレスを変更する場合は、Control キーを押しながらマウスをクリックして、複数のアドレスを選択します。Shift キーを押しながらマウスをクリックして、一定範囲のアドレスを選択することもできます。
4. 「編集 (Edit)」メニューから「属性 (Properties)」を選択します。
   「アドレスの属性 (Address Properties)」ダイアログボックスまたは「複数アドレスの変更 (Modify Multiple Addresses)」ダイアログボックスが開きます。
5. 適切な属性を変更します。
   属性については、「ヘルプ」ボタンをクリックするか、表 15–4 を参照してください。
6. 「了解」をクリックします。

▼ IP アドレスの属性を変更する方法 (pntadm)

1. スーパーユーザーになるか、DHCP 管理プロファイルに割り当てられている役割またはユーザー名になります。
   DHCP 管理プロファイルの詳細については、359 ページの「DHCP コマンドへのユーザーアクセスの設定」を参照してください。
   役割には、認証と特権コマンドが含まれます。役割の詳細については、『Solaris のシステム管理: セキュリティーサービス』の「RBAC の構成 (タスクマップ)」を参照してください。
2. 次の書式のコマンドを入力して IP アドレスの属性を変更します。
   ```
   # pntadm -M ip-address options network-address
   ```
   pntadm コマンドでは、多数のオプションが使用できます。詳細は、pntadm(1M) のマニュアルページを参照してください。
   表 15–4 には、オプションを指定した pntadm コマンドの例が記載されています。

DHCP サービスからの IP アドレスの削除

特定の 1 つまたは複数の IP アドレスについて、DHCP サービスによる管理を停止したい場合があります。DHCP からアドレスを削除する方法は、その変更が一時的なものか永続的なものかによって異なります。

- アドレスを一時的に使用できないようにするには、「Address Properties (アドレスの属性)」ダイアログボックスでそのアドレスを使用不可として設定できます。詳細は、402 ページの「DHCP サービスで IP アドレスを使用不可にする」を参照してください。
- アドレスを DHCP クライアントから永続的に使用できないようにするには、DHCP ネットワークテーブルからそのアドレスを削除する必要があります。詳細は、403 ページの「DHCP サービスからの IP アドレスの削除」を参照してください。

DHCP サービスで IP アドレスを使用不可にする

`-f UNUSABLE` オプションを付けて `pntadm -M` コマンドを使用すると、アドレスを使用不可に指定できます。

個々のアドレスの設定には、DHCP マネージャの「アドレスの属性 (Address Properties)」ダイアログボックス (図 15–10) を使用します。複数のアドレスの設定には、次の手順に説明されている「複数アドレスの変更 (Modify Multiple Addresses)」ダイアログボックス (図 15–11) を使用します。

▼ IP アドレスを使用不可に指定する方法 (DHCP マネージャ)

1. **DHCP マネージャの「アドレス (Addresses)」タブをクリックします。**
   358 ページの「DHCP マネージャを起動および停止する方法」を参照してください。
2. **その IP アドレスのネットワークを選択します。**
3. **使用不可に指定したい IP アドレスを 1 つまたは複数選択します。**
   複数のアドレスを使用不可に指定する場合は、Control キーを押しながらマウスをクリックして、複数のアドレスを選択します。Shift キーを押しながらマウスをクリックして、一定範囲のアドレスを選択することもできます。
4. **「編集 (Edit)」メニューから「属性 (Properties)」を選択します。**
   「アドレスの属性 (Address Properties)」ダイアログボックスまたは「複数アドレスの変更 (Modify Multiple Addresses)」ダイアログボックスが開きます。

5 アドレスを 1 つ変更する場合は、「リース (Lease)」タブを選択します。

6 「アドレスを使用しない (Address is Unusable)」を選択します。
複数のアドレスを編集する場合は、「すべてのアドレスを使用しない (Mark All Addresses Unusable)」を選択します。

7 「了解」をクリックします。

▼ IP アドレスを使用不可に指定する方法 (pntadm)

1 スーパーユーザーになるか、DHCP 管理プロファイルに割り当てられている役割またはユーザー名になります。
DHCP 管理プロファイルの詳細については、359 ページの「DHCP コマンドへのユーザーアクセスの設定」を参照してください。

役割には、認証と特権コマンドが含まれます。役割の詳細については、『Solaris のシステム管理: セキュリティーサービス』の「RBAC の構成 (タスクマップ)」を参照してください。

2 次の書式のコマンドを入力して IP アドレスを使用不可に指定します。

```
# pntadm -M ip-address -f UNUSABLE network-address
```

たとえば、アドレス 10.64.3.3 を使用不可に指定するには、次のように入力します。

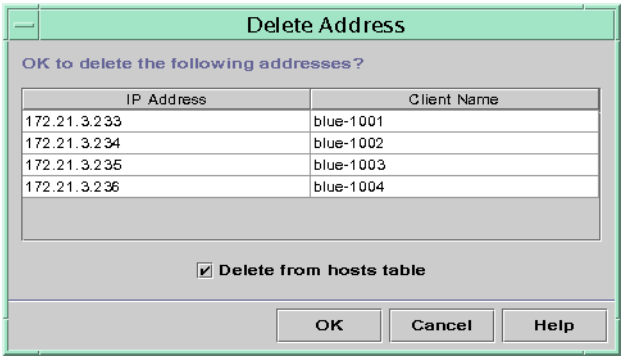
```
pntadm -M 10.64.3.3 -f UNUSABLE 10.64.3.0
```

DHCP サービスからの IP アドレスの削除

IP アドレスを DHCP で管理したくない場合は、DHCP ネットワークテーブルからそのアドレスを削除する必要があります。pntadm -D コマンドまたは DHCP マネージャの「アドレスの削除 (Delete Address)」ダイアログボックスを使用できます。

次に、「アドレスの削除 (Delete Address)」ダイアログボックスを示します。

図 15–12　DHCP マネージャの「アドレスの削除 (Delete Address)」ダイアログボックス



▼ DHCP サービスから IP アドレスを削除する方法 (DHCP マネージャ)

1. **DHCP マネージャの「アドレス (Addresses)」タブをクリックします。**
   358 ページの「DHCP マネージャを起動および停止する方法」を参照してください。

2. **その IP アドレスのネットワークを選択します。**

3. **削除する IP アドレスを選択します。**
   複数のアドレスを削除する場合は、Control キーを押しながらマウスをクリックして、複数のアドレスを選択します。Shift キーを押しながらマウスをクリックして、一定範囲のアドレスを選択することもできます。

4. **「編集 (Edit)」メニューから「削除 (Delete)」を選択します。**
   「アドレスの削除 (Delete Address)」ダイアログボックスに、選択したアドレスがリストされるので、削除する内容を確認できます。

5. **ホスト名をホストテーブルから削除したい場合、「ホストテーブルから削除 (Delete From Hosts Table)」を選択します。**
   ホスト名が DHCP マネージャによって生成されたものである場合、ホストテーブルからその名前を削除できます。

6. **「了解」をクリックします。**

▼ DHCP サービスから IP アドレスを削除する方法 (pntadm)

1 スーパーユーザーになるか、**DHCP** 管理プロファイルに割り当てられている役割またはユーザー名になります。

DHCP 管理プロファイルの詳細については、359 ページの「DHCP コマンドへのユーザーアクセスの設定」を参照してください。

役割には、認証と特権コマンドが含まれます。役割の詳細については、『Solaris のシステム管理: セキュリティーサービス』の「RBAC の構成 (タスクマップ)」を参照してください。

2 次の書式のコマンドを入力して **IP** アドレスを削除します。

```
# pntadm -D ip-address options network-address
```

-y オプションを指定した場合、ホスト名を保持しているネームサービスからホスト名が削除されます。

たとえば、アドレス `10.64.3.3` をネットワーク `10.64.3.0` から削除して、対応するホスト名を削除するには、次のように入力します。

```
pntadm -D 10.64.3.3 -y 10.64.3.0
```

予約済み IP アドレスを DHCP クライアントに割り当てる

DHCP サービスは、以前に DHCP を使用してアドレスを取得したクライアントに同じ IP アドレスを与えようとします。ただし、ときには、アドレスがすでに別のクライアントに再割り当てられていることがあります。

ルーターや、NIS/NIS+ サーバー、DNS サーバーなど、ネットワークにとって致命的なホストは DHCP クライアントになるべきではありません。ネットワークにサービスを提供するホストは、自らの IP アドレスを取得する際にネットワークに依存することは避ける必要があります。同じように、印刷サーバーやファイルサーバーなどのクライアントは、固定 IP アドレスをもつ必要があります。このようなクライアントは、そのネットワーク構成を受け取り、DHCP サーバーから固定 IP アドレスの割り当てることができます。

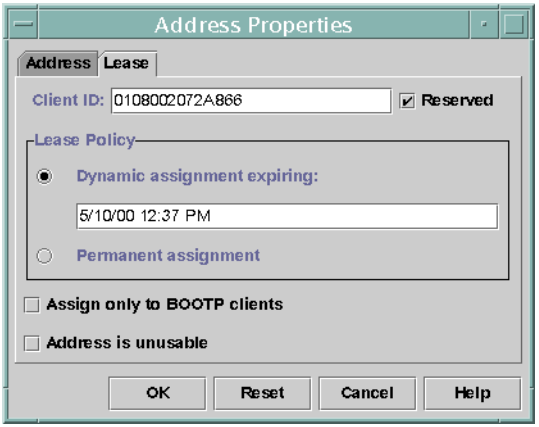
クライアントがその構成を要求するたびに同じ IP アドレスをクライアントに割り当てるように DHCP サーバーを設定できます。そのクライアント用に IP アドレスを予約するには、クライアントに割り当てるアドレスにクライアントの ID を手動で割り当てる必要があります。予約済みアドレスでは、動的リースか常時リースを使用できます。クライアントのアドレスで動的リースを使用する場合には、アドレスの使用を追跡することは簡単です。動的リースで予約済みアドレスを使用すべきクライ

アントの例としては、ディスクレスクライアントがあります。クライアントのアドレスで常時リースを使用する場合には、アドレスの使用を追跡することはできません。クライアントは常時リースを取得してしまうと、クライアントはサーバーに再度アクセスしません。クライアントは、更新された構成情報を取得するためには、IP アドレスを解放し、 DHCP とリースのネゴシエーションを再開する必要があります。

リース属性の設定には、`pntadm -M` コマンドか DHCP マネージャの「アドレスの属性 (Address Properties)」ダイアログボックスが使用できます。

次に、リースを変更するために使用する「アドレスの属性 (Address Properties)」ダイアログボックスの「リース (Lease)」タブを示します。

図 15–13　DHCP マネージャの「リース (Lease)」タブ



▼ 固定 IP アドレスを DHCP クライアントに割り当てる方法 (DHCP マネージャ)

1. **DHCP マネージャの「アドレス (Addresses)」タブをクリックします。**
   358 ページの「DHCP マネージャを起動および停止する方法」を参照してください。
2. 適切なネットワークを選択します。
3. クライアントで使用したい **IP** アドレスをダブルクリックします。
   「アドレスの属性 (Address Properties)」ウィンドウが開きます。
4. 「リース **(Lease)**」タブを選択します。

5 「クライアント ID (Client ID)」フィールドにクライアント ID を入力します。

クライアント ID は、クライアントのハードウェアアドレスから派生したものです。詳細は、表 15–4 の「クライアント ID」の項を参照してください。

6 「予約 (Reserved)」オプションを選択して、その IP アドレスがサーバーによって返還を要求されないようにします。

7 「アドレスの属性 (Address Properties)」ウィンドウの「リースポリシー (Lease Policy)」領域で、「動的 (Dynamic)」または「常時 (Permanent)」の割り当てを選択します。

クライアントでリースを更新するネゴシエーションを行なって、アドレスが使用されている場合に追跡できるようにしたい場合は、「動的 (Dynamic)」を選択します。「予約 (Reserved)」を選択しているので、動的リースが使用されていても、アドレスは再利用できません。このリースの有効期限は指定する必要がありません。DHCP サーバーが、リース期間を使って有効期限を計算します。

「常時 (Permanent)」を選択した場合、トランザクションの記録を有効にしない限り、IP アドレスの使用を追跡できません。

8 「了解」をクリックします。

▼ 固定 IP アドレスを DHCP クライアントに割り当てる方法 (pntadm)

1 スーパーユーザーになるか、DHCP 管理プロファイルに割り当てられている役割またはユーザー名になります。

DHCP 管理プロファイルの詳細については、359 ページの「DHCP コマンドへのユーザーアクセスの設定」を参照してください。

役割には、認証と特権コマンドが含まれます。役割の詳細については、『Solaris のシステム管理: セキュリティーサービス』の「RBAC の構成 (タスクマップ)」を参照してください。

2 次の書式のコマンドを入力してリースフラグを設定します。

```
# pntadm -M ip-address -i client-id -f MANUAL+BOOTP network-address
```

たとえば、MAC アドレスが `08:00:20:94:12:1E` である DHCP クライアントに常に IP アドレス `10.21.5.12` を割り当てるためには、次のように入力します。

```
pntadm -M 10.21.5.12 -i 0108002094121E -f MANUAL+BOOTP 10.21.0.0
```

ヒント – クライアント識別子を決定する方法についての詳細は、表 15–4 の「クライアント ID」の項を参照してください。

DHCP マクロを使用した作業 (作業マップ)

「*DHCP* マクロ」は、DHCP オプションのコンテナです。DHCP サービスはマクロを使用して、クライアントに渡す必要があるオプションをまとめます。サーバーが構成されると、DHCP マネージャと `dhcpconfig` ユーティリティーは、いくつかのマクロを自動的に作成します。マクロの背景情報については、322 ページの「DHCP マクロについて」を参照してください。デフォルトで作成されるマクロについては、第 14 章「DHCP サービスの構成 (手順)」を参照してください。

ネットワークに変更が生じると、クライアントに渡す構成情報を変更しなければならない場合があります。構成情報を変更する場合には、DHCP マクロを使用する必要があります。DHCP マクロは、表示、作成、変更、複製、削除できます。

マクロを使用する場合には、DHCP の標準オプションの知識が必要になります。標準オプションについては、`dhcp_inittab(4)` のマニュアルページを参照してください。

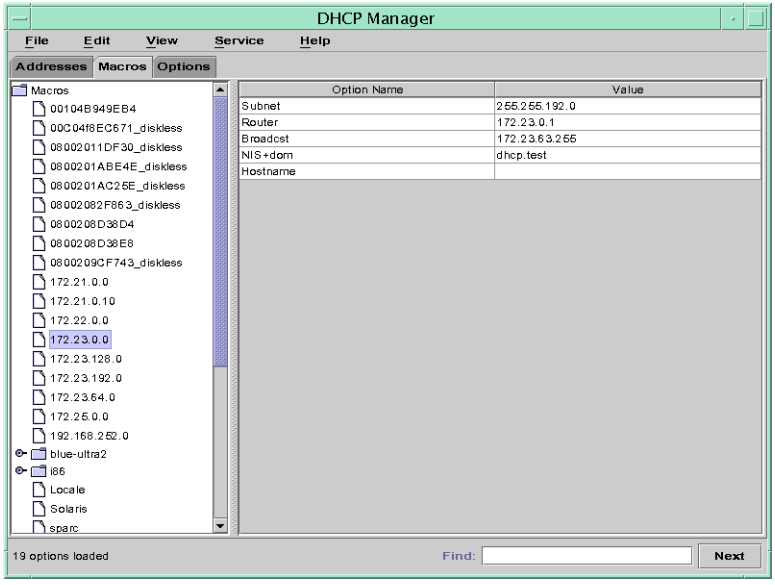
次の作業マップに、DHCP マクロを表示、作成、変更、および削除するのに必要な作業を示します。マップには、各タスクを完了する方法を詳しく説明したセクションのリンクも含まれています。

タスク	説明	参照先
DHCP マクロを表示します。	DHCP サーバーで定義されているすべてのマクロのリストを表示します。	410 ページの「DHCP サーバー上で定義されたマクロを表示する方法 (DHCP マネージャ)」 410 ページの「DHCP サーバー上で定義されたマクロを表示する方法 (dhtadm)」
DHCP マクロを作成します。	DHCP クライアントをサポートする新しいマクロを追加します。	416 ページの「DHCP マクロを作成する方法 (DHCP マネージャ)」 417 ページの「DHCP マクロを作成する方法 (dhtadm)」

タスク	説明	参照先
DHCP クライアントに渡されるマクロ内の値を変更します。	既存のオプションの変更、マクロへのオプションの追加、マクロからのオプションの削除によって、マクロを変更します。	411 ページの「DHCP マクロ内のオプションの値を変更する方法 (DHCP マネージャ)」 412 ページの「DHCP マクロ内のオプションの値を変更する方法 (dhtadm)」 413 ページの「DHCP マクロにオプションを追加する方法 (DHCP マネージャ)」 414 ページの「DHCP マクロにオプションを追加する方法 (dhtadm)」 414 ページの「DHCP マクロからオプションを削除する方法 (DHCP マネージャ)」 415 ページの「DHCP マクロからオプションを削除する方法 (dhtadm)」
DHCP マクロを削除します。	使用しない DHCP マクロを削除します。	418 ページの「DHCP マクロを削除する方法 (DHCP マネージャ)」 418 ページの「DHCP マクロを削除する方法 (dhtadm)」

次に、DHCP マネージャウィンドウの「マクロ (Macros)」タブを示します。

図 15–14　DHCP マネージャの「マクロ (Macros)」タブ



▼ DHCP サーバー上で定義されたマクロを表示する方法 (DHCP マネージャ)

1 DHCP マネージャの「マクロ (Macros)」タブを選択します。

358 ページの「DHCP マネージャを起動および停止する方法」を参照してください。

ウィンドウ左側の「マクロ (Macros)」領域に、この DHCP サーバーで定義されたすべてのマクロがアルファベット順に表示されます。前にフォルダアイコンが付いたマクロには、ほかのマクロへの参照が含まれていますが、前にドキュメントアイコンが付いたマクロには、ほかのマクロへの参照が含まれていません。

2 マクロフォルダを開くには、フォルダアイコンの左にあるハンドルアイコンをクリックします。

選択したマクロに含まれるマクロがリストされます。

3 マクロの内容を表示するには、マクロ名をクリックします。

オプションとそれらに割り当てられた値が表示されます。

▼ DHCP サーバー上で定義されたマクロを表示する方法 (dhtadm)

1 スーパーユーザーになるか、DHCP 管理プロファイルに割り当てられている役割またはユーザー名になります。

DHCP 管理プロファイルの詳細については、359 ページの「DHCP コマンドへのユーザーアクセスの設定」を参照してください。

役割には、認証と特権コマンドが含まれます。役割の詳細については、『Solaris のシステム管理: セキュリティーサービス』の「RBAC の構成 (タスクマップ)」を参照してください。

2 次のコマンドを入力してマクロを表示します。

```
# dhtadm -P
```

このコマンドは、`dhcptab` テーブルの内容 (DHCP サーバー上で定義されたすべてのマクロとシンボルを含む) をフォーマットして標準出力に出力します。

DHCP マクロの変更

ネットワークの一部の設定が変更され、1 台または複数の DHCP クライアントにその変更を通知する必要がある場合、マクロを変更する必要がある場合があります。た

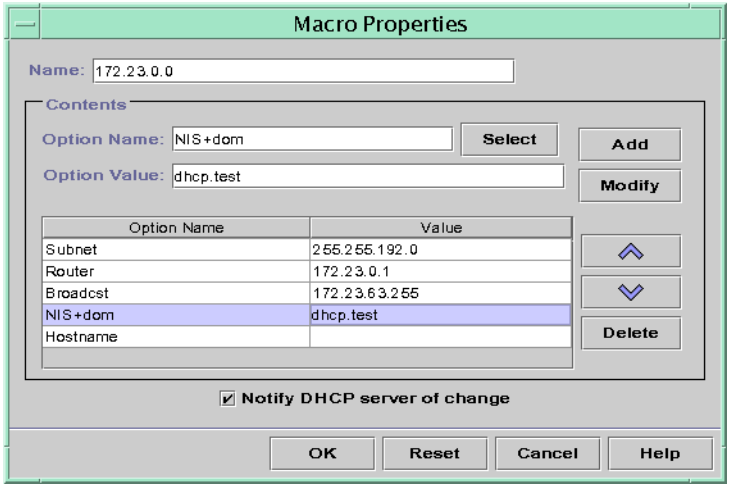
とえば、ルーターや NIS サーバーを追加したり、新しいサブネットを作成したり、リースポリシーを変更したりした場合です。

マクロを変更する際には、まず、変更、追加、または削除したい DHCP オプションの名前が何か知らなければなりません。標準的な DHCP オプションについては、DHCP マネージャのヘルプや `dhcp_inittab(4)` のマニュアルページを参照してください。

`dhtadm -M -m` コマンドまたは DHCP マネージャを使用して、マクロを変更できます。`dhtadm` の詳細については、`dhtadm(1M)` のマニュアルページを参照してください。

次に、DHCP マネージャの「マクロの属性 (Macro Properties)」ダイアログボックスを示します。

図 15–15　DHCP マネージャの「マクロの属性 (Macro Properties)」ダイアログボックス



▼ DHCP マクロ内のオプションの値を変更する方法 (DHCP マネージャ)

1. **DHCP マネージャの「マクロ (Macros)」タブを選択します。**
   358 ページの「DHCP マネージャを起動および停止する方法」を参照してください。
2. 変更するマクロを選択します。

3 「編集 (Edit)」メニューから「属性 (Properties)」を選択します。
「マクロの属性 (Macro Properties)」ダイアログボックスが開きます。

4 「オプション (Options)」のテーブルで、変更するオプションを選択します。
このオプションの名前とその値は、「オプション名 (Option Name)」と「オプションの値 (Option Value)」のフィールドに表示されます。

5 「オプションの値 (Option Value)」フィールドで、古い値を選択し、そのオプションの新しい値を入力します。

6 「変更」をクリックします。
新しい値がオプションテーブルに表示されます。

7 「DHCP サーバーに変更を通知 (Notify DHCP Server of Change)」を選択します。
この選択によって、DHCP サーバーは `dhcptab` テーブルを再読み込みし、「了解 (OK)」をクリックすると直ちに変更が適用されます。

8 「了解」をクリックします。

▼ DHCP マクロ内のオプションの値を変更する方法 (dhtadm)

1 スーパーユーザーになるか、DHCP 管理プロファイルに割り当てられている役割またはユーザー名になります。
DHCP 管理プロファイルの詳細については、359 ページの「DHCP コマンドへのユーザーアクセスの設定」を参照してください。

役割には、認証と特権コマンドが含まれます。役割の詳細については、『Solaris のシステム管理: セキュリティーサービス』の「RBAC の構成 (タスクマップ)」を参照してください。

2 次の書式のコマンドを入力してオプションの値を変更します。

```
# dhtadm -M -m macroname -e 'option=value:option=value' -g
```

たとえば、マクロ `bluenote` 内のリース期間、および UTC との時間差を変更する場合には、次のように入力します。

```
# dhtadm -M -m bluenote -e 'LeaseTim=43200:UTCOffst=28800' -g
```

▼ DHCP マクロにオプションを追加する方法 (DHCP マネージャ)

1. DHCP マネージャの「マクロ (Macros)」タブを選択します。
   358 ページの「DHCP マネージャを起動および停止する方法」を参照してください。
2. 変更するマクロを選択します。
3. 「編集 (Edit)」メニューから「属性 (Properties)」を選択します。
   「マクロの属性 (Macro Properties)」ダイアログボックスが開きます。
4. 「オプション名 (Option Name)」フィールドで、次のどちらかの方法を使用して、オプション名を指定します。
   - 「オプション名 (Option Name)」フィールドの隣にある「選択 (Select)」ボタンをクリックして、マクロに追加するオプションを選択します。
     「オプションの選択 (Select Option)」ダイアログボックスに、標準カテゴリのオプションの名前と説明がアルファベット順にリストされます。標準カテゴリ以外のオプションを追加する場合は、「カテゴリ (Category)」リストを使用してカテゴリを選択してください。

     マクロのカテゴリについては、322 ページの「DHCP マクロについて」を参照してください。
   - 既存のマクロへの参照を新しいマクロに含めたい場合は、**Include** と入力してください。
5. 「オプションの値 (Option Value)」フィールドにオプションの値を入力します。
   オプション名を **Include** と入力した場合は、「オプションの値 (Option Value)」フィールドに既存のマクロの名前を指定する必要があります。
6. 「追加」をクリックします。
   このオプションは、このマクロのオプションリストの一番下に追加されます。マクロ内でのオプションの位置を変更する場合は、オプションを選択し、矢印ボタンをクリックしてオプションを上下に移動させます。
7. 「DHCP サーバーに変更を通知 (Notify DHCP Server of Change)」を選択します。
   この選択によって、DHCP サーバーは dhcptab テーブルを再読み込みし、「了解 (OK)」をクリックすると直ちに変更が適用されます。
8. 「了解」をクリックします。

▼ DHCP マクロにオプションを追加する方法 (dhtadm)

1 スーパーユーザーになるか、DHCP 管理プロファイルに割り当てられている役割またはユーザー名になります。

DHCP 管理プロファイルの詳細については、359 ページの「DHCP コマンドへのユーザーアクセスの設定」を参照してください。

役割には、認証と特権コマンドが含まれます。役割の詳細については、『Solaris のシステム管理: セキュリティーサービス』の「RBAC の構成 (タスクマップ)」を参照してください。

2 次の書式のコマンドを入力してマクロにオプションを追加します。

```
# dhtadm -M -m macroname -e 'option=value' -g
```

たとえば、リースのネゴシエーションを行うオプションをマクロ bluenote に追加するには、次のコマンドを入力します。

```
# dhtadm -M -m bluenote -e 'LeaseNeg=_NULL_VALUE' -g
```

値を必要としないオプションの場合、オプションの値として _NULL_VALUE を使用してください。

▼ DHCP マクロからオプションを削除する方法 (DHCP マネージャ)

1 DHCP マネージャの「マクロ (Macros)」タブを選択します。

358 ページの「DHCP マネージャを起動および停止する方法」を参照してください。

2 変更するマクロを選択します。

3 「編集 (Edit)」メニューから「属性 (Properties)」を選択します。

「マクロの属性 (Macro Properties)」ダイアログボックスが開きます。

4 マクロから削除するオプションを選択します。

5 「削除」をクリックします。

選択されたオプションが、このマクロに関するオプションのリストから削除されます。

6 「**DHCP サーバーに変更を通知 (Notify DHCP Server of Change)**」を選択します。

この選択によって、DHCP サーバーは `dhcptab` テーブルを再読み込みし、「了解 (OK)」をクリックすると直ちに変更が適用されます。

7 「了解」をクリックします。

▼ DHCP マクロからオプションを削除する方法 (dhtadm)

1 スーパーユーザーになるか、**DHCP 管理プロファイル**に割り当てられている役割またはユーザー名になります。

DHCP 管理プロファイルの詳細については、359 ページの「DHCP コマンドへのユーザーアクセスの設定」を参照してください。

役割には、認証と特権コマンドが含まれます。役割の詳細については、『Solaris のシステム管理: セキュリティーサービス』の「RBAC の構成 (タスクマップ)」を参照してください。

2 次の書式のコマンドを入力してマクロからオプションを削除します。

```
# dhtadm -M -m macroname -e 'option=' -g
```

たとえば、リースのネゴシエーションを行うオプションをマクロ `bluenote` から削除するには、次のコマンドを入力します。

```
# dhtadm -M -m bluenote -e 'LeaseNeg=' -g
```

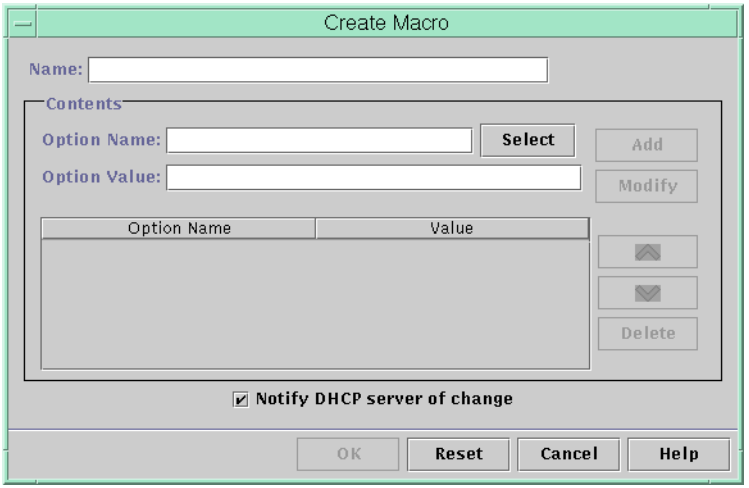
オプションに値を指定しなかった場合、オプションはマクロから削除されます。

DHCP マクロの作成

特別な要求を持ったクライアントをサポートするために、DHCP サービスに新しいマクロを追加する場合があります。`dhtadm -A -m` コマンドまたは DHCP マネージャの「マクロの作成 (Create Macro)」ダイアログボックスを使用して、マクロを追加できます。`dhtadm` コマンドの詳細については、`dhtadm(1M)` のマニュアルページを参照してください。

次に、DHCP マネージャの「マクロの作成 (Create Macro)」ダイアログボックスを示します。

図 15–16　DHCP マネージャの「マクロの作成 (Create Macro)」ダイアログボックス



▼ DHCP マクロを作成する方法 (DHCP マネージャ)

1. **DHCP マネージャの「マクロ (Macros)」タブを選択します。**
   358 ページの「DHCP マネージャを起動および停止する方法」を参照してください。

2. **「編集 (Edit)」メニューから「作成 (Create)」を選択します。**
   「マクロの作成 (Create Macro)」ダイアログボックスが開きます。

3. **そのマクロの名前 (固有の名前) を入力します。**
   名前には 128 文字までの英数字を使用できます。ベンダークラス識別子、ネットワークアドレス、またはクライアント ID に一致する名前を使用している場合は、そのマクロは適切なクライアントに対して自動的に処理されます。異なる名前を使用している場合は、自動的には処理されません。そのマクロは、特定の IP アドレスに割り当てられているか、または自動的に処理される別のマクロに含まれていなければなりません。詳細は、322 ページの「DHCP サーバーによるマクロ処理」を参照してください。

4. **「オプション名 (Optioin Name)」フィールドの隣にある「選択 (Select)」ボタンをクリックします。**
   「オプションの選択 (Select Option)」ダイアログボックスに、標準カテゴリのオプションの名前と説明がアルファベット順にリストされます。標準カテゴリ以外のオプションを追加したい場合は、「カテゴリ (Category)」リストを使用します。必要なカテゴリを「カテゴリ (Category)」リストから選択してください。オプションのカテゴリについての詳細は、321 ページの「DHCP オプションについて」を参照してください。

5 マクロに追加するオプションを選択して、「了解 (OK)」をクリックします。
「マクロの属性 (Macro Properties)」ダイアログボックスが、「オプション名 (Option Name)」フィールドに選択されたオプションを表示します。

6 「オプションの値 (Option Value)」フィールドにオプションの値を入力し、「追加 (Add)」をクリックします。
このオプションは、このマクロのオプションリストの一番下に追加されます。マクロ内でのオプションの位置を変更する場合は、オプションを選択し、矢印ボタンをクリックしてオプションを上下に移動させます。

7 マクロに追加するオプションごとに、手順 5 と手順 6 を繰り返します。

8 オプションの追加が終了したら、「DHCP サーバーに変更を通知 (Notify DHCP Server of Change)」を選択します。
この選択によって、DHCP サーバーは dhcptab テーブルを再読み込みし、「了解 (OK)」をクリックすると直ちに変更が適用されます。

9 「了解」をクリックします。

▼ DHCP マクロを作成する方法 (dhtadm)

1 スーパーユーザーになるか、DHCP 管理プロファイルに割り当てられている役割またはユーザー名になります。
DHCP 管理プロファイルの詳細については、359 ページの「DHCP コマンドへのユーザーアクセスの設定」を参照してください。

役割には、認証と特権コマンドが含まれます。役割の詳細については、『Solaris のシステム管理: セキュリティーサービス』の「RBAC の構成 (タスクマップ)」を参照してください。

2 次の書式のコマンドを入力してマクロを作成します。

```
# dhtadm -A -m macroname -d ':option=value:option=value:option=value:' -g
```

d への引数として指定する -option=value ペアの数に制限はありません。引数はコロンで始まり、コロンで終わる必要があります。さらに、それぞれの *option=value* ペアはコロンで区切る必要があります。また、文字列全体を引用符で囲む必要があります。

たとえば、マクロ bluenote を作成するには、次のコマンドを入力します。

```
# dhtadm -A -m bluenote -d ':Router=10.63.6.121\ :LeaseNeg=_NULL_VALUE:
DNSserv=10.63.28.12:' -g
```

値を必要としないオプションの場合、オプションの値として _NULL_VALUE を使用してください。

DHCP マクロの削除

DHCP サービスからマクロを削除する場合があります。たとえば、DHCP サービスからネットワークを削除する場合、関連するネットワークマクロも削除できます。

dhtadm -D -m コマンドまたは DHCP マネージャを使用して、マクロを削除できます。

▼ DHCP マクロを削除する方法 (DHCP マネージャ)

1. **DHCP マネージャの「マクロ (Macros)」タブを選択します。**
   358 ページの「DHCP マネージャを起動および停止する方法」を参照してください。
2. 削除するマクロを選択します。
   「マクロの削除 (Delete Macro)」ダイアログボックスは、指定したマクロの削除を確認するように求めます。
3. **「DHCP サーバーに変更を通知 (Notify DHCP Server of Change)」を選択します。**
   この選択によって、DHCP サーバーは dhcptab テーブルを再読み込みし、「了解 (OK)」をクリックすると直ちに変更が適用されます。
4. 「了解」をクリックします。

▼ DHCP マクロを削除する方法 (dhtadm)

1. スーパーユーザーになるか、**DHCP 管理プロファイルに割り当てられている役割またはユーザー名になります。**
   DHCP 管理プロファイルの詳細については、359 ページの「DHCP コマンドへのユーザーアクセスの設定」を参照してください。
   役割には、認証と特権コマンドが含まれます。役割の詳細については、『Solaris のシステム管理: セキュリティーサービス』の「RBAC の構成 (タスクマップ)」を参照してください。
2. 次の書式のコマンドを入力してマクロを削除します。
   # dhtadm -D -m *macroname* **-g**
   たとえば、マクロ bluenote を削除するには、次のコマンドを入力します。

```
# dhtadm -D -m bluenote -g
```

DHCP オプションを使用した作業 (作業マップ)

オプションは、DHCP サーバーがクライアントに渡すネットワーク構成パラメータのキーワードです。DHCP サービスでは、標準の DHCP オプションを作成、削除、変更することはできません。標準オプションは DHCP プロトコルによって定義されたものだからです。つまり、タスクの対象となるオプションは、サイト用に独自に作成したものだけです。そのため、初めて DHCP サービスを設定すると、サイト用のオプションを作成するまでは、DHCP マネージャの「オプション (Options)」タブは空です。

DHCP サーバー上でオプションを作成する場合、DHCP クライアント上でもそのオプションに関する情報を追加する必要があります。DHCP クライアント用に、`/etc/dhcp/inittab` ファイルを編集して、新しいオプションのエントリを追加する必要があります。このファイルについては、`dhcp_inittab(4)` のマニュアルページを参照してください。

Oracle Solaris クライアント以外の DHCP クライアントを使用している場合、オプションまたはシンボルを追加する方法については、使用しているクライアント用のマニュアルを参照してください。DHCP オプションの詳細は、321 ページの「DHCP オプションについて」を参照してください。

DHCP マネージャまたは `dhtadm` コマンドを使用して、オプションを作成、変更、削除できます。

ヒント – DHCP の文献では、オプションを「シンボル」と呼びます。`dhtadm` コマンドとそれに関連するマニュアルページでもオプションをシンボルと呼びます。

次の作業マップに、DHCP オプションを作成、変更、および削除するのに必要な作業を示します。タスクマップには、それぞれの手順へのリンクが含まれています。

タスク	説明	参照先
DHCP オプションを作成します。	標準的な DHCP オプションで扱わない情報に関する新しいオプションを追加します。	423 ページの「DHCP オプションを作成する方法 (DHCP マネージャ)」 424 ページの「DHCP オプションを作成する方法 (dhtadm)」 428 ページの「DHCP クライアントのオプション情報の変更」

タスク	説明	参照先
DHCP オプションを変更します。	作成済みの DHCP オプションの属性を変更します。	425 ページの「DHCP オプションの属性を変更する方法 (DHCP マネージャ)」 426 ページの「DHCP オプションの属性を変更する方法 (dhtadm)」
DHCP オプションを削除します。	作成済みの DHCP オプションを削除します。	427 ページの「DHCP オプションを削除する方法 (DHCP マネージャ)」 428 ページの「DHCP オプションを削除する方法 (dhtadm)」

DHCP オプションを作成する前に、次の表に示すオプションの属性をよく理解しておく必要があります。

表 15–5 DHCP オプションの属性

オプションの属性	説明
カテゴリ	オプションの「カテゴリ」は、次のいずれかにする必要があります。 ■ ベンダー – クライアントのベンダーのプラットフォームに固有のオプションであり、ハードウェアかソフトウェアになります。 ■ サイト – サイトに固有のオプション。 ■ 拡張 – DHCP プロトコルに追加された新しいオプションですが、まだ DHCP の標準オプションとして実装されていません。
コード	「コード」は、オプションに割り当てる一意の番号です。同じオプションカテゴリ内のほかのオプションで、同じコードを使用することはできません。オプションカテゴリに対して適切なコードにする必要があります。 ■ ベンダー – ベンダークラスごとに 1 から 254 のコード値 ■ サイト – 128 から 254 のコード値 ■ 拡張 – 77 から 127 のコード値

表 15–5 DHCP オプションの属性 (続き)

オプションの属性	説明
データ型	「データ型」は、そのオプションの値として割り当てることができるデータの種類を指定します。有効なデータ型は次のとおりです。 ■ ASCII – テキスト文字列値。 ■ BOOLEAN – ブール型のデータ型に関連値はありません。このオプションが存在すれば条件は真となり、存在しなければ偽となります。たとえば、`Hostname` オプションはブール型です。マクロに `Hostname` が含まれていると、DHCP サーバーは、割り当てられたアドレスに関連するホスト名を検索します。 ■ IP – ドットで区切られた 10 進法形式 (*xxx.xxx.xxx.xxx*) の 1 つまたは複数のアドレス。 ■ OCTET – 2 進データを翻訳されない ASCII で表示したもの。たとえば、クライアント ID は、この 16 進形式のデータ型を使用します。有効な文字は 0–9、A–F、a–f です。8 ビットを表すには 2 つの ASCII 文字が必要です。 ■ UNUMBER8, UNUMBER16, UNUMBER32, UNUMBER64, SNUMBER8, SNUMBER16, SNUMBER32, または SNUMBER64 の数値。初めの U や S は、数値が符号付きか符号なしかを表します。終わりの数字は、数値にいくつのビットが含まれているかを表します。
データ単位数	「データの単位数」では、オプション値全体を表すために必要なデータ型の「インスタンス」の個数を指定します。たとえば、IP のデータ型でデータ単位数 2 の場合、オプション値には 2 つの IP アドレスが含まれる必要があります。
最大値	オプションについて指定可能な値の最大個数。たとえば、最大値が 2、データの単位数が 2、データ型が IP だとします。この場合には、オプションの値に最大 2 つの IP アドレスペアを含めることができます。

表 15–5 DHCP オプションの属性 (続き)

オプションの属性	説明
ベンダークライアントクラス	このオプションは、オプションカテゴリがベンダーの場合のみ利用できます。ベンダークライアントクラスとは、そのベンダーオプションが関連付けられているクライアントクラスを識別するものです。このクラスは、クライアントのマシンタイプやオペレーティングシステムを表すASCII 文字列からなります。たとえば、Sun ワークステーションのあるモデルのクラス文字列は SUNW.Sun-Blade-100 です。このタイプのオプションで定義する構成パラメータは、同じクラスのすべてのクライアント (かつ、そのクラスのクライアントだけ) に渡されます。 複数のクライアントクラスを指定できます。指定するクラスと一致するクライアントクラス値の DHCP クライアントだけが、そのクラスに含まれるオプションを受け取ります。 クライアントクラスは DHCP クライアントのベンダーによって決定されます。Oracle Solaris クライアント以外の DHCP クライアントの場合、クライアントクラスについては、DHCP クライアントのベンダーのマニュアルを参照してください。 Oracle Solaris クライアントの場合、クライアント上で prtconf -b コマンドを入力すると、ベンダーのクライアントクラスを取得できます。ベンダークライアントクラスを指定するには、uname コマンドで返される文字列の中のすべてのカンマをピリオドに置き換えます。たとえば、prtconf -b コマンドで文字列 SUNW,Sun-Blade-100 が返される場合、ベンダーのクライアントクラスを SUNW.Sun-Blade-100 と指定してください。

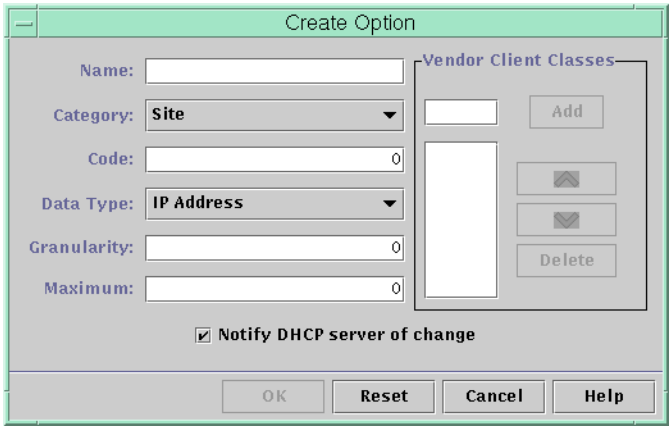
DHCP オプションの作成

渡す必要があるクライアント情報に対応するオプションが DHCP プロトコルにない場合は、オプションを作成できます。独自のオプションを作成する前に、dhcp_inittab(4) のマニュアルページを参照して、DHCP で定義されているオプションのリストを調べてください。

dhtadm -A -s コマンドまたは DHCP マネージャの「オプションの作成 (Create Option)」ダイアログボックスを使用して、新しいオプションを作成できます。

次に、DHCP マネージャの「オプションの作成 (Create Option)」ダイアログボックスを示します。

図 15–17　DHCP マネージャの「オプションの作成 (Create Option)」ダイアログボックス



▼ DHCP オプションを作成する方法 (DHCP マネージャ)

1. **DHCP マネージャの「オプション (Options)」タブを選択します。**
   358 ページの「DHCP マネージャを起動および停止する方法」を参照してください。

2. **「編集 (Edit)」メニューから「作成 (Create)」を選択します。**
   「オプションの作成 (Create Option)」ダイアログボックスが開きます。

3. **新しいオプションの略式記述名を入力します。**
   この名前には、128 文字までの英数字と空白文字を含めることができます。

4. **ダイアログボックスの各設定について、値を入力または選択します。**
   各設定については、表 15–5 を参照するか、DHCP マネージャーのヘルプを参照してください。

5. **オプションの作成が終わったら、「DHCP サーバーに変更を通知 (Notify DHCP Server of Change)」を選択します。**
   この選択によって、DHCP サーバーは `dhcptab` テーブルを再読み込みし、「了解 (OK)」をクリックすると直ちに変更が適用されます。

6. **「了解」をクリックします。**
   これでオプションをマクロに追加し、クライアントに渡すオプションに値を割り当てることができます。

▼ DHCP オプションを作成する方法 (dhtadm)

1 スーパーユーザーになるか、**DHCP** 管理プロファイルに割り当てられている役割またはユーザー名になります。

DHCP 管理プロファイルの詳細については、359 ページの「DHCP コマンドへのユーザーアクセスの設定」を参照してください。

役割には、認証と特権コマンドが含まれます。役割の詳細については、『Solaris のシステム管理: セキュリティーサービス』の「RBAC の構成 (タスクマップ)」を参照してください。

2 次の書式のコマンドを入力して **DHCP** オプションを作成します。

```
# dhtadm -A -s option-name -d 'category,code,data-type,granularity,maximum' -g
```

option-name 128 文字以内の英数字文字列です。

category 次のうちの 1 つです。`Site`、`Extend`、または `Vendor=`*list-of-classes*。*list-of-classes* は、オプションが適用されるベンダークライアントクラスを空白文字で区切ったものです。ベンダークライアントクラスを決定する方法については、表 15–5 を参照してください。

code このオプションカテゴリに適した 1 つの数値です。表 15–5 の説明を参照してください。

data-type このオプションで渡されるデータのタイプを示すキーワードで指定されます。表 15–5 の説明を参照してください。

granularity 負ではない数として指定されます。表 15–5 の説明を参照してください。

maximum 負ではない数です。表 15–5 の説明を参照してください。

例 15–3 dhtadm による DHCP オプションの作成

次のコマンドを実行すると、Site カテゴリに属する `NewOpt` というオプションが作成されます。このオプションのコードは 130 です。このオプションの値には、1 つの 8 ビット無符号整数が設定できます。

```
# dhtadm -A -s NewOpt -d 'Site,130,UNUMBER8,1,1' -g
```

次のコマンドを実行すると、Vendor カテゴリオプションに属する `NewServ` というオプションが作成されます。このオプションは、マシンタイプが `SUNW,Sun-Blade-100` または `SUNW,Sun-Blade-1000` であるクライアントに適用されます。このオプションのコードは 200 です。このオプションの値には、1 つの IP アドレスが設定できます。

```
# dhtadm -A -s NewServ -d 'Vendor=SUNW.Sun-Blade-100 \ SUNW.Sun-Blade-1000,200,IP,1,1' -g
```

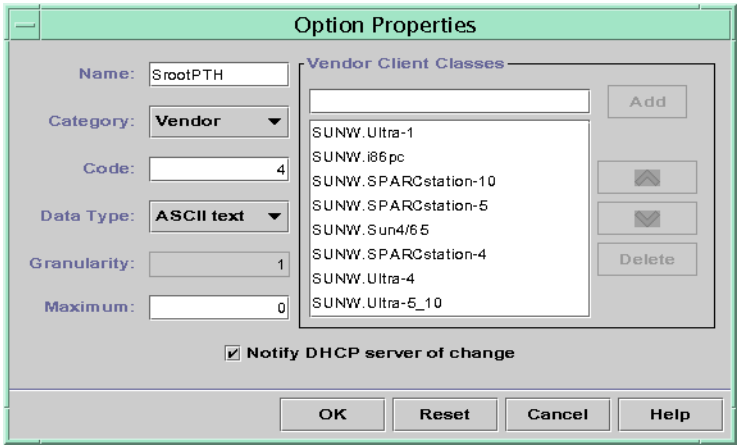
DHCP オプションの変更

DHCP サービス用のオプションをすでに作成している場合、そのオプションの属性は変更できます。オプションの変更には、`dhtadm -M -s` コマンドか、DHCP マネージャの「オプションの属性 (Option Properties)」ダイアログボックスを使用します。

DHCP クライアントのオプション情報を変更して、その変更内容を DHCP サーバーに反映する必要があることに注意してください。428 ページの「DHCP クライアントのオプション情報の変更」を参照してください。

次に、DHCP マネージャの「オプションの属性 (Option Properties)」ダイアログボックスを示します。

図 15–18　DHCP マネージャの「オプションの属性 (Option Properties)」ダイアログボックス



▼ DHCP オプションの属性を変更する方法 (DHCP マネージャ)

1. **DHCP マネージャの「オプション (Options)」タブを選択します。**
   358 ページの「DHCP マネージャを起動および停止する方法」を参照してください。
2. 変更するオプションを選択します。
3. 「編集 (Edit)」メニューから「属性 (Properties)」を選択します。
   「オプションの属性 (Option Properties)」ダイアログボックスが開きます。

4 必要に応じて属性を編集します。
属性については、表 15–5 を参照するか、DHCP マネージャのヘルプを参照してください。

5 オプションの処理が終了したら、「**DHCP サーバーに変更を通知 (Notify DHCP Server of Change)**」を選択します。
変更は dhcptab テーブルに対して行われます。DHCP サーバーは、dhcptab テーブルを再読み込みして変更を有効にします。

6 「了解」をクリックします。

▼ DHCP オプションの属性を変更する方法 (dhtadm)

1 スーパーユーザーになるか、**DHCP** 管理プロファイルに割り当てられている役割またはユーザー名になります。
DHCP 管理プロファイルの詳細については、359 ページの「DHCP コマンドへのユーザーアクセスの設定」を参照してください。
役割には、認証と特権コマンドが含まれます。役割の詳細については、『Solaris のシステム管理: セキュリティーサービス』の「RBAC の構成 (タスクマップ)」を参照してください。

2 次の書式のコマンドを入力してオプションを変更します。

```
# dhtadm -M -s option-name -d 'category,code,data-type,granularity,maximum' -g
```

option-name 変更するオプションの名前を指定します。

category Site、Extend、または Vendor=*list-of-classes* のうちの 1 つです。*list-of-classes* は、オプションが適用されるベンダークライアントクラスを空白文字で区切ったものです。たとえば、SUNW.Sun-Blade-100 SUNW.Ultra-80 SUNWi86pc。

code このオプションカテゴリに適した 1 つの数値を指定します。表 15–5 の説明を参照してください。

data-type このオプションで渡されるデータのタイプを示すキーワードを指定します。表 15–5 の説明を参照してください。

granularity 負ではない数です。表 15–5 の説明を参照してください。

maximum 負ではない数です。表 15–5 の説明を参照してください。

変更する属性だけでなく、DHCP オプション属性すべてを -d スイッチで指定する必要があります。

例 15–4 dhtadm による DHCP オプションの変更

次のコマンドでは、NewOpt というオプションを変更します。このオプションは Site カテゴリのオプションです。このオプションのコードは 135 です。このオプションの値には、1 つの 8 ビット無符号整数が設定できます。

```
# dhtadm -M -s NewOpt -d 'Site,135,UNUMBER8,1,1'
```

次のコマンドでは、Vendor カテゴリに属する NewServ というオプションを変更します。その結果、このオプションは、マシンタイプが SUNW,Sun-Blade-100 または SUNW,i86pc であるクライアントに適用されます。このオプションのコードは 200 です。このオプションの値には、1 つの IP アドレスが設定できます。

```
# dhtadm -M -s NewServ -d 'Vendor=SUNW.Sun-Blade-100 \ SUNW.i86pc,200,IP,1,1' -g
```

DHCP オプションの削除

標準の DHCP オプションを削除することはできません。しかし、ユーザーが DHCP サービス用に定義したオプションは、DHCP マネージャか dhtadm コマンドを使って削除できます。

▼ DHCP オプションを削除する方法 (DHCP マネージャ)

1. DHCP マネージャの「オプション (Options)」タブを選択します。
   358 ページの「DHCP マネージャを起動および停止する方法」を参照してください。
2. 削除したいオプションを選択します。
3. 「編集 (Edit)」メニューから「削除 (Delete)」を選択します。
   「オプションの削除 (Delete Option)」ダイアログボックスが開きます。
4. オプションの削除が終わったら、「DHCP サーバーに変更を通知 (Notify DHCP Server of Change)」を選択します。
   この選択によって、DHCP サーバーは dhcptab テーブルを再読み込みし、「了解 (OK)」をクリックすると直ちに変更が適用されます。
5. 「了解」をクリックします。

▼ DHCP オプションを削除する方法 (dhtadm)

1 スーパーユーザーになるか、**DHCP** 管理プロファイルに割り当てられている役割またはユーザー名になります。

DHCP 管理プロファイルの詳細については、359 ページの「DHCP コマンドへのユーザーアクセスの設定」を参照してください。

役割には、認証と特権コマンドが含まれます。役割の詳細については、『Solaris のシステム管理: セキュリティーサービス』の「RBAC の構成 (タスクマップ)」を参照してください。

2 次の書式のコマンドを入力して **DHCP** オプションを削除します。

```
# dhtadm -D -s option-name -g
```

DHCP クライアントのオプション情報の変更

新しい DHCP オプションを DHCP サーバーに追加する場合、各 DHCP クライアントのオプション情報に、補足エントリを追加する必要があります。DHCP クライアント以外のクライアントを使用している場合、そのクライアントのドキュメントでオプションまたはシンボルの追加に関する情報を確認してください。

DHCP クライアントでは、`/etc/dhcp/inittab` ファイルを編集して、DHCP サーバーに追加するオプションごとにエントリを追加する必要があります。後にそのオプションをサーバー上で変更する場合、クライアントの `/etc/dhcp/inittab` ファイルのエントリも変更する必要があります。

`/etc/dhcp/inittab` ファイルの構文については、`dhcp_inittab(4)` のマニュアルページを参照してください。

注 – 以前の Oracle Solaris リリースで `dhcptags` ファイルに DHCP オプションを追加していた場合、それらのオプションを `/etc/dhcp/inittab` ファイルに追加する必要があります。詳細は、498 ページの「DHCP のオプション情報」を参照してください。

DHCP サービスを使用した Oracle Solaris ネットワークインストールのサポート

DHCP を使えば、ネットワーク上の一定のクライアントシステムに Oracle Solaris をインストールできます。この機能を使用できるシステムは、Oracle Solaris のハードウェア要件を満たす `sun4u` ベースシステムと x86 システムだけです。DHCP を使って、クライアントシステムをそのブート時にネットワーク向けに自動的に構成

する方法については、『Oracle Solaris 10 1/13 インストールガイド: ネットワークベースのインストール』の第 2 章「システム構成情報の事前構成 (タスク)」を参照してください。

さらに、DHCP では、HTTP を使ってリモートサーバーから広域ネットワーク (WAN) を介して Oracle Solaris クライアントシステムのブートとインストールを行うことができます。リモートからブートとインストールを行うこの方法を「*WAN ブートインストール*」といいます。WAN ブートを使用すると、大規模なパブリックネットワークを介して Oracle Solaris を SPARC ベースのシステムにインストールできますが、このようなネットワークはインフラストラクチャーの信頼性が低い場合があります。WAN ブートをセキュリティー機能とともに使用することによって、データの機密性とインストールイメージの完全性を保護できます。

DHCP で WAN ブートを使ってリモートからクライアントシステムのブートとインストールをできるようにするには、次の情報をクライアントに提供できるように DHCP サーバーを構成する必要があります。

- プロキシサーバーの IP アドレス
- `wanboot-cgi` プログラムの場所

この情報が得られるように DHCP サーバーを構成する方法については、『Oracle Solaris 10 1/13 インストールガイド: ネットワークベースのインストール』の第 2 章「システム構成情報の事前構成 (タスク)」を参照してください。DHCP サーバーから WAN を介してクライアントシステムのブートとインストールを行う方法については、『Oracle Solaris 10 1/13 インストールガイド: ネットワークベースのインストール』の第 10 章「WAN ブート (概要)」を参照してください。

ディスクレスクライアントのサポートについては、429 ページの「リモートブートクライアントとディスクレスブートクライアントのサポート (タスクマップ)」を参照してください。

リモートブートクライアントとディスクレスブートクライアントのサポート (タスクマップ)

DHCP サービスは、オペレーティングシステムのファイルを別のマシン (OS サーバー) からリモートでマウントする Oracle Solaris クライアントシステムをサポートしています。このようなクライアントを「ディスクレスクライアント」と呼びます。ディスクレスクライアントは、固定的なリモートブートシステムであるといえます。ディスクレスクライアントがブートされるたびに、このクライアントは、自身のオペレーティングシステムファイルを提供するサーバーの名前と IP アドレスを取得する必要があります。それによって、リモートのこれらのファイルからこのディスクレスクライアントをブートできるようになります。

各ディスクレスクライアントは、OS サーバー上に自分のルートパーティションを持っており、これらはクライアントのホスト名で共有されます。DHCP サーバーは、ディスクレスクライアントに常に同じ IP アドレスを返す必要があります。そのアドレスは、DNS などのネームサービスで常に同じホスト名にマップされていなければなりません。ディスクレスクライアントは、固定した IP アドレスを受け取ると、固定したホスト名を使用します。このクライアントは、OS サーバーにある自身の root パーティションにアクセスできます。

DHCP サーバーは、IP アドレスとホスト名のほかに、ディスクレスクライアントのオペレーティングシステムファイルの場所を提供できます。ただし、DHCP メッセージパケットでこの情報を渡すためのオプションとマクロを作成する必要があります。

次のタスクマップは、ディスクレスクライアントなどの固定的なリモートブートクライアントをサポートするのに必要なタスクを示しています。さらに、タスクマップには、タスクの実行に必要な手順へのリンクが含まれています。

タスク	説明	参照先
Oracle Solaris サーバーで OS サービスを設定します。	`smosservice` コマンドを使用して、クライアント用のオペレーティングシステムファイルを作成します。	『Oracle Solaris の管理: 基本管理』の第 7 章「ディスクレスクライアントの管理 (タスク)」 `smosservice(1M)` のマニュアルページも参照してください。
ネットワークブートクライアントをサポートするための DHCP サービスを設定します。	DHCP マネージャまたは `dhtadm` コマンドを使用して、ブート情報をクライアントに渡すために DHCP サーバーが使用できる新しいベンダーオプションとマクロを作成します。 ネットワークインストールクライアント用のオプションをすでに作成している場合は、ディスクレスクライアントのベンダークライアントタイプ用のマクロを作成するだけで十分です。	『Oracle Solaris 10 1/13 インストールガイド: ネットワークベースのインストール』の第 2 章「システム構成情報の事前構成 (タスク)」
ディスクレスクライアントへ予約済み IP アドレスを割り当てます。	DHCP マネージャを使用してアドレスを予約済みにするか、`pntadm` コマンドを使用してディスクレスクライアント用のアドレスを `MANUAL` にします。	405 ページの「予約済み IP アドレスを DHCP クライアントに割り当てる」
OS サービス用のディスクレスクライアントを設定します。	`smdiskless` コマンドを使用して、クライアントごとにオペレーティングシステムサポートを OS サーバーに追加します。クライアントごとに予約済みの IP アドレスを指定します。	『Oracle Solaris の管理: 基本管理』の第 7 章「ディスクレスクライアントの管理 (タスク)」 `smdiskless(1M)` のマニュアルページも参照してください。

情報だけを受け取るように DHCP クライアントを設定 (タスクマップ)

ネットワークによっては、DHCP サービスから構成情報だけをクライアントに提供する場合があります。情報 (リースではなく) を必要とするクライアントシステムは、DHCP クライアントから INFORM メッセージを送信できます。INFORM メッセージを受け取ると、DHCP サーバーは適切な構成情報をクライアントに送信します。

情報だけを必要とするクライアントをサポートするように DHCP サーバーを設定することができます。そのためには、クライアントをもつネットワークに対応する空のネットワークテーブルを作成する必要があります。それによって、DHCP サーバーはそのネットワークのクライアントに応答できます。

次のタスクマップは、情報のみクライアントのサポートに必要なタスクを示しています。さらに、タスクマップには、タスクの実行に必要な手順へのリンクが含まれています。

タスク	説明	参照先
空のネットワークテーブルを作成します。	DHCP マネージャか pntadm コマンドを使って、情報のみクライアントのネットワークに対応するネットワークテーブルを作成します。	381 ページの「DHCP ネットワークの追加」
クライアントが必要とする情報を含むマクロを作成します。	DHCP マネージャか dhtadm コマンドを使って、必要な情報をクライアントに渡すマクロを作成します。	415 ページの「DHCP マクロの作成」
DHCP クライアントから INFORM メッセージを発行します。	ifconfig *int* dhcp inform コマンドを使って、DHCP クライアントから INFORM メッセージを発行します。	448 ページの「DHCP クライアントの起動」 454 ページの「DHCP クライアントで使用する ifconfig コマンドオプション」 ifconfig(1M) のマニュアルページ

新しい DHCP データストアへの変換

DHCP には、DHCP 構成データをあるデータストアから別のデータストアに変換するユーティリティーがあります。このような新しいデータストアへの変換が必要になる場合はいくつかあります。たとえば、DHCP クライアントが増加し、DHCP サービスに対してより高いパフォーマンスや容量が求められる場合があります。あるいは、DHCP サーバーの負荷を複数のサーバーで分担したい場合があります。データストアの各タイプの相対的な利点と欠点の比較については、331 ページの「DHCP データストアの選択」を参照してください。

注 – Solaris 8 7/01 リリースより前の Oracle Solaris リリースからアップグレードした場合は、この注をよくお読みください。

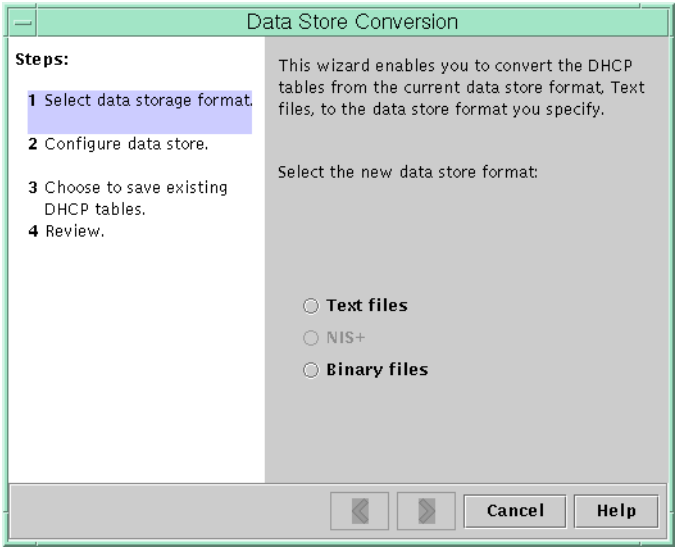
Oracle Solaris をインストールしたあとに DHCP ツールを実行すると、どのツールでも、新しいデータストアへの変換を促すメッセージが表示されます。Solaris 8 7/01 リリースでファイルと NIS+ の両方でデータストアのフォーマットが変更されているので、この変換は必須です。新しいデータストアへの移行が行われないと、DHCP サーバーは引き続き古いデータストアを使用します。ただし、サーバーができることは、既存のクライアントのリースを延長することだけです。古いデータテーブルを使用していると、新しい DHCP クライアントを登録したり、DHCP 管理ツールを使用したりすることはできません。

変換ユーティリティーは、Sun 提供のデータストアを サードパーティー製のデータストアに変換する際にも便利です。変換ユーティリティーは、既存のデータストアのエントリを調べて、同じデータを含む新しいエントリを新しいデータストアに追加します。データストアアクセスは、データストアごとに別個のモジュールとして実装されています。このモジュールアプローチのおかげで、変換ユーティリティーは、DHCP データを任意のデータストア形式から別のデータストア形式に変換できます。それぞれのデータストアには、DHCP サービスから使用できるモジュールが含まれていなければなりません。サードパーティー製のデータストアをサポートするモジュールを作成する方法については、『Solaris DHCP サービス開発ガイド』を参照してください。

データストアの変換は、DHCP マネージャのデータストア変換ウィザードまたは `dhcpconfig -C` コマンドで実行できます。

次に、データストア変換ウィザードの初期ダイアログボックスを示します。

図 15–19　DHCP マネージャの「データストア変換ウィザード (Data Store Conversion Wizard)」ダイアログボックス



変換を開始する前、古いデータストアのテーブル (dhcptab テーブルとネットワークテーブル) を保存するかどうかを指定する必要があります。次に、変換ユーティリティーは DHCP サーバーを停止し、データストアを変換し、変換が完了したあとに、サーバーを再起動します。古いテーブルを保存すると指定しない場合、変換が完了したあと、変換ユーティリティーは古いテーブルを削除します。変換の処理には、長い時間がかかることがあります。変換は背景で実行されますが、進捗状況を示す図が表示されます。

▼ DHCP データストアを変換する方法 (DHCP マネージャ)

1. **DHCP マネージャで、「サービス (Service)」メニューから「データストアの変換 (Convert Data Store)」を選択します。**

   358 ページの「DHCP マネージャを起動および停止する方法」を参照してください。

   データストア変換ウィザードが開きます。

2. ウィザードの質問に答えます。

   質問に対する回答がわからない場合は、「ヘルプ (Help)」をクリックすると、各ダイアログボックスについての詳細な情報を見ることができます。

3 選択内容を確認し、「完了 (Finish)」をクリックしてデータストアを変換します。

変換が完了すると、DHCP サーバーは再起動されます。サーバーは、直ちに新しいデータストアを使用します。

▼ DHCP データストアを変換する方法 (dhcpconfig -C)

1 スーパーユーザーになるか、DHCP 管理プロファイルに割り当てられている役割またはユーザー名になります。

DHCP 管理プロファイルの詳細については、359 ページの「DHCP コマンドへのユーザーアクセスの設定」を参照してください。

役割には、認証と特権コマンドが含まれます。役割の詳細については、『Solaris のシステム管理: セキュリティーサービス』の「RBAC の構成 (タスクマップ)」を参照してください。

2 次の書式のコマンドを入力してデータストアを変換します。

```
# /usr/sbin/dhcpconfig -C -r resource -p path
```

resource 新しいデータストアタイプ (SUNWbinfiles など)

path データへのパス (/var/dhcp など)

変換後も元のデータ (古いデータストア) を保存しておきたい場合は、-k オプションを指定してください。たとえば、自分のデータストアを SUNWbinfiles に変換し、古いデータストアを保存する場合は、次のように入力します。

```
# /usr/sbin/dhcpconfig -C -r SUNWbinfiles -p /var/dhcp -k
```

dhcpconfig ユーティリティーについての詳細は、dhcpconfig(1M) のマニュアルページを参照してください。

DHCP サーバー間での構成データの移動 (タスクマップ)

DHCP マネージャと dhcpconfig ユーティリティーを使って、DHCP 構成データの一部またはすべてを、ある DHCP サーバーから別のサーバーに移動できます。その場合には、ネットワーク全体と、ネットワークに関連するすべてのアドレス、マクロ、およびオプションを移動できます。あるいは、特定の IP アドレス、マクロ、およびオプションだけを選択して移動することも可能です。さらに、マクロやオプションを元のサーバーから削除せずに、コピーだけを行うこともできます。

データを移動するのは、次のような場合です。

- サーバーを追加して DHCP の処理を分担させる。
- DHCP サーバーのシステムを交換する。

- データストアへのパスを変更する (同じデータストアを使用したままで)。

次のタスクマップに、DHCP 構成データを移動する場合に実行する必要がある手順を示します。マップには、タスクの実行手順へのリンクが含まれています。

タスク	説明	参照先
1. 移動元のサーバーからデータをエクスポートします。	移動先のサーバーに移動するデータを選択し、それをエクスポートしたデータのファイルを作成します。	437 ページの「DHCP サーバーからデータをエクスポートする方法 (DHCP マネージャ)」 437 ページの「DHCP サーバーからデータをエクスポートする方法 (dhcpconfig -X)」
2. 移動先のサーバーにデータをインポートします。	エクスポートしたデータを移動先の DHCP サーバーのデータストアにコピーします。	439 ページの「DHCP サーバーにデータをインポートする方法 (DHCP マネージャ)」 439 ページの「DHCP サーバーにデータをインポートする方法 (dhcpconfig -I)」
3. インポートされたデータを新しいサーバー環境に合わせて変更します。	サーバー固有の構成データを新しいサーバーの情報に一致するように変更します。	440 ページの「インポートした DHCP データを変更する方法 (DHCP マネージャ)」 441 ページの「インポートした DHCP データを変更する方法 (pntadm、dhtadm)」

DHCP マネージャでは、「データをエクスポート (Export Data)」ウィザードと「データをインポート (Import Data)」ウィザードを使用して、データをあるサーバーから別のサーバーに移動します。そして、そのあとに「マクロ (Macros)」タブでマクロを変更します。次に、「データをエクスポート (Export Data)」ウィザードと「データをインポート (Import Data)」ウィザードの初期ダイアログボックスを示します。

図 15–20 DHCP マネージャの「データをエクスポート (Export Data)」ウィザードダイアログボックス

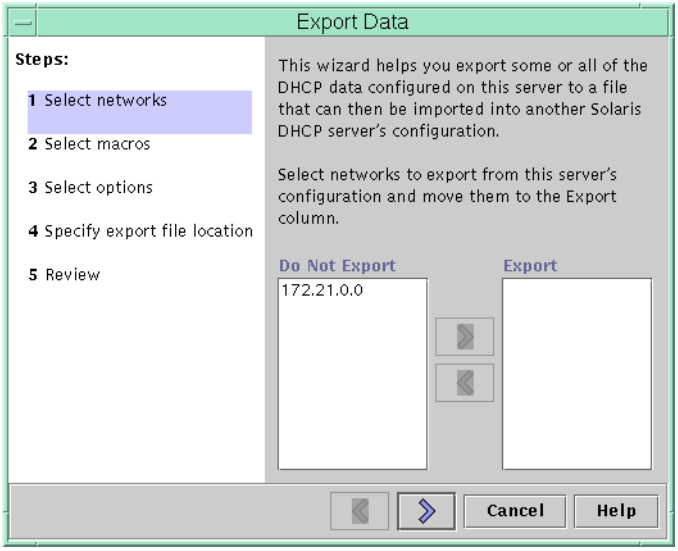
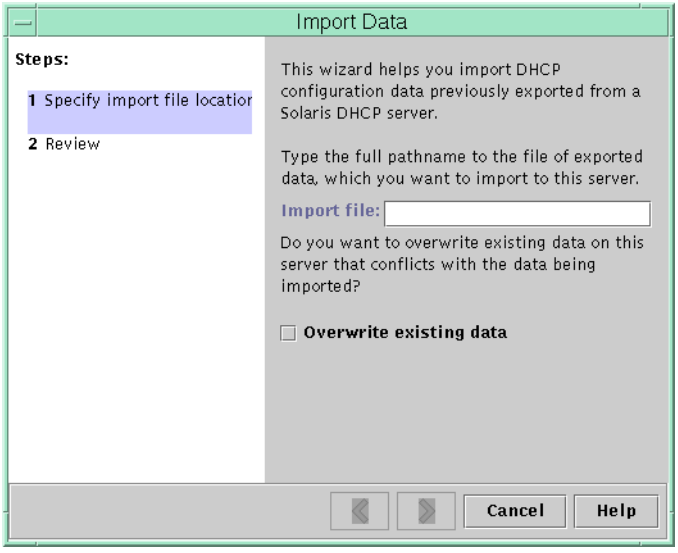


図 15–21 DHCP マネージャの「データをインポート (Import Data)」ウィザードダイアログボックス



▼ DHCP サーバーからデータをエクスポートする方法 (DHCP マネージャ)

1. データの移動 (またはコピー) 元のサーバー上で、**DHCP** マネージャを起動します。
   358 ページの「DHCP マネージャを起動および停止する方法」を参照してください。
2. 「サービス (**Service**)」メニューから「データをエクスポート (**Export Data**)」を選択します。
   「データをエクスポート (Export Data)」ウィザードが表示されます。図 15–20 を参照してください。
3. ウィザードの質問に答えます。
   質問に対する回答がわからない場合は、「ヘルプ (Help)」をクリックすると、質問についての詳細な情報を見ることができます。
4. エクスポートするデータが入ったファイルを、データをインポートする **DHCP** サーバーがアクセス可能なファイルシステムに移動します。

参照 データをインポートします (439 ページの「DHCP サーバーにデータをインポートする方法 (DHCP マネージャ)」を参照)。

▼ DHCP サーバーからデータをエクスポートする方法 (dhcpconfig -X)

1. データの移動 (またはコピー) 元のサーバーにログインします。
2. スーパーユーザーになるか、**DHCP** 管理プロファイルに割り当てられている役割またはユーザー名になります。
   DHCP 管理プロファイルの詳細については、359 ページの「DHCP コマンドへのユーザーアクセスの設定」を参照してください。
   役割には、認証と特権コマンドが含まれます。役割の詳細については、『Solaris のシステム管理: セキュリティーサービス』の「RBAC の構成 (タスクマップ)」を参照してください。

3 データをエクスポートします。

DHCP データ全体をエクスポートすることも、その特定部分だけをエクスポートすることもできます。

- 特定のアドレスやマクロ、オプションをエクスポートする場合は、次の書式のコマンドを入力します。

  # dhcpconfig -X *filename* **-a** *network-addresses* **-m** *macros* **-o** *options*

  filename には、エクスポートするデータを圧縮して格納するための完全パス名を指定します。エクスポートするネットワークアドレスや DHCP マクロ、DHCP オプションは、コンマで区切って指定する必要があります。次の図は、特定のネットワーク、マクロ、オプションをエクスポートする例です。

  ```
  # dhcpconfig -X /var/dhcp/0dhcp1065_data \ -a 10.63.0.0,10.62.0.0 \ -m
  10.63.0.0,10.62.0.0,SUNW.Sun-Blade-100 -o Sterm
  ```

- DHCP データ全体をエクスポートする場合は、コマンドに **ALL** キーワードを指定します。

  # dhcpconfig -X *filename* **-a ALL -m ALL -o ALL**

  filename には、エクスポートするデータを圧縮して格納するための完全パス名を指定します。キーワード ALL をコマンドオプションとともに使用すれば、すべてのネットワークアドレス、マクロ、またはオプションをエクスポートできます。次の図は、ALL キーワードを使用した例です。

  ```
  # dhcpconfig -X /var/dhcp/dhcp1065_data -a ALL -m ALL -o ALL
  ```

ヒント – 特定のタイプのデータをエクスポートから除外したい場合は、そのタイプに対する dhcpconfig コマンドオプションを省略します。たとえば、-m オプションを指定しなければ、DHCP マクロはエクスポートされません。

---

dhcpconfig コマンドについての詳細は、dhcpconfig(1M) のマニュアルページを参照してください。

4 エクスポートするファイルを、そのデータをインポートするサーバーからアクセス可能な場所に移動します。

参照 データをインポートします (439 ページの「DHCP サーバーにデータをインポートする方法 (dhcpconfig -I)」 を参照)。

▼ DHCP サーバーにデータをインポートする方法 (DHCP マネージャ)

1 **DHCP** サーバーからエクスポートしたデータの移動先サーバーで **DHCP** マネージャを起動します。

358 ページの「DHCP マネージャを起動および停止する方法」を参照してください。

2 「サービス **(Service)**」から「データをインポート **(Import Data)**」を選択します。

「データをインポート (Import Data)」ウィザードが表示されます。図 15–21 を参照してください。

3 ウィザードの質問に答えます。

質問に対する回答がわからない場合は、「ヘルプ (Help)」をクリックすると、質問についての詳細な情報を見ることができます。

4 必要なら、インポートされたデータを変更します。

詳細は、440 ページの「インポートした DHCP データを変更する方法 (DHCP マネージャ)」を参照してください。

▼ DHCP サーバーにデータをインポートする方法 (dhcpconfig -I)

1 データをインポートするサーバーにログインします。

2 スーパーユーザーになるか、**DHCP** 管理プロファイルに割り当てられている役割またはユーザー名になります。

DHCP 管理プロファイルの詳細については、359 ページの「DHCP コマンドへのユーザーアクセスの設定」を参照してください。

役割には、認証と特権コマンドが含まれます。役割の詳細については、『Solaris のシステム管理: セキュリティーサービス』の「RBAC の構成 (タスクマップ)」を参照してください。

3 次の書式のコマンドを入力してデータをインポートします。

```
# dhcpconfig -I filename
```

filename は、エクスポートされるデータを含むファイルの名前です。

4 必要なら、インポートされたデータを変更します。

詳細は、441 ページの「インポートした DHCP データを変更する方法 (pntadm、dhtadm)」を参照してください。

▼ インポートした DHCP データを変更する方法 (DHCP マネージャ)

1. データをインポートしたサーバーで **DHCP** マネージャを起動します。
   358 ページの「DHCP マネージャを起動および停止する方法」を参照してください。

2. インポートしたデータを調べて、変更する必要があるネットワーク固有情報を見つけます。
   ネットワークを移動した場合は、「アドレス (Addresses)」タブを開いて、移動 (インポート) したネットワーク内にあるアドレスの所有サーバーを変更する必要があります。また、「マクロ (Macros)」タブを開いて、一部のマクロの中にある NIS、NIS+、または DNS のドメイン名を変更する必要があります。

3. 「アドレス **(Addresses)**」タブを開いて、インポートしたネットワークを選択します。

4. すべてのアドレスを選択するには、最初のアドレスをクリックして、**Shift** キーを押したまま、最後のアドレスをクリックします。

5. 「編集 **(Edit)**」メニューから「属性 **(Properties)**」を選択します。
   「複数アドレスの変更 (Modify Multiple Addresses)」ダイアログボックスが開きます。

6. 「管理サーバー **(Managing Server)**」プロンプトで、新しいサーバーの名前を選択します。

7. 「構成マクロ **(Configuration Macro)**」プロンプトで、当該ネットワーク上にあるすべてのクライアントに使用されるマクロを選択し、「了解 **(OK)**」をクリックします。

8. 「マクロ **(Macros)**」タブを開きます。

9. 「**(Find)**」ボタンを使って、値の変更が必要と思われるオプションを見つけます。
   「(Find)」ボタンはウィザードの下端にあります。

   新しいサーバー上で変更する必要がありそうなオプションには、`DNSdmain`、`DNSserv`、`NISservs`、`NIS+serv`、`NISdmain` などがあります。

10. 該当するマクロのオプションを変更します。
    オプションの変更手順については、425 ページの「DHCP オプションの属性を変更する方法 (DHCP マネージャ)」を参照してください。

▼ インポートした DHCP データを変更する方法 (pntadm、dhtadm)

1. データをインポートしたサーバーにログインします。

2. スーパーユーザーになるか、DHCP 管理プロファイルに割り当てられている役割またはユーザー名になります。

   DHCP 管理プロファイルの詳細については、359 ページの「DHCP コマンドへのユーザーアクセスの設定」を参照してください。

   役割には、認証と特権コマンドが含まれます。役割の詳細については、『Solaris のシステム管理: セキュリティーサービス』の「RBAC の構成 (タスクマップ)」を参照してください。

3. ネットワークテーブルを調べて、変更する必要があるデータを見つけます。

   ネットワークを移動した場合は、`pntadm -P` *network-address* コマンドを使用して、移動したネットワークのネットワークテーブルを出力します。

4. **pntadm** コマンドを使って IP アドレス情報を変更します。

   インポートしたアドレス用の所有サーバーと構成マクロを変更する必要もあります。たとえば、アドレス `10.63.0.2` の所有サーバー (`10.60.3.4`) とマクロ (`dhcpsrv-1060`) を変更するには、次のコマンドを使用します。

   ```
   pntadm -M 10.63.0.2 -s 10.60.3.4 -m dhcpsrv-1060 10.60.0.0
   ```

   アドレスの数が多い場合は、個々のアドレスを変更する一連のコマンドを含むスクリプトファイルを作成します。`pntadm -B` コマンドでこのスクリプトを実行すると、`pntadm` がバッチモードで実行されます。詳細は、`pntadm`(1M) のマニュアルページを参照してください。

5. **dhcptab** マクロを調べて、値を変更する必要があるオプションを見つけます。

   `dhcptab` テーブル全体を画面に表示する場合は、`dhtadm -P` コマンドを使用します。変更するオプションや値を検索する場合は、`grep` などのツールを使用します。

6. マクロ中のオプションを変更する必要がある場合は、**dhtadm -M** コマンドを使って変更します。

   たとえば、マクロ中の NIS、NIS+、または DNS のドメイン名やサーバー名を変更する必要もあります。たとえば、次のコマンドは、マクロ `mymacro` 内にある `DNSdmain` と `DNSserv` の値を変更します。

   ```
   dhtadm -M -m mymacro -e 'DNSserv=dnssrv2:DNSdmain=example.net' -g
   ```

◆ ◆ ◆ 第 16 章

DHCP クライアントの構成と管理

この章では、Oracle Solaris に含まれている動的ホスト構成プロトコル (DHCP) クライアントについて説明します。この章では、クライアントの DHCPv4 プロトコルおよび DHCPv6 プロトコルの機能と、クライアントの動作の変更方法について説明します。

一方のプロトコル DHCPv4 は、かなり以前から Oracle Solaris に含まれています。これを使用すると、DHCP サーバーは IPv4 ネットワークアドレスなどの構成パラメータを IPv4 ノードに渡すことができます。

もう一方のプロトコル DHCPv6 を使用すると、DHCP サーバーは IPv6 ネットワークアドレスなどの構成パラメータを IPv6 ノードに渡すことができます。DHCPv6 は、「IPv6 ステートレスアドレスの自動構成」(RFC 2462) に対応するステートフルアドレス版であり、構成パラメータを取得するためにステートレスアドレスとは別に使用することも同時に使用することもできます。

この章では、次の内容について説明します。

DHCP クライアントについて

DHCP クライアントは dhcpagent デーモンです。Oracle Solaris をインストールしようとすると、DHCP を使ってネットワークインタフェースを構成するかどうかを確認するメッセージが表示されます。これに対して、DHCPv4 を使用するとして応答すると、Oracle Solaris のインストール中に、使用しているシステム上でそのプロトコルが使用可能になります。インストール時に DHCPv6 を特に指定するオプションはありません。ただし、IPv6 に関連する質問があります。IPv6 を有効にすると、DHCPv6 をサポートするローカルネットワーク上で DHCPv6 も有効になります。

DHCP を使用するために Oracle Solaris クライアントに対して必要な作業はこれだけです。DHCP サービスを使用する DHCP クライアントシステムにどのような情報が与えられるかは、DHCP サーバーの構成によります。

Oracle Solaris でクライアントシステムがすでに動作しており、DHCP を使用していない場合は、クライアントシステムを再構成すれば DHCP を使用できるようになります。さらに、DHCP クライアントシステムで DHCP の使用を止め、与えられた静的なネットワーク情報を使用したい場合にも、DHCP クライアントシステムを再構成できます。詳細は、452 ページの「DHCP クライアントを使用可能または使用不可にする」を参照してください。

DHCPv6 サーバー

Sun Microsystems は Oracle Solaris 用の DHCPv6 サーバーを提供していません。サードパーティーが提供しているサーバーには Sun の DHCPv6 との互換性があり、ネットワーク上に DHCPv6 サーバーが存在している場合、Sun の DHCPv6 クライアントはそれを使用します。

Sun の DHCPv4 サーバーについては、314 ページの「DHCP サーバー」を参照してください。

DHCPv4 と DHCPv6 の相違点

DHCPv4 と DHCPv6 の主な相違点は次の 2 つです。

- 管理モデル
  - DHCPv4 - 管理者が各インタフェースに対して DHCP を有効にします。管理は論理インタフェースごとに行われます。
  - DHCPv6 - 明示的な構成は必要ありません。このプロトコルは、特定の物理インタフェース上で有効にされます。
- プロトコルの詳細
  - DHCPv4 - DHCP サーバーが各アドレスのサブネットマスクを提供します。ホスト名オプションによってシステム全体のノード名が設定されます。

- DHCPv6 - サブネットマスクは、DHCPv6 サーバーではなくルーター広告によって提供されます。DHCPv6 のホスト名オプションはありません。

DHCP 管理モデル

「**DHCPv4**」では、クライアントを明示的に構成する必要があります。必要な場合はアドレス指定用の DHCPv4 システムを設定します。この設定は通常、システムを最初にインストールするときに行うか、`ifconfig(1M)`のオプションを使用して動的に行います。

「**DHCPv6**」では、クライアントを明示的に構成する必要はありません。DHCP の使用はネットワークの属性であり、DHCP を使用する指示は、ローカルルーターからのルーター広告メッセージで伝送されます。DHCP クライアントは、必要に応じて論理インタフェースを自動的に作成したり破棄したりします。

DHCPv6 メカニズムは、管理上、既存の IPv6 ステートレス (自動) アドレス構成によく似ています。ステートレスアドレス構成の場合は、ローカルルーターにフラグを設定することにより、一連の接頭辞に対して各クライアントが独自に自動的なアドレス生成を行うように指示します。このときクライアントは、通知された接頭辞に加え、ローカルインタフェースのトークンまたは乱数を使用します。DHCPv6 の場合は、同じ接頭辞が必要ですが、アドレスは「ランダムに」割り当てられるのではなく、DHCPv6 サーバーを介して取得され管理されます。

MAC アドレスとクライアント ID

「**DHCPv4**」では、アドレスを割り当てるためのクライアントの識別に、MAC アドレスおよび任意でクライアント ID が使用されます。ネットワークに入るたびに、同じクライアントは可能であれば同じアドレスを取得します。

「**DHCPv6**」でも基本的に同じスキームが使用されますが、クライアント ID は必須になり、それに基づく構造が義務付けられます。DHCPv6 のクライアント ID は、次の 2 つの部分で構成されます。 DUID (DHCP Unique Identifier) と IAID (Identity Association Identifier) です。DUID は (DHCPv4 の場合のようにインタフェースだけを識別するのではなく) クライアントの「システム」を識別し、IAID はそのシステム上のインタフェースを識別します。

RFC 3315 で説明されているように、サーバーとクライアントはアイデンティティーアソシエーション (IA) を使用して、関連する一連の IPv6 アドレスの識別、グループ化、および管理を行います。クライアントは、そのネットワークインタフェースそれぞれに個別の IA を少なくとも 1 つ関連付けてから、割り当てた IA を使用して、そのインタフェースの構成情報をサーバーから取得する必要があります。IA の詳細については、次の「プロトコルの詳細」のセクションを参照してください。

DUID+IAID は DHCPv4 でも使用できます。これらを互いに一義的に連結して、クライアント ID として使用できます。互換性の理由から、これは通常の IPv4 インタフェースでは行われません。ただし、クライアント ID が構成されていない場合、論理インタフェース (`hme0:1`) には DUID+IAID が使用されます。

IPv4 DHCP とは異なり、DHCPv6 には「クライアント名」オプションがないため、DHCPv6 だけに基づく名前をシステムに付けることはできません。その代わり、DHCPv6 から提供されるアドレスに対応する DNS 名を知る必要がある場合は、DNS 逆解決 (`getaddrinfo(3SOCKET)` 関数を使用したアドレスからの名前照会) を使用します。したがって、DHCPv6 だけを使用している場合にノードに特定の名前を与えるには、システムに `/etc/nodename` を設定する必要があります。

プロトコルの詳細

DHCPv4 では、割り当てられたアドレスに使用すべきサブネットマスクは、DHCP サーバーによって指定されます。DHCPv6 では、サブネットマスク (「接頭辞長」とも呼ばれる) は DHCP サーバーによって制御されるのではなく、ルーター広告によって割り当てられます。

DHCPv4 には「ホスト名」オプションがあり、これを使用してシステム全体のノード名が設定されます。DHCPv6 にはそのようなオプションはありません。

DHCPv6 のクライアント ID を構成するには、システムで自動的に選択させる代わりに、DUID を指定する必要があります。この設定は、デーモンに対してグローバルに行うか、インタフェースごとに行うことができます。グローバルな DUID を設定するには、次の書式を使用します (先頭にドットを付ける)。

`.v6.CLIENT_ID=`*DUID*

特定のインタフェースが特定の DUID を使用するように設定して、システムが DHCPv6 サーバーに対して複数の独立したクライアントに見えるようにするには、次のように指定します。

`hme0.v6.CLIENT ID=`*DUID*

各アイデンティティーアソシエーション (IA) は、1 種類のアドレスを保持します。たとえば、一時アドレス用アイデンティティーアソシエーション (IA_TA) は一時アドレスを保持し、非一時アドレス用アイデンティティーアソシエーション (IA_NA) は割り当てられた永続的なアドレスを保持します。このドキュメントで説明する DHCPv6 のバージョンでは、IA_NA アソシエーションだけが提供されています。

Oracle Solaris は、要求に応じて各インタフェースに 1 つの IAID を割り当てます。この IAID はルートファイルシステム内のファイルに格納され、マシンの寿命にわたって保持されます。

論理インタフェース

DHCPv4 クライアントの論理インタフェースは、それぞれが独立した管理単位です。0 番目の論理インタフェース (デフォルトで識別子がインタフェースの MAC アドレスになる) に加え、ユーザーは特定の論理インタフェースを構成して DHCP を実行することができます。そのためには、`dhcpagent` 構成ファイルに CLIENT_ID を指定します。次に例を示します。

```
hme0:1.CLIENT_ID=orangutan
```

DHCPv6 の動作は異なります。IPv4 とは異なり、IPv6 インタフェースの 0 番目の論理インタフェースは常にリンクローカルです。リンクローカルは、DHCP サーバーなどのほかの割り当て方法が利用できない場合に、IP ネットワーク内のデバイスに IP アドレスを自動的に割り当てるために使用されます。0 番目の論理インタフェースは、DHCP の制御下に置くことはできません。そのため、DHCPv6 は 0 番目の論理インタフェース (「物理インタフェース」とも呼ばれる) 上で実行されるにもかかわらず、0 番目以外の論理インタフェースだけにアドレスを割り当てます。

DHCPv6 サーバーは、DHCPv6 クライアント要求に応答して、クライアントで構成すべきアドレスのリストを返します。

オプションのネゴシエーション

DHCPv6 には「オプション要求」オプションがあり、クライアントがどの情報を優先的に望んでいるかについて、サーバーにヒントを提供します。使用可能なすべてのオプションをサーバーからクライアントに送信すると、送信される情報が大量になり、クライアントに到達するまでにその一部をドロップする必要が生じる可能性があります。サーバーはヒントを使用して、応答に含めるオプションを選択することができます。あるいは、サーバーはヒントを無視し、ほかの項目を選択して含めることもできます。たとえば、Oracle Solaris の場合、優先するオプションには Oracle Solaris DNS アドレスドメインや NIS アドレスドメインなどが含まれる可能性がありますが、NetBIOS サーバーが含まれる可能性はわずかです。

同じ種類のヒントが DHCPv4 にも用意されていますが、この特別な「オプション要求」オプションはありません。代わりに、DHCPv4 では、`/etc/default/dhcpagent` の `PARAM_REQUEST_LIST` が使用されます。

構成の構文

DHCPv6 クライアントを構成するには、既存の DHCPv4 クライアントの場合とほぼ同様に、`/etc/default/dhcpagent` を使用します。

構文は、インタフェース名 (存在する場合) と構成対象のパラメータの間に挿入される「.v6」マーカーで拡張されます。たとえば、グローバルな IPv4 オプション要求リストは、次のように設定されます。

```
PARAM_REQUEST_LIST=1,3,6,12,15,28,43
```

特定のインタフェースでホスト名オプションを省略するには、次のように構成します。

```
hme0.PARAM_REQUEST_LIST=1,3,6,15,28,43
```

DHCPv6 のグローバルな要求リストを設定する場合は、先頭にドットを付加します。

```
.v6.PARAM_REQUEST_LIST=23,24
```

特定のインタフェースを設定する場合は、次の例に従います。

```
hme0.v6.PARAM_REQUEST_LIST=21,22,23,24
```

参考として、DHCPv6 構成の実際の `/etc/default/dhcpagent` ファイルを次に示します。

```
# The default DHCPv6 parameter request list has preference (7), unicast (12),
# DNS addresses (23), DNS search list (24), NIS addresses (27), and
# NIS domain (29).  This may be changed by altering the following parameter-
# value pair.  The numbers correspond to the values defined in RFC 3315 and
# the IANA dhcpv6-parameters registry.
.v6.PARAM_REQUEST_LIST=7,12,23,24,27,29
```

DHCP クライアントの起動

ほとんどの場合、DHCPv6 クライアントを起動するために操作は必要ありません。`in.ndpd` デーモンが必要に応じて DHCPv6 を自動的に起動します。ブート時に IPv6 用に plumb されるようにインタフェースを構成するには、`/etc/hostname6.$IFNAME` の変更が必要となる場合があります。ただし、インストール時にシステムで IPv6 を有効にした場合、これはインストーラによってすでに実行されています。

これに対し、DHCPv4 では、Oracle Solaris のインストール時にクライアントの起動を要求しなかった場合は、これを要求する必要があります。452 ページの「DHCP クライアントを有効にする方法」を参照してください。

`dhcpagent` デーモンは、システムのブートに関与するほかのプロセスに必要な構成情報を取得します。そのため、システム起動スクリプトは、ブートプロセスの初期段階に `dhcpagent` を起動し、DHCP サーバーからネットワーク構成情報が到着するのを待ちます。

デフォルトでは DHCPv6 が実行されますが、DHCPv6 を実行しないように選択することもできます。DHCPv6 の実行開始後は、`ifconfig` コマンドで停止することができます。`/etc/inet/ndpd.conf` ファイルを変更して DHCPv6 を無効にし、リブート時に DHCPv6 が起動しないようにすることもできます。

次に、名前が hme0 のインタフェースで DHCPv6 を即時にシャットダウンする方法の例を示します。

```
ex# echo ifdefault StatefulAddrConf false >> /etc/inet/ndpd.conf
ex# pkill -HUP -x in.ndpd
ex# ifconfig hme0 inet6 dhcp release
```

/etc/dhcp.*interface* ファイル (たとえば、Sun Fire 880 システム上の /etc/dhcp.ce0) が存在していると、起動スクリプトは、指定されたインタフェース上で DHCPv4 が使用されるものとみなします。起動スクリプトは、dhcp.*interface* ファイルを見つけると、dhcpagent を起動します。

起動された dhcpagent は、ネットワークインタフェースの構成を行う指示を受信するまで待機します。そのあと、起動スクリプトは、ifconfig *interface* dhcp start コマンドを発行して、dhcpagent に DHCPv4 の起動を指示します (311 ページの「DHCP の動作」を参照)。dhcp.*interface* ファイルにコマンドが含まれている場合は、それらのコマンドが ifconfig の dhcp start オプションに付加されます。ifconfig *interface* dhcp コマンドのオプションについては、ifconfig(1M) のマニュアルページを参照してください。

DHCPv6 通信

DHCPv4 は手動構成によって起動されるのに対し、DHCPv6 はルーター広告 (RA) によって起動されます。ルーターの構成に応じて、システムはルーター広告メッセージが受信されたインタフェースで DHCPv6 を自動的に起動し、DHCP を使用してアドレスとほかのパラメータを取得するか、DHCPv6 でアドレス以外のデータ (DNS サーバーなど) だけを要求します。

in.ndpd デーモンはルーター広告メッセージを受信します。これは、システムで IPv6 用に plumb されているすべてのインタフェースで、自動的に実行されます。in.ndpd は、DHCPv6 を実行するように指定する RA を検出すると、DHCPv6 を起動します。

in.ndpd が DHCPv6 を起動しないようにするには、/etc/inet/ndpd.conf ファイルを変更します。

次のいずれかの ifconfig コマンドを使用して、DHCPv6 の起動後に DHCPv6 を停止することもできます。

```
ifconfig <interface>inet6 dhcp drop
```

または

```
ifconfig <interface> inet6 dhcp release
```

DHCP クライアントプロトコルはネットワーク構成情報をどのように管理するか

DHCPv4 クライアントプロトコルと DHCPv6 クライアントプロトコルでは、ネットワーク構成情報の管理方法が異なります。主な相違点は、DHCPv4 では単一のアドレスのリースとそれに関連するいくつかのオプションのためにネゴシエーションが行われ、DHCPv6 では一連のアドレスとオプションに対して一括でネゴシエーションが行われることです。

DHCPv4 クライアントとサーバー間の対話の概要については、第 12 章「DHCP について (概要)」を参照してください。

DHCPv4 クライアントはネットワーク構成情報をどのように管理するか

DHCP サーバーから情報パケットを取得すると、`dhcpagent` はネットワークインタフェースを構成し、使用可能にします。デーモンは、そのインタフェースを IP アドレスのリース期間が終わるまで制御し、その構成データを内部テーブルに保持します。システム起動スクリプトは `dhcpinfo` コマンドを使用して内部テーブルから構成オプションの値を抽出します。それらの値は、システムを構成し、システムがネットワーク上で通信できるようにするために使用されます。

`dhcpagent` デーモンは、一定時間 (通常はリース期間の半分) が過ぎるまで何もせずに待機します。この時間が過ぎると、デーモンは、リースの延長を DHCP サーバーに要求します。`dhcpagent` デーモンは、インタフェースが停止していたり、IP アドレスが変更されていることをシステムから通知されると、`ifconfig` コマンドから指示があるまでそのインタフェースの制御を行いません。また、`dhcpagent` は、インタフェースが適切に動作し、IP アドレスが変更されていないことを検出すると、リースの更新要求をサーバーに送信します。リースを更新できない場合、`dhcpagent` はリース期間の満了時にそのインタフェースを停止します。

`dhcpagent` は、リースに関連する活動を行うたびに、`/etc/dhcp/eventhook` という実行可能ファイルを探します。この名前の実行可能ファイルが見つかると、`dhcpagent` はこのファイルを起動します。イベント実行可能ファイルの使用については、464 ページの「DHCP クライアントのイベントスクリプト」を参照してください。

DHCPv6 クライアントはネットワーク構成情報をどのように管理するか

クライアントとサーバーの間の DHCPv6 通信は、クライアントがサーバーを見つけるために要請メッセージを送信することによって開始されます。応答として、DHCP サービスに使用可能なすべてのサーバーが通知メッセージを送信します。サーバーのメッセージには、複数の IA_NA (非一時アドレス用アイデンティティーアソシエーション) レコードに加え、サーバーが提供できるほかのオプション (DNS サーバーアドレスなど) が含まれています。

クライアントは、独自の IA_NA/IAADDR レコードを要求メッセージに設定することにより、特定のアドレス (またはその複数) を要求できます。通常、クライアントが特定のアドレスを要求するのは、古いアドレスが記録されており、可能な限り同じアドレスがサーバーから提供されることを望む場合です。クライアントの動作にかかわらず (クライアントがまったくアドレスを要求しない場合でも)、サーバーは 1 つの DHCPv6 トランザクション用に任意の数のアドレスをクライアントに提供することができます。

クライアントとサーバーの間で行われるメッセージのやり取りは次のとおりです。

- クライアントがサーバーを見つけるために要請メッセージを送信します。
- サーバーは通知メッセージを送信して、DHCP サービスに使用可能であることを示します。
- クライアントは要求メッセージを送信して、もっとも大きい優先値を持つサーバーに、IP アドレスなどの構成パラメータを要求します。サーバーの優先値は、最低値 0 から最高値 255 の範囲で、管理者によって設定されます。
- サーバーは、アドレスリースと構成データを含む応答メッセージを送信します。

通知メッセージ内の優先値が 255 であれば、DHCPv6 クライアントはただちにそのサーバーを選択します。もっとも優先値の高いサーバーが応答しない場合や要求メッセージに正常な応答を返すことができない場合、クライアントは、取得済みの通知メッセージの中で優先値の高いものから順にサーバーを検索します。すべての通知メッセージの検索が終わると、クライアントは再び要請メッセージを送信して処理を繰り返します。

選択されたサーバーは、要請メッセージまたは要求メッセージへの応答として、割り当てるアドレスと構成パラメータを含む応答メッセージを送信します。

DHCP クライアントのシャットダウン

クライアントはシャットダウン時に、クライアントにアドレスを割り当てたサーバーに解放メッセージを送信して、割り当てられたアドレスの 1 つ以上をクライアントが使用しなくなることを示します。DHCPv4 クライアントシステムが正常にシャットダウンするとき、`dhcpagent` は現在の構成情報をファイルに書き込みます (ファイルが存在する場合)。ファイル名は、DHCPv4 の場合は `/etc/dhcp/`*interface*`.dhc` で、DHCPv6 の場合は `/etc/dhcp/`*interface*`.dh6` です。デフォルトでは、リースは解放されずに保存されるため、IP アドレスが使用されなくなったことを DHCP サーバーは検出できません。そのため、クライアントは次のブート時にそのアドレスを簡単に再取得できます。このデフォルトの動作は、`ifconfig <interface> dhcp drop` コマンドと同じです。

システムのリブート時にそのファイル内のリースが依然として有効であると、`dhcpagent` は、同じ IP アドレスとネットワーク構成情報を使用する (短縮形の) 要求を送信します。DHCPv4 の場合、これは要求メッセージです。DHCPv6 の場合、これは確認メッセージです。

DHCP サーバーがこれを許可した場合、dhcpagent はシステムのシャットダウン時にディスクに書き込んだ情報を使用できます。クライアントがこの情報を使用することをサーバーが許可しなかった場合、dhcpagent は DHCP プロトコルシーケンスを開始します (311 ページの「DHCP の動作」を参照)。その結果、クライアントは、新しいネットワーク構成情報を取得します。

DHCP クライアントを使用可能または使用不可にする

Oracle Solaris はすでに動作している、DHCP がまだ使用されていないサーバーで DHCP クライアントを使用可能にするためには、まず、システムを構成解除する必要があります。システムがブートされたら、ある一連のコマンドを実行してシステムを設定し、DHCP クライアントを使用可能にします。

注 – 多くの配備では、インフラストラクチャーの重要な部分には DHCP を使用せずに静的 IP アドレスを設定することが一般的です。ネットワーク上のルーターや特定のサーバーなど、クライアントになるべきデバイスとそうでないデバイスの判定については、このドキュメントでは説明しません。

▼ DHCP クライアントを有効にする方法

この手順が必要なのは、Oracle Solaris のインストール時に DHCPv4 が使用可能にされていない場合だけです。DHCPv6 の場合、この手順は不要です。

1. クライアントシステムでスーパーユーザーになります。

2. このシステムで事前構成 (対話型構成ではなく) を使用している場合は、**sysidcfg** ファイルを編集します。**sysidcfg** ファイル内の **network_interface** キーワードに **dhcp** サブキーを追加します。

   たとえば、network_interface=hme0 {dhcp} のように指定します。詳細は、sysidcfg(4) のマニュアルページを参照してください。

3. システムを構成解除し、シャットダウンします。

   ```
   # sys-unconfig
   ```

   このコマンドで削除される構成情報についての詳細は、sys-unconfig(1M) のマニュアルページを参照してください。

4. シャットダウンが完了したら、システムをリブートします。

   事前構成を使用している場合、システムはブート時に、sysidcfg ファイルの dhcp サブキーによって、DHCP クライアントを使用するように構成されます。

事前構成を使用していない場合には、システムのリブート時に、システム構成情報の入力を sysidtool プログラムから求められます。詳細は、sysidtool(1M) のマニュアルページを参照してください。

5 **DHCP** を使用してネットワークインタフェースを構成するようにプロンプトが表示されたら、**Yes** を選択します。

▼ DHCP クライアントを無効にする方法

1 クライアントシステムでスーパーユーザーになります。

2 **sysidcfg** ファイルを使用してシステムを事前構成した場合には、**dhcp** サブキーを **network_interface** キーワードから削除します。

3 システムを構成解除し、シャットダウンします。

```
# sys-unconfig
```

このコマンドで削除される構成情報についての詳細は、sys-unconfig(1M) のマニュアルページを参照してください。

4 シャットダウンが完了したら、システムをリブートします。

システムで事前構成を使用している場合には、構成情報を求めるメッセージは表示されず、DHCP クライアントは構成されません。

事前構成を使用していない場合には、システムのリブート時に、システム構成情報の入力を sysidtool プログラムから求められます。詳細は、sysidtool(1M) のマニュアルページを参照してください。

5 **DHCP** を使用してネットワークインタフェースを構成するように要求するプロンプトが表示されたら、**No** を選択します。

DHCP クライアントの管理

通常のシステムオペレーションのもとでは、DHCP クライアントソフトウェアは管理を必要としません。dhcpagent デーモンはシステムブート時に自動的に起動し、リースについてサーバーとネゴシエーションを行い、シャットダウン時に停止します。dhcpagent デーモンを手動で直接、起動または停止しないようにしてください。代わりに、クライアントシステム上のスーパーユーザーであれば、必要に応じて ifconfig コマンドを使い、dhcpagent によるネットワークインタフェースの管理を変更できます。

DHCP クライアントで使用する ifconfig コマンドオプション

この節では、ifconfig(1M) のマニュアルページに記載されているコマンドオプションについてまとめます。これらのコマンドの DHCPv4 バージョンと DHCPv6 バージョンでは、「inet6」キーワードだけが異なります。DHCPv6 の場合は「inet6」キーワードを使用し、DHCPv4 の場合は省略してください。

ifconfig コマンドでは、次のことができます。

- **DHCP** クライアントの起動 – ifconfig *interface* [inet6] dhcp start コマンドは、dhcpagent と DHCP サーバー間の対話を開始して、IP アドレスと新しい構成オプションのセットを取得します。このコマンドは、IP アドレスを追加したり、サブネットマスクを変更する場合など、情報を変更してそれをクライアントですぐに使用したいときに便利です。
- ネットワーク構成情報だけの要求 – ifconfig *interface* [inet6] dhcp inform コマンドを実行すると、dhcpagent は、IP アドレス以外のネットワーク構成パラメータを求める要求を発行します。このコマンドは、ネットワークインタフェースが静的 IP アドレスを持っているが、クライアントシステムが更新されたネットワークオプションを必要としているような場合に便利です。たとえば、DHCP を IP アドレスの管理には使用しないが、ネットワーク上のホストの構成には使用したいような場合です。
- リース延長の要求 – ifconfig *interface* [inet6] dhcp extendipadm refresh-addr *dhcp-addrobj* コマンドを実行すると、 dhcpagent はリースをリフレッシュするリクエストを発行します。クライアントは、リースの延長を自動的に要求します。ただし、リース期間を変更し、次のリース更新を待たずにクライアントで新しいリース期間をただちに使用したい場合は、このコマンドを使用できます。
- **IP** アドレスの解放 – ifconfig *interface* [inet6] dhcp release コマンドを実行すると、dhcpagent はネットワークインタフェースで使用されている IP アドレスを解放します。IP アドレスの解放は、リースの期限が切れると自動的に行われます。たとえば、ラップトップをネットワークから切り離し、別のネットワーク上で起動する予定の場合に、このコマンドを実行することをお勧めします。/etc/default/dhcpagent 構成ファイルの RELEASE_ON_SIGTERM プロパティーも参照してください。
- **IP** アドレスの中断 – ifconfig *interface* [inet6] dhcp drop コマンドを実行すると、dhcpagent は、DHCPサーバーに通知せずに、ネットワークインタフェースを停止し、リースをファイルシステムにキャッシュします。この処理により、クライアントは次回リブート時に同じ IP アドレスを使用できます。
- ネットワークインタフェースへの **ping** – ifconfig *interface* [inet6] dhcp ping は、インタフェースが DHCP の制御下にあるかどうかを示します。
- ネットワークインタフェースの **DHCP** 構成状態の表示 – ifconfig *interface* [inet6] dhcp status コマンドは、DHCP クライアントの現在の状態を表示します。この表示には、次の情報が含まれています。

- クライアントに IP アドレスがバインドされているかどうか
- 送信、受信、および拒否された要求の数
- このインタフェースが一時インタフェースかどうか
- リースが取得された時刻、リースが期限切れになった時刻、リース更新の開始が予定されている時刻

次に例を示します。

```
# ifconfig hme0 dhcp status
Interface  State         Sent  Recv  Declined  Flags
hme0       BOUND            1     1         0   [PRIMARY]
(Began,Expires,Renew)=(08/16/2005 15:27, 08/18/2005 13:31, 08/17/2005 15:24)

# ifconfig hme0 inet6 dhcp status
Interface  State         Sent  Recv  Declined  Flags
hme0       BOUND            1     0         0   [PRIMARY]
(Began,Expires,Renew)=(11/22/2006 20:39, 11/22/2006 20:41, 11/22/2006 20:40)
```

DHCP クライアント構成パラメータの設定

クライアントシステムの `/etc/default/dhcpagent` ファイルには、`dhcpagent` に対する調整可能パラメータが含まれています。テキストエディタを使用して、クライアントの動作に影響を与えるパラメータを変更できます。`/etc/default/dhcpagent` ファイルには十分な説明が記載されていますので、詳細については、`dhcpagent(1M)` のマニュアルページだけでなく、このファイルも参照してください。

`/etc/dhcp.`*interface* ファイルは、DHCP クライアントに影響を及ぼすパラメータの設定が行われるもう 1 つの場所です。このファイルで設定されたパラメータは、システム起動スクリプトの `ifconfig` コマンドによって使用されます。ただし、これは DHCPv4 にのみ当てはまります。DHCPv6 には、これに相当するものはありません。

デフォルトで、DHCP クライアントは次のように構成されます。

DHCPv4 の場合

- クライアントシステムは特定のホスト名を必要としない。

  特定のホスト名をクライアントから要求する場合は、457 ページの「DHCPv4 クライアントのホスト名」を参照してください。
- クライアントのデフォルトの要求は `/etc/default/dhcpagent` で指定され、これには DNS サーバー、DNS ドメイン、ブロードキャストアドレスが含まれる。

  DHCP クライアントのパラメータファイルを適切に設定すれば、`/etc/default/dhcpagent` ファイルの `PARAM_REQUEST_LIST` キーワードでより多くのオプションを要求できます。さらに、DHCP サーバーを適切に構成すれば、特別に要求されているオプション以外のオプションを提供できます。DHCP

サーバーマクロを使用してクライアントに情報を送信することについては、dhcpd(8) のマニュアルページおよび408 ページの「DHCP マクロを使用した作業 (作業マップ)」を参照してください。

DHCPv4 および DHCPv6 の場合

- クライアントシステムは、1 つの物理ネットワークインタフェースで DHCP を使用する。

  複数の物理ネットワークインタフェースで DHCP を使用する場合は、456 ページの「複数のネットワークインタフェースを備えた DHCP クライアントシステム」を参照してください。
- Oracle Solaris のインストールよりあとに DHCP クライアントが構成されている場合、クライアントは自動的にネームサービスとして構成されるわけではない。

  DHCP クライアントでネームサービスを使用する場合は、459 ページの「DHCP クライアントシステムとネームサービス」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を備えた DHCP クライアントシステム

DHCP クライアントは、1 つのシステム上にあるいくつかの異なるインタフェースを同時に管理できます。インタフェースは、物理インタフェースでも論理インタフェースでもかまいません。個々のインタフェースは、独自の IP アドレスとリース時間をもっています。複数のネットワークインタフェースが DHCP 用に構成されていると、クライアントは個別の要求を出してそれらのインタフェースを構成します。クライアントは、インタフェースごとに別々のネットワーク構成パラメータ群を維持します。パラメータは別々に格納されますが、パラメータの中にはその性質上、広域的なものがあります。グローバルパラメータは、システム全体 (特定のネットワークインタフェースではなく) に適用されます。

グローバルパラメータには、ホスト名、NIS ドメイン名、時間帯などがあります。通常、グローバルパラメータの値はインタフェースごとに異なります。ただし、各システムに関連付けられたグローバルパラメータには、それぞれ 1 つの値だけを使用できます。グローバルパラメータの問い合わせに対して応答が 1 つだけ返されるようにするために、プライマリネットワークインタフェース用のパラメータだけが使用されます。プライマリインタフェースとして扱うインタフェースについては、そのインタフェースの `/etc/dhcp.`*interface* ファイルに `primary` という文字を挿入できます。`primary` というキーワードが使用されていないと、英字順で見た最初のインタフェースがプライマリインタフェースとみなされます。

DHCP クライアントは、論理インタフェースの場合も、物理インタフェースの場合も、そのリースを同じように管理します。ただし、論理インタフェースの場合は、次の制限があります。

- DHCP クライアントは、論理インタフェースに関連付けられたデフォルトルートを管理しません。

  Oracle Solaris カーネルは、ルートを物理インタフェース (論理インタフェースではなく) と関連付けます。通常は、物理インタフェースの IP アドレスが確立されると、必要なデフォルトルートがルーティングテーブルに入れられます。そのあと、 DHCP を使って、その物理インタフェースに関連付けられた論理インタフェースを構成した場合、通常、必要なルートはすでに決まっています。したがって、この論理インタフェースは同じルートを使用します。

  ある物理インタフェースのリースが期限切れになると、DHCP クライアントは、そのインタフェースに関連付けられているデフォルトルートを削除します。しかし、ある論理インタフェースのリースが期限切れになっても、DHCP クライアントは、その論理インタフェースに関連付けられているデフォルトルートを削除しません。対応する物理インタフェースは (場合によっては、そのほかの論理インタフェースも)、前と同じルートを使用する必要がある場合があります。

  DHCP 制御のインタフェースに関連付けられたデフォルトルートの追加や削除が必要な場合は、DHCP クライアントのイベントスクリプトメカニズムを使用できます。詳細は、464 ページの「DHCP クライアントのイベントスクリプト」を参照してください。

DHCPv4 クライアントのホスト名

デフォルトでは、DHCPv4 クライアントは、それ自身のホスト名を提供しません。DHCP サーバーがホスト名を提供するとみなすからです。デフォルトでは、DHCPv4 サーバーが、DHCPv4 クライアントにホスト名を提供するように構成されています。DHCPv4 クライアントとサーバーを一緒に使用する場合には、これらのデフォルト設定が有効に機能します。しかし、DHCPv4 クライアントを他社製の DHCP サーバーと一緒に使用する場合には、ホスト名がサーバーからクライアントに提供されないことがあります。ホスト名が DHCP を通して DHCP クライアントに提供されない場合、クライアントシステムは、ホスト名として使用する名前が `/etc/nodename` ファイルにあるか確認します。ホスト名がファイルにない場合は、`unknown` に設定されます。

DHCP サーバーが DHCP `Hostname` オプションで名前を提供している場合には、`/etc/nodename` ファイルの値が異なっていても、このホスト名が使用されます。クライアントで特定のホスト名を使用する場合は、その名前をクライアントから要求できます。次の手順を参照してください。

注 – 次の手順は、すべての DHCP サーバーで機能するとは限りません。この手順では、クライアントに対し、特定のホスト名を DHCP サーバーに送信して同じ名前を応答で受け取るように指示します。

ただし、DHCP サーバーはこの要求を尊重する必要はないため、尊重しないことがあります。その場合は、単に別の名前を返します。

▼ DHCPv4 クライアントが特定のホスト名を要求できるようにする方法

1 クライアントシステム上で、スーパーユーザーとして **/etc/default/dhcpagent** ファイルを編集します。

2 **/etc/default/dhcpagent** ファイルで **REQUEST_HOSTNAME** キーワードを見つけ、次のように変更します。

```
REQUEST_HOSTNAME=yes
```

REQUEST_HOSTNAME の前にコメント符号 (#) がある場合は、# を削除します。REQUEST_HOSTNAME キーワードがない場合は、これを挿入します。

3 クライアントシステム上で **/etc/hostname.***interface* ファイルを編集して、次の行を追加します。

inet *hostname*

hostname には、クライアントで使用する名前を指定します。

4 次のコマンドを入力し、クライアントはリブート時に完全な **DHCP** ネゴシエーションを実行します。

```
# ifconfig interface dhcp release
# reboot
```

クライアントにキャッシュされている DHCP データは削除されます。クライアントは、プロトコルを再開して、新しい構成情報 (新しいホスト名を含む) を要求します。DHCP サーバーは、そのホスト名がネットワークの別のシステムで使用されていないかをまず確認します。次にサーバーはホスト名をクライアントに割り当てます。DHCP サーバーは、ネームサービスの情報をクライアントのホスト名で更新できます。ただし、そのためには、DHCP サーバーがそのように構成されていなければなりません。

このホスト名をあとで変更する場合は、手順 3 から手順 4 までを繰り返します。

DHCP クライアントシステムとネームサービス

Oracle Solaris システムでは、ネームサービスとして、DNS、NIS、NIS+、および ローカルファイルストア (`/etc/inet/hosts`) がサポートされます。これらのネーム サービスを使用するためには、ある程度の事前構成が必要です。使用するネーム サービスを指定するために、ネームサービススイッチ構成ファイル (`nsswitch.conf(4)` を参照) も正しく設定してください。

ネームサービスのクライアントとしてシステムを構成しないと、DHCP クライアントシステムでネームサービスを使用することはできません。デフォルトでは、システムのインストール時に異なる構成を指定しないかぎり、ローカルファイルだけが使用されます。

次の表は、DHCP に関連する考慮事項をネームサービスごとに要約したものです。表には、クライアントで各ネームサービスを設定するときに役立つドキュメントへの相互参照が含まれています。

表 16–1　DHCP クライアントシステムに対するネームサービスクライアント設定情報

ネームサービス	クライアント設定情報

NIS	DHCP を使って Oracle Solaris ネットワークインストール情報をクライアントシステムに送信する場合には、`NISservs` と `NISdmain` オプションを含む構成マクロを使用できます。これらのオプションは、NIS サーバーの IP アドレスと、NIS ドメイン名をクライアントに渡すためのものです。これによって、クライアントは自動的に NIS クライアントになります。
	
	DHCP クライアントシステムで Oracle Solaris がすでに動作している場合、DHCP サーバーが NIS 情報をクライアントに送信しても、クライアントシステムが自動的に NIS クライアントとして構成されるわけではありません。
	
	DHCP クライアントシステムに NIS 情報を送信するように DHCP サーバーが構成されている場合には、クライアントで次の `dhcpinfo` コマンドを実行すれば、クライアントに渡された値を表示することができます。
	
	`# /usr/sbin/dhcpinfo NISdmain`
	
	`# /usr/sbin/dhcpinfo NISservs`
	
	注 – DHCPv6 の場合は、次のようにコマンドに `-v6` および異なるプロトコルキーワードを含めます。
	
	`# /usr/sbin/dhcpinfo -v6 NISDomain`
	
	`# /usr/sbin/dhcpinfo -v6 NISServers`
	
	NIS ドメイン名と NIS サーバーの値は、システムを NIS クライアントとして構成するときに使用します。
	
	DHCP クライアントシステム用の NIS クライアントを標準的な方法で設定します (『System Administration Guide: Naming and Directory Services (DNS, NIS, and LDAP)』の第 5 章「Setting Up and Configuring NIS Service」を参照)。
	
	ヒント – スクリプトを作成すれば、`dhcpinfo` や `ypinit` を使って、DHCP クライアントシステムにおける NIS クライアントの構成を自動的に行うことができます。

表 16–1 DHCP クライアントシステムに対するネームサービスクライアント設定情報 (続き)

ネームサービス	クライアント設定情報
NIS+	DHCP クライアントシステム用の NIS+ クライアントを従来の方法で設定した場合は、DHCP サーバーからクライアントに毎回異なるアドレスが割り当てられることがあります。NIS+ のセキュリティーには構成の一部として IP アドレスが含まれているため、これによってセキュリティーの問題が発生します。クライアントが毎回同じアドレスを確実に取得できるようにするには、DHCP クライアントシステム用の NIS+ クライアントを標準でない方法で設定します。その方法については、461 ページの「NIS+ クライアントとしての DHCP クライアントの設定」を参照してください。 DHCP クライアントシステムに手動で IP アドレスがすでに割り当てられている場合には、クライアントのアドレスは常に同じです。NIS+ クライアントは標準的な方法で設定できます (『System Administration Guide: Naming and Directory Services (NIS+)』の「Setting Up NIS+ Client Machines」を参照)。
`/etc/inet/hosts`	ネームサービスとして `/etc/inet/hosts` を使用する DHCP クライアントシステムには、`/etc/inet/hosts` ファイルを設定します。 DHCP クライアントシステム自身の `/etc/inet/hosts` ファイルには、そのホスト名が DHCP ツールによって追加されます。ただし、同じネットワークにあるほかのシステムの `/etc/inet/hosts` ファイルには、このホスト名を手動で追加する必要があります。さらに、DHCP サーバーシステムが名前を解決するために `/etc/inet/hosts` を使用する場合は、このシステムにもクライアントのホスト名を手動で追加する必要があります。
DNS	DHCP クライアントシステムが DNS ドメイン名を DHCP から取得する場合には、クライアントシステムの `/etc/resolv.conf` ファイルは自動的に構成されます。`/etc/nsswitch.conf` ファイルも自動的に更新され、検索順序でほかのネームサービスより後ろにある `hosts` 行に `dns` が付加されます。DNS の詳細については、『Solaris のシステム管理 (ネーミングとディレクトリサービス : DNS、NIS、LDAP 編)』を参照してください。

NIS+ クライアントとしての DHCP クライアントの設定

DHCP クライアントである Oracle Solaris システムで NIS+ ネームサービスを使用できます。ただし、DHCP サーバーが毎回異なるアドレスを提供する可能性があると、NIS+ のセキュリティー強化機能の 1 つである Data Encryption Standard (DES) 資格の作成が部分的に省略されます。セキュリティーのためには、常に同じアドレスを提供するように DHCP サーバーを構成してください。つまり、DHCP を使用しない NIS+ クライアントを設定する際に、クライアント固有の DES 資格を NIS+ サーバーに追加します。`nisclient` スクリプトや `nisaddcred` コマンドを使用するなど、資格を作成する方法はいくつかあります。

NIS+ 資格を生成するためには、クライアントが静的なホスト名をもち、資格の作成と格納ができなければなりません。NIS+ と DHCP を使用するクライアントを設定するときは、すべての DHCP クライアントのホスト名に使用できる同一の資格を作成する必要があります。この方法では、DHCP クライアントがどのような IP アドレスと、関連するホスト名を受け取っても、同じ DES 資格を使用できます。

次に、すべての DHCP ホスト名に使用できる同一の資格を作成する方法を示します。この手順は、DHCP クライアントで使用されるホスト名が分かる場合にのみ使用できます。たとえば、DHCP サーバーがそれらのホスト名を生成する場合、クライアントが受け取る可能性のあるホスト名が分かります。

▼ DHCP クライアントを NIS+ クライアントとして設定する方法

NIS+ クライアントになる DHCP クライアントシステムは、NIS+ ドメイン内にある別の NIS+ クライアントシステムに属する資格を使用する必要があります。この手順では、当該システムのみの資格が生成され、その資格は当該システムにログインしたスーパーユーザーだけに適用されます。DHCP クライアントシステムにログインするほかのユーザーには、NIS+ サーバー内で固有の独自の資格が必要です。これらの資格は、『Solaris のシステム管理 (ネーミングとディレクトリサービス : NIS+ 編)』に示されている手順に従って作成されます。

1 **NIS+ サーバーで次のコマンドを入力して、クライアントの資格を作成します。**

```
# nisgrep nisplus-client-name cred.org_dir > /tmp/file
```

このコマンドは、NIS+ クライアント用の `cred.org_dir` テーブルエントリを一時ファイルに書き込みます。

2 **cat コマンドで一時ファイルの内容を表示します。**

あるいは、テキストエディターを使用します。

3 **DHCP クライアント用に使用する資格をコピーします。**

公開鍵と非公開鍵をコピーする必要があります。両者とも、コロンで区切られた数字と文字からなる長い文字列です。この情報は、次のステップでコマンドに貼り付けられます。

4 **次のコマンドを入力して DHCP クライアント用の資格を追加します。**

```
# nistbladm -a cname=" dhcp-client-name@nisplus-domain" auth_type=DES \
auth_name="unix.dhcp-client-name@nisplus-domain" \
public_data=copied-public-key \
private_data=copied-private-key
```

copied-public-key には、一時ファイルからコピーした公開鍵情報を貼り付けます。*copied-private-key* には、一時ファイルからコピーした非公開鍵情報を貼り付けます。

5 **DHCP** クライアントシステムで次のコマンドを入力して、**NIS+** クライアントシステムから **DHCP** クライアントシステムにファイルをリモートコピーします。

```
# rcp nisplus-client-name:/var/nis/NIS_COLD_START /var/nis
# rcp nisplus-client-name:/etc/.rootkey /etc
# rcp nisplus-client-name:/etc/defaultdomain /etc
```

「permission denied (アクセスが拒否された)」というメッセージを受信した場合、システムはリモートコピーを許可するように設定されていない可能性があります。この場合には、通常のユーザーとしてこれらのファイルを中間の場所にコピーします。次に、スーパーユーザーとして、中間の場所にあるファイルを DHCP クライアントシステムの適切な場所にコピーします。

6 **DHCP** クライアントシステムで次のコマンドを入力して、**NIS+** 用の適切なネームサービススイッチファイルをコピーします。

```
# cp /etc/nsswitch.nisplus /etc/nsswitch.conf
```

7 **DHCP** クライアントシステムをリブートします。

これで、DHCP クライアントシステムは NIS+ サービスを使用できます。

例 16–1 DHCP クライアントシステムを NIS+ クライアントとして設定する方法

次の例では、`nisei` というシステムがあります。これは、NIS+ ドメイン `dev.example.net` の NIS+ クライアントです。さらに、`dhow` という DHCP クライアントシステムがあります。この例では `dhow` を NIS+ クライアントとして設定します。

```
    (First log in as superuser on the NIS+ server)
# nisgrep nisei cred.org_dir > /tmp/nisei-cred
# cat /tmp/nisei-cred
nisei.dev.example.net.:DES:unix.nisei@dev.example.net:46199279911a84045b8e0
c76822179138173a20edbd8eab4:90f2e2bb6ffe7e3547346dda624ec4c7f0fe1d5f37e21cff63830
c05bc1c724b
# nistbladm -a cname="dhow@dev.example.net." \
auth_type=DES auth_name="unix.dhow@dev.example.net" \
public_data=46199279911a84045b8e0c76822179138173a20edbd8eab4 \
private_data=90f2e2bb6ffe7e3547346dda624ec4c7f0fe1d5f37e21cff63830\
c05bc1c724b
# rlogin dhow
    (Log in as superuser on dhow)
# rcp nisei:/var/nis/NIS_COLD_START /var/nis
# rcp nisei:/etc/.rootkey /etc
# rcp nisei:/etc/defaultdomain /etc
# cp /etc/nsswitch.nisplus /etc/nsswitch.conf
# reboot
```

これで、DHCP クライアントシステム `dhow` は NIS+ サービスを使用できます。

例 16–2 スクリプトによる資格の追加

多数の DHCP クライアントシステムを NIS+ クライアントとして設定する場合は、スクリプトを作成できます。スクリプトを使えば、cred.org_dir NIS+ テーブルのエントリをすばやく追加できます。次に、サンプルスクリプトを示します。

```
#! /usr/bin/ksh
#
# Copyright (c) by Sun Microsystems, Inc. All rights reserved.
#
# Sample script for cloning a credential. Hosts file is already populated
# with entries of the form dhcp-[0-9][0-9][0-9]. The entry we're cloning
# is dhcp-001.
#
#
PUBLIC_DATA=6e72878d8dc095a8b5aea951733d6ea91b4ec59e136bd3b3
PRIVATE_DATA=3a86729b685e2b2320cd7e26d4f1519ee070a60620a93e48a8682c5031058df4
HOST="dhcp-"
DOMAIN="mydomain.example.com"

for
i in 002 003 004 005 006 007 008 009 010 011 012 013 014 015 016 017 018 019
do
     print - ${HOST}${i}
     #nistbladm -r [cname="${HOST}${i}.${DOMAIN}."]cred.org_dir
     nistbladm -a cname="${HOST}${i}.${DOMAIN}." \
         auth_type=DES auth_name="unix.${HOST}${i}@${DOMAIN}" \
         public_data=${PUBLIC_DATA} private_data=${PRIVATE_DTA} cred.org_Dir
done

exit 0
```

DHCP クライアントのイベントスクリプト

DHCP クライアントを適切に設定すれば、実行可能なプログラムやスクリプトを実行して、クライアントシステムに必要な任意のアクションを行うことができます。「イベントスクリプト」と呼ばれるこのプログラムやスクリプトは、一定の DHCP リースイベントが発生すると自動的に実行されます。イベントスクリプトを使用すれば、特定のリースイベントに応じてほかのコマンドやプログラム、スクリプトを実行できます。この機能を使用するためには、独自のイベントスクリプトを作成する必要があります。

dhcpagent では、DHCP リースイベントを表すために次のイベントキーワードが使用されます。

イベントキーワード	機能説明
BOUND と BOUND6	インタフェースが DHCP 用に構成されました。クライアントは、DHCP サーバーから肯定メッセージ (DHCPv4 ACK) または (DHCPv6 Reply) を受け取り、IP アドレスに対するリース要求を認められます。このイベントスクリプトは、インタフェースの構成が正常に終わると直ちに呼び出されます。

EXTEND と EXTEND6	クライアントによるリースの延長が成功しました。このイベントスクリプトは、クライアントが更新要求に対する肯定メッセージを DHCP サーバーから受け取ると直ちに呼び出されます。
EXPIRE と EXPIRE6	リース時間が終了すると、リースが期限切れになります。DHCPv4 の場合、このイベントスクリプトは、リースされたアドレスがインタフェースから削除され、インタフェースが停止状態にされる直前に呼び出されます。DHCPv6 の場合、このイベントスクリプトは、最後に残っているリースされたアドレスがインタフェースから削除される直前に呼び出されます。
DROP と DROP6	クライアントがインタフェースを DHCP 制御下から削除する目的でリースを中断しました。このイベントスクリプトは、インタフェースが DHCP 制御から削除される直前に呼び出されます。
RELEASE と RELEASE6	クライアントが IP アドレスを解放します。このイベントスクリプトは、クライアントがインタフェース上のアドレスを解放し、DHCPv4 `RELEASE` または DHCPv6 `Release` パケットを DHCP サーバーに送信する直前に呼び出されます。
INFORM と INFORM6	インタフェースは、DHCPv4 `INFORM` または DHCPv6 `Information-Request` メッセージを使用して、新しい構成情報または更新された構成情報を DHCP サーバーから取得します。これらのイベントは、DHCP クライアントがサーバーから構成パラメータだけを取得し、IP アドレスリースを取得しない場合に発生します。
LOSS6	リースが期限切れになったとき、有効なリースが 1 つ以上残っている場合は、期限切れのアドレスが削除される直前にこのイベントスクリプトが呼び出されます。削除されるアドレスは `IFF_DEPRECATED` フラグでマークされます。

これらのイベントが発生するたびに、`dhcpagent` は次のコマンドを呼び出します。

```
/etc/dhcp/eventhook interface event
```

ここで *interface* は DHCP を使用しているインタフェースを、*event* は前述のイベントキーワードの 1 つをそれぞれ表します。たとえば、最初に `ce0` インタフェースを DHCP 用に構成すると、`dhcpagent` は、イベントスクリプトを次のように呼び出します。

```
/etc/dhcp/eventhook net0 BOUND
```

イベントスクリプト機能を使用するためには、次のことを行う必要があります。

- 実行可能ファイルに /etc/dhcp/eventhook という名前を付けます。
- ファイルの所有者を root にする。
- アクセス権限を 755 (rwxr-xr-x) にします。
- 前述のイベントに応えて一連のアクションを行うスクリプトまたはプログラムを記述します。Sun は新しいイベントを追加する場合があるため、プログラムは、認識されないイベントや処理を必要としないイベントについては何もせずに無視する必要があります。たとえば、このプログラムまたはスクリプトでは、イベントが RELEASE の場合はログファイルに書き込み、それ以外のイベントは無視します。
- スクリプトやプログラムを非対話型にします。イベントスクリプトが呼び出される前に、stdin、stdout、stderr は /dev/null に接続されます。出力またはエラーを見るためには、ファイルにリダイレクトする必要があります。

イベントスクリプトは、そのプログラム環境を dhcpagent から継承し、root 特権で実行します。スクリプトでは、必要に応じて、dhcpinfo ユーティリティーを使ってより詳しいインタフェースの情報を取得できます。詳細は、dhcpinfo(1) のマニュアルページを参照してください。

dhcpagent デーモンは、イベントスクリプトがすべてのイベントに対して終了するまで待ちます。55 秒経ってもイベントスクリプトが終了しないと、dhcpagent は SIGTERM シグナルをスクリプトプロセスに送信します。さらに、追加の 3 秒が過ぎてもプロセスが終了しないと、デーモンは SIGKILL シグナルを送信してプロセスを強制的に終了させます。

dhcpagent(1M) のマニュアルページにイベントスクリプトの一例が示されています。

例 16–3 は、DHCP イベントスクリプトを使って /etc/resolv.conf ファイルの内容を最新の状態に保つ方法を示しています。BOUND や EXTEND イベントが発生すると、スクリプトは、ドメインサーバーとネームサーバーの名前を変更します。EXPIRE や DROP、RELEASE イベントが発生すると、スクリプトは、ドメインサーバーとネームサーバーの名前をファイルから削除します。

注 – このスクリプト例では、DHCP が、ドメインサーバー名およびネームサーバー名の正式な情報源であることを想定しています。さらに、DHCP 制御下のすべてのインタフェースが、一貫性のある最新の情報を返すことを想定しています。これらの前提は、実際のシステムの条件と一致しない場合があります。

例 16–3 /etc/resolv.conf ファイルを更新するイベントスクリプト

```
#!/bin/ksh -p

PATH=/bin:/sbin export PATH
```

例 16–3 /etc/resolv.conf ファイルを更新するイベントスクリプト (続き)

```
umask 0222

# Refresh the domain and name servers on /etc/resolv.conf

insert ()
{
    dnsservers=‘dhcpinfo -i $1 DNSserv‘
    if [ -n "$dnsservers" ]; then
        # remove the old domain and name servers
        if [ -f /etc/resolv.conf ]; then
            rm -f /tmp/resolv.conf.$$
            sed -e ’/^domain/d’ -e ’/^nameserver/d’ \
                /etc/resolv.conf > /tmp/resolv.conf.$$
        fi

        # add the new domain
        dnsdomain=‘dhcpinfo -i $1 DNSdmain‘
        if [ -n "$dnsdomain" ]; then
            echo "domain $dnsdomain" >> /tmp/resolv.conf.$$
        fi

        # add new name servers
        for name in $dnsservers; do
            echo nameserver $name >> /tmp/resolv.conf.$$
        done
        mv -f /tmp/resolv.conf.$$ /etc/resolv.conf
    fi
}

# Remove the domain and name servers from /etc/resolv.conf

remove ()
{
    if [ -f /etc/resolv.conf ]; then
        rm -f /tmp/resolv.conf.$$
        sed -e ’/^domain/d’ -e ’/^nameserver/d’ \
            /etc/resolv.conf > /tmp/resolv.conf.$$
        mv -f /tmp/resolv.conf.$$ /etc/resolv.conf
    fi
}

case $2 in
BOUND | EXTEND)
    insert $1
    exit 0
    ;;
EXPIRE | DROP | RELEASE)
    remove
    exit 0
    ;;
*)
    exit 0
    ;;
esac
```

◆ ◆ ◆ 第 17 章

17 DHCP のトラブルシューティング (リファレンス)

この章では、DHCP サーバーまたは DHCP クライアントを構成する際に発生する可能性がある問題点の解決に役立つ情報を提供します。さらに、構成が完了したあとで DHCP を使用しているときに起こる問題とその解決方法についても説明します。

この章の内容は次のとおりです。

DHCP サーバーの構成については、第 14 章「DHCP サービスの構成 (手順)」を参照してください。DHCP クライアントの構成については、452 ページの「DHCP クライアントを使用可能または使用不可にする」 を参照してください。

DHCP サーバーの問題の障害追跡

サーバーを構成する際に発生する問題は、次のカテゴリに分類されます。

NIS+ の問題と DHCP データストア

DHCP データストアとして NIS+ を使用する場合に起こる問題は、次のカテゴリに分けられます。

NIS+ を DHCP データストアとして選択できない

NIS+ をデータストアとして使用しようとしても、DHCP マネージャが NIS+ をデータストアの候補として認識しないことがあります。`dhcpconfig` コマンドを使用すると、NIS+ がインストールされておらず、動作していないという意味のメッセージが表示されることがあります。どちらの問題の場合も、このネットワークでは NIS+ が使用されている可能性はあるが、このサーバーには NIS+ が構成されていないことを意味します。NIS+ をデータとして選択するためには、サーバーマシンが NIS+ クライアントとして構成されている必要があります。

DHCP サーバーシステムを NIS+ クライアントとして設定する前に、次の要件が満たされていなければなりません。

- ドメインがすでに構成されている必要があります。
- NIS+ ドメインのマスターサーバーが動作している必要があります。
- マスターサーバーのテーブルにデータが格納されている必要があります。
- ホストテーブルに、新しいクライアントシステム (DHCP サーバーシステム) のエントリがすでに存在する必要があります。

NIS+ クライアントの構成に関する詳しい情報については、『System Administration Guide: Naming and Directory Services (NIS+)』の「Setting Up NIS+ Client Machines」を参照してください。

NIS+ が DHCP データストア用に適切に構成されていない

DHCP で NIS+ を使用できるようになったあとに NIS+ を変更すると、エラーになることがあります。このような変更が構成の問題を引き起こす可能性があるからです。問題と解決策の次の説明に従って、構成の問題の原因を判別してください。

問題: ルートオブジェクトが NIS+ ドメインに存在しない。

対処方法: 次のコマンドを入力します。

```
/usr/lib/nis/nisstat
```

このコマンドによって、ドメインの統計情報が表示されます。ルートオブジェクトが存在しない場合は、統計情報は表示されません。

『Solaris のシステム管理 (ネーミングとディレクトリサービス : NIS+ 編)』に従って NIS+ ドメインを設定します。

問題: `passwd` と `publickey` の情報について NIS+ が使用されていない。

対処方法: 次のコマンドを入力して、ネームサービススイッチの構成ファイルを表示します。

```
cat /etc/nsswitch.conf
```

この「nisplus」キーワードに関する passwd と publickey の項目を確認します。ネームサービススイッチの構成については、『Solaris のシステム管理 (ネーミングとディレクトリサービス : NIS+ 編)』を参照してください。

問題: ドメイン名が空である。

対処方法: 次のコマンドを入力します。

```
domainname
```

このコマンドによって空の文字列がリストされた場合は、このドメインについてドメイン名が設定されていません。データストアにローカルファイルを使用するか、あるいは、ネットワーク用に NIS+ ドメインを設定します。詳細は、『Solaris のシステム管理 (ネーミングとディレクトリサービス : NIS+ 編)』を参照してください。

問題: NIS_COLD_START ファイルが存在しない。

対処方法: サーバーシステムで次のコマンドを入力して、ファイルの存在を確認します。

```
cat /var/nis/NIS_COLD_START
```

データストアのローカルファイルを使用するか、あるいは、NIS+ クライアントを作成します。詳細は、『Solaris のシステム管理 (ネーミングとディレクトリサービス : NIS+ 編)』を参照してください。

DHCP データストアに対する NIS+ アクセス権の問題

NIS+ のアクセス権に問題があると、DES 資格が適切でない、またはアクセス権が不十分なため NIS+ オブジェクトやテーブルを更新できないというエラーメッセージが表示されることがあります。問題と解決策の次の説明に従って、NIS+ アクセス権のエラーの原因を判別します。

問題: NIS+ ドメイン内の org_dir オブジェクトへの作成アクセス権が DHCP サーバーシステムにない。

対処方法: 次のコマンドを入力します。

```
nisls -ld org_dir
```

アクセス権は r---rmcdrmcdr--- といった形式でリストされます。これらのアクセス権はそれぞれ、未認証、所有者、グループ、その他に対応しています。オブジェクトの所有者が次にリストされます。

通常は、所有者とグループの両方に、org_dir ディレクトリオブジェクトへの完全なアクセス権があります。完全な権限は、読み取り、変更、作成、破棄からなります。その他と未認証のクラスには、org_dir ディレクトリオブジェクトへの読み取りアクセス権だけがあります。

DHCP サーバー名は、org_dir オブジェクトの所有者か、グループ内の主体としてリストされていなければなりません。グループには作成アクセス権が必要です。次のコマンドでグループをリストします。

nisls -ldg org_dir

必要な場合は、nischmod コマンドを使って、org_dir に対するアクセス権を変更します。たとえば、グループに作成アクセス権を追加する場合は、次のコマンドを使用します。

nischmod g+c org_dir

詳細は、nischmod(1) のマニュアルページを参照してください。

問題: DHCP サーバーに、org_dir オブジェクトの下にテーブルを作成するアクセス権がない。

通常、この問題は、サーバーシステムの主体名が org_dir オブジェクトの所有グループのメンバーでないか、所有グループが存在しないことを意味します。

対処方法: 次のコマンドを入力して所有グループ名を検索します。

niscat -o org_dir

次のような行を見つけます。

```
Group : "admin.example.com."
```

次のコマンドを使ってグループ内の主体名をリストする

nisgrpadm -l *groupname*

たとえば、次のコマンドを実行すると、グループ admin.example.com の主体名が表示されます。

```
nisgrpadm -l admin.example.com
```

サーバーシステムの名前がグループの明示的なメンバーとしてリストされるか、グループの暗黙的なメンバーとして含まれているはずです。必要なら、nisgrpadm コマンドを使ってサーバーシステムの名前をグループに追加します。

たとえば、サーバー名 pacific をグループ admin.example.com に追加するには、次のように入力します。

```
nisgrpadm -a admin.example.com pacific.example.com
```

詳細は、nisgrpadm(1) のマニュアルページを参照してください。

問題: DHCP サーバーの NIS+ cred テーブルに有効な Data Encryption Standard (DES) 資格が存在しない。

対処方法: 資格の問題がある場合には、ユーザーが NIS+ ネームサービスに DES 資格を持っていないことを示すエラーメッセージが表示されます。

nisaddcred コマンドを使って、DHCP サーバーシステムのセキュリティー資格を追加します。

次の例では、ドメイン example.com にあるシステム mercury についての DES 資格を追加する方法を示します。 。

```
nisaddcred -p unix.mercury@example.com \
-P mercury.example.com. DES example.com.
```

このコマンドは、暗号化された秘密鍵の生成に必要なスーパーユーザーのパスワードを要求します。

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

DHCP における IP アドレスの割り当てエラー

クライアントが IP アドレスを取得または検証しようとすると、問題が syslog に記録されたり、サーバーデバッグモード出力に書き込まれることがあります。よく見られる次の各エラーメッセージが、考えられる原因と解決策を示しています。

`There is no` *n.n.n.n* `dhcp-network table for DHCP client's network`

原因: クライアントが、特定の IP アドレスを要求しているか、現在の IP アドレスのリースを延長しようとしています。DHCP サーバーは、DHCP ネットワークテーブルでそのアドレスを発見できません。

対処方法: DHCP ネットワークテーブルが誤って削除されている場合があります。DHCP マネージャか dhcpconfig コマンドを使ってネットワークを再び追加し、ネットワークテーブルを作り直すことができます。

`ICMP ECHO reply to OFFER candidate:` *n.n.n.n*`, disabling`

原因: DHCP クライアントに提供されようとしている IP アドレスがすでに使用されています。このアドレスを複数の DHCP サーバーで所有していると、この問題が起こることがあります。さらに、アドレスが DHCP 以外のネットワーククライアント用に手動で構成されている場合にも、この問題が起こることがあります。

対処方法: アドレスの正しい所有者を判別します。DHCP サーバーデータベースか、ホストのネットワーク構成を修正します。

`ICMP ECHO reply to OFFER candidate:` *n.n.n.n*`. No corresponding dhcp network record.`

原因: DHCP クライアントに提供されようとしている IP アドレスのレコードがネットワークテーブルにありません。このエラーは、IP アドレスが、選択されたあとに DHCP ネットワークテーブルレコードから削除されたことを示します。このエラーが起こるのは (もし起こるのなら)、アドレスの重複チェックが終了するまでの短い時間においてだけです。

対処方法: DHCP マネージャまたは `pntadm` コマンドを使って、DHCP ネットワークテーブルを表示します。その IP アドレスが存在しない場合は、DHCP マネージャの「アドレス (Address)」タブから「編集 (Edit)」メニューの「作成 (Create)」を選択してアドレスを作成します。さらに、`pntadm` 使ってこの IP アドレスを作成することもできます。

`DHCP network record for` *n.n.n.n* `is unavailable, ignoring request.`
原因: 要求された IP アドレスのレコードは DHCP ネットワークテーブルに存在しないので、サーバーが要求をドロップします。

対処方法: DHCP マネージャまたは `pntadm` コマンドを使って、DHCP ネットワークテーブルを表示します。その IP アドレスが存在しない場合は、DHCP マネージャの「アドレス (Address)」タブから「編集 (Edit)」メニューの「作成 (Create)」を選択してアドレスを作成します。さらに、`pntadm` 使ってこのアドレスを作成することもできます。

n.n.n.n `currently marked as unusable.`
原因: ネットワークテーブルで使用不可に指定されているため、要求された IP アドレスを提供できません。

対処方法: DHCP マネージャまたは `pntadm` コマンドを使って、アドレスを使用可能にします。

n.n.n.n `was manually allocated. No dynamic address will be allocated.`
原因: クライアントの ID は、手動で割り当てられたアドレスに割り当てられています。そのアドレスは使用不可に指定されています。そのため、サーバーがこのクライアントに別のアドレスを割り当てることができません。

対処方法: DHCP マネージャまたは `pntadm` コマンドを使って、そのアドレスを使用できるようにするか、そのクライアントに別のアドレスを手動で割り当てます。

`Manual allocation (`*n.n.n.n, client ID*`) has` *n* `other records. Should have 0.`
原因: 指定されたクライアント ID を持つクライアントに、複数の IP アドレスが手動で割り当てられています。クライアントには、1 つのアドレスだけが割り当てられているべきです。サーバーは、ネットワークテーブルにある、最後に手動で割り当てられたアドレスを選択します。

対処方法: DHCP マネージャまたは `pntadm` コマンドを使って、IP アドレスを修正し、余分な手動割り当てを取り除きます。

`No more IP addresses on` *n.n.n.n*`network.`
原因: 指定されたネットワーク上で DHCP が現在管理しているすべての IP アドレスは、すでに割り当てられています。

対処方法: DHCP マネージャまたは `pntadm` コマンドを使って、このネットワーク用に新しい IP アドレスを作成します。

`Client: clientid lease on n.n.n.n expired.`
原因: リースがネゴシエーション可能ではなく、有効期限が切れています。

対処方法: クライアントは、プロトコルを自動的に再起動して新しいリースを取得すべきです。

`Offer expired for client: n.n.n.n`
原因: サーバーがクライアントに IP アドレスを提供したが、クライアントの応答に時間がかかり過ぎ、このオファーは期限切れとなっています。

対処方法: クライアントは、新たな検索メッセージを自動的に発行すべきです。このメッセージも期限切れとなった場合は、DHCP サーバーのキャッシュオファータイムアウトを増加させます。DHCP マネージャでは、「サービス (Service)」メニューから「変更 (Modify)」を選択する

`Client: clientid REQUEST is missing requested IP option.`
原因: クライアントの要求が、提供された IP アドレスを指定しなかったため、DHCP サーバーはこの要求を無視しました。更新された DHCP プロトコル (RFC 2131) に準拠していない、サードパーティー製の DHCP クライアントを使用すると、この問題が起こることがあります。

対処方法: クライアントのソフトウェアを更新してください。

`Client: clientid is trying to renew n.n.n.n, an IP address it has not leased.`
原因: DHCP ネットワークテーブルにあるこのクライアント用の IP アドレスが、クライアントが更新要求で指定した IP アドレスと一致しません。DHCP サーバーはこのリースを更新しません。この問題は、クライアントがまだ IP アドレスを使用しているのに、クライアントのレコードを削除した場合に発生することがあります。

対処方法: DHCP マネージャまたは `pntadm` コマンドを使って、ネットワークテーブルを調べて、必要であれば、クライアントのレコードを修正します。クライアント ID は、指定された IP アドレスと結合されていなければなりません。結合されていない場合は、アドレスプロパティーを編集してこのクライアント ID を追加します。

`Client: clientid is trying to verify unrecorded address: n.n.n.n, ignored.`
原因: 指定されたクライアントは、このアドレスに対して DHCP ネットワークテーブルに登録されていません。そのため、要求は DHCP サーバーに無視されます。

このネットワークの別の DHCP サーバーで、このクライアントにアドレスを割り当てられます。ただし、クライアントがこの IP アドレスをまだ使用しているのにそのクライアントのレコードが削除されていることが原因の場合もあります。

対処方法: DHCP マネージャまたは pntadm コマンドを使って、このサーバーとネットワーク上のほかの DHCP サーバーで、ネットワークテーブルを調べます。必要なら修正します。

さらに、何もせずにリースを期限切れにすることもできます。クライアントは自動的に新しいリースを要求します。

クライアントに新しいリースをすぐに取得させたい場合は、次のコマンドを使って、このクライアント上で DHCP プロトコルを再起動する

```
ifconfig interface dhcp release
ifconfig interface dhcp start
```

DHCP クライアント構成の障害追跡

DHCP クライアントに関連する問題は、一般的に次のカテゴリに分類されます。

- 476 ページの「DHCP サーバーとの通信の問題」
- 485 ページの「不正確な DHCP 構成情報に伴う問題」

DHCP サーバーとの通信の問題

この節では、ネットワークに DHCP クライアントを追加する際に発生する可能性がある問題について説明します。

クライアントソフトウェアを使用可能にし、システムをリブートすると、クライアントはそのネットワーク構成を DHCP サーバーから取得しようとします。クライアントがサーバーと通信できない場合は、次のようなエラーメッセージが表示されます。

```
DHCP or BOOTP server not responding
```

問題を特定するには、クライアントとサーバーの両方から診断情報を収集する必要があります。情報を収集するには、次の方法があります。

1. 477 ページの「DHCP クライアントをデバッグモードで実行する方法」
2. 477 ページの「DHCP サーバーをデバッグモードで実行する方法」
3. 478 ページの「snoop を使用して DHCP ネットワークトラフィックを監視する方法」

これらの方法を個別に、または同時に実行できます。

収集した情報は、問題が発生した場所がクライアントなのか、サーバーなのか、リレーエージェントなのかを判別する上で役立ちます。そして、そのあとで解決策を見つけます。

▼ DHCP クライアントをデバッグモードで実行する方法

DHCP クライアント以外のクライアントをデバッグモードで実行する方法については、そのクライアントのマニュアルを参照してください。

DHCP クライアントの場合は、次のようにします。

1. **DHCP クライアントシステムでスーパーユーザーになります。**
2. **DHCP クライアントデーモンを終了します。**

   ```
   # pkill -x dhcpagent
   ```

3. **デーモンをデバッグモードで再起動します。**

   ```
   # /sbin/dhcpagent -d1 -f &
   ```

   `-d` スイッチによって、DHCP クライアントは、詳細レベル 1 のデバッグモードで動作します。`-f` スイッチによって、出力は、`syslog` の代わりにコンソールに送信されます。

4. **DHCP ネゴシエーションを開始するようにインタフェースを構成します。**

   ```
   # ifconfig interface dhcp start
   ```

   interface には、クライアントネットワークインタフェースの名前 (たとえば、`ge0`) を指定します。

   デバッグモードで実行すると、クライアントデーモンは、DHCP 要求を実行している間、画面にメッセージを表示します。クライアントのデバッグモード出力については、478 ページの「デバッグモードで動作する DHCP クライアントの出力」を参照してください。

▼ DHCP サーバーをデバッグモードで実行する方法

1. **サーバーシステム上でスーパーユーザーになります。**
2. **DHCP サーバーを一時的に停止します。**

   ```
   # svcadm disable -t svc:/network/dhcp-server
   ```

   DHCP マネージャか `dhcpconfig` でサーバーを停止することもできます。

3. **デーモンをデバッグモードで再起動します。**

   ```
   # /usr/lib/inet/in.dhcpd -d -v
   ```

   また、デーモンを実行する際に通常使用する `in.dhcpd` コマンド行オプションも使用する必要があります。たとえば、デーモンを BOOTP リレーエージェントとして実行する場合は、`in.dhcpd -d -v` コマンドに `-r` オプションを付けます。

デバッグモードで動作しているデーモンは、DHCP 要求や BOOTP 要求を処理している間、画面にメッセージを表示します。サーバーのデバッグモードの出力については、479 ページの「デバッグモードで動作する DHCP サーバーの出力」を参照してください。

▼ snoop を使用して DHCP ネットワークトラフィックを監視する方法

1. **DHCP** サーバーシステム上でスーパーユーザーになります。
2. **snoop** を起動して、サーバーのネットワークインタフェース間のネットワークトラフィックの追跡を開始します。

   # /usr/sbin/snoop -d *interface* **-o** *snoop-output-filename* **udp port 67 or udp port 68**

   たとえば、次のコマンドを入力します。

   ```
   #  /usr/sbin/snoop -d hme0 -o /tmp/snoop.output udp port 67 or udp port 68
   ```

   必要な情報を入手したあとも、Control-C を押して snoop を停止するまで、snoop はインタフェースを監視し続けます。
3. クライアントシステムを起動するか、クライアントシステムで **dhcpagent** を再起動します。

   477 ページの「DHCP クライアントをデバッグモードで実行する方法」 を再起動する方法については、 How to Run the DHCP Client in Debugging Mode を参照してください。
4. サーバーシステムで **snoop** を使用して、ネットワークパケットの内容を含む出力ファイルを表示させます。

   # /usr/sbin/snoop -i *snoop-output-filename* **-x0 -v**

   たとえば、次のコマンドを入力します。

   ```
   # /usr/sbin/snoop -i /tmp/snoop.output -x0 -v
   ```

参照　出力の解釈については、482 ページの「DHCP snoop 出力」を参照してください。

デバッグモードで動作する DHCP クライアントの出力

次の例では、デバッグモードで動作する DHCP クライアントが DHCP 要求を送信し、DHCP サーバーから構成情報を受信した場合の、通常のデバッグ出力を示しています。

例 17–1　デバッグモードで動作する DHCP クライアントの通常の出力

```
/sbin/dhcpagent: debug: set_packet_filter: set filter 0x27fc8 (DHCP filter)
/sbin/dhcpagent: debug: init_ifs: initted interface hme0
/sbin/dhcpagent: debug: insert_ifs: hme0: sdumax 1500, optmax 1260, hwtype 1, hwlen 6
/sbin/dhcpagent: debug: insert_ifs: inserted interface hme0
/sbin/dhcpagent: debug: register_acknak: registered acknak id 5
/sbin/dhcpagent: debug: unregister_acknak: unregistered acknak id 5
/sbin/dhcpagent: debug: set_packet_filter: set filter 0x26018 (ARP reply filter)
/sbin/dhcpagent: info: setting IP netmask on hme0 to 255.255.192.0
/sbin/dhcpagent: info: setting IP address on hme0 to 10.23.3.233
/sbin/dhcpagent: info: setting broadcast address on hme0 to 10.23.63.255
/sbin/dhcpagent: info: added default router 10.23.0.1 on hme0
/sbin/dhcpagent: debug: set_packet_filter: set filter 0x28054 (blackhole filter)
/sbin/dhcpagent: debug: configure_if: bound ifsp->if_sock_ip_fd
/sbin/dhcpagent: info: hme0 acquired lease, expires Tue Aug 10 16:18:33 2006
/sbin/dhcpagent: info: hme0 begins renewal at Tue Aug 10 15:49:44 2006
/sbin/dhcpagent: info: hme0 begins rebinding at Tue Aug 10 16:11:03 2006
```

クライアントから DHCP サーバーにアクセスできないと、通常、次のようなデバッグモード出力が表示されます。

例 17–2　デバッグモードで動作する DHCP クライアントの出力 (問題があることを示す)

```
/sbin/dhcpagent: debug: set_packet_filter: set filter 0x27fc8 (DHCP filter)
/sbin/dhcpagent: debug: init_ifs: initted interface hme0
/sbin/dhcpagent: debug: select_best: no valid OFFER/BOOTP reply
/sbin/dhcpagent: debug: select_best: no valid OFFER/BOOTP reply
/sbin/dhcpagent: debug: select_best: no valid OFFER/BOOTP reply
```

このメッセージは、クライアント要求がサーバーに届いていないか、サーバーが回答をクライアントに送信できないことを意味します。サーバーで snoop を実行して、クライアントのパケットがサーバーに届いたかどうかを判別します (478 ページの「snoop を使用して DHCP ネットワークトラフィックを監視する方法」 を参照)。

デバッグモードで動作する DHCP サーバーの出力

サーバーからの通常のデバッグモード出力には、サーバー構成情報が表示され、続いて、デーモンの起動とともに各ネットワークインタフェースの情報が表示されます。デーモンが起動されると、デバッグモード出力には、デーモンが処理している要求の情報が表示されます。例 17–3 は、起動したばかりの DHCP サーバーに関するデバッグモード出力を表しています。サーバーはクライアントのリースを延長します。このクライアントは現在、応答しない別の DHCP サーバーによって所有されているアドレスを使用しています。

例 17–3　デバッグモードで動作する DHCP サーバーの通常の出力

```
Daemon Version: 3.1
Maximum relay hops: 4
Transaction logging to console enabled.
```

例 17–3 デバッグモードで動作する DHCP サーバーの通常の出力 (続き)

```
Run mode is: DHCP Server Mode.
Datastore: nisplus
Path: org_dir.dhcp.test..:dhcp.test..:$
DHCP offer TTL: 10
Ethers compatibility enabled.
BOOTP compatibility enabled.
ICMP validation timeout: 1000 milliseconds, Attempts: 2.
Monitor (0005/hme0) started...
Thread Id: 0005 - Monitoring Interface: hme0 *****
MTU: 1500      Type: DLPI
Broadcast: 10.21.255.255
Netmask: 255.255.0.0
Address: 10.21.0.2
Monitor (0006/nf0) started...
Thread Id: 0006 - Monitoring Interface: nf0 *****
MTU: 4352      Type: DLPI
Broadcast: 10.22.255.255
Netmask: 255.255.0.0
Address: 10.22.0.1
Monitor (0007/qfe0) started...
Thread Id: 0007 - Monitoring Interface: qfe0 *****
MTU: 1500      Type: DLPI
Broadcast: 10.23.63.255
Netmask: 255.255.192.0
Address: 10.23.0.1
Read 33 entries from DHCP macro database on Tue Aug 10 15:10:27 2006
Datagram received on network device: qfe0
Client: 0800201DBA3A is requesting verification of address owned by 10.21.0.4
Datagram received on network device: qfe0
Client: 0800201DBA3A is requesting verification of address owned by 10.21.0.4
Datagram received on network device: qfe0
Client: 0800201DBA3A is requesting verification of address owned by 10.21.0.4
Datagram received on network device: qfe0
Client: 0800201DBA3A maps to IP: 10.23.3.233
Unicasting datagram to 10.23.3.233 address.
Adding ARP entry: 10.23.3.233 == 0800201DBA3A
DHCP EXTEND 0934312543 0934316143 10.23.3.233 10.21.0.2
          0800201DBA3A SUNW.Ultra-5_10 0800201DBA3A
```

例 17–4 は、BOOTP リレーエージェントとして起動された DHCP デーモンのデバッグモード出力を表しています。エージェントは、クライアントの要求を DHCP サーバーに中継し、サーバーの応答をクライアントに中継します。

例 17–4 デバッグモードで動作する BOOTP リレーの通常の出力

```
Relay destination: 10.21.0.4 (blue-servr2)     network: 10.21.0.0
Daemon Version: 3.1
Maximum relay hops: 4
Transaction logging to console enabled.
Run mode is: Relay Agent Mode.
Monitor (0005/hme0) started...
Thread Id: 0005 - Monitoring Interface: hme0 *****
MTU: 1500      Type: DLPI
Broadcast: 10.21.255.255
```

例 17–4 デバッグモードで動作する BOOTP リレーの通常の出力 (続き)

```
Netmask: 255.255.0.0
Address: 10.21.0.2
Monitor (0006/nf0) started...
Thread Id: 0006 - Monitoring Interface: nf0 *****
MTU: 4352      Type: DLPI
Broadcast: 10.22.255.255
Netmask: 255.255.0.0
Address: 10.22.0.1
Monitor (0007/qfe0) started...
Thread Id: 0007 - Monitoring Interface: qfe0 *****
MTU: 1500      Type: DLPI
Broadcast: 10.23.63.255
Netmask: 255.255.192.0
Address: 10.23.0.1
Relaying request 0800201DBA3A to 10.21.0.4, server port.
BOOTP RELAY-SRVR 0934297685 0000000000 0.0.0.0 10.21.0.4 0800201DBA3A
N/A 0800201DBA3A
Packet received from relay agent: 10.23.0.1
Relaying reply to client 0800201DBA3A
Unicasting datagram to 10.23.3.233 address.
Adding ARP entry: 10.23.3.233 == 0800201DBA3A
BOOTP RELAY-CLNT 0934297688 0000000000 10.23.0.1 10.23.3.233 0800201DBA3A
N/A 0800201DBA3A
Relaying request 0800201DBA3A to 10.21.0.4, server port.
BOOTP RELAY-SRVR 0934297689 0000000000 0.0.0.0 10.21.0.4 0800201DBA3A
N/A 0800201DBA3A
Packet received from relay agent: 10.23.0.1
Relaying reply to client 0800201DBA3A
Unicasting datagram to 10.23.3.233 address.
Adding ARP entry: 10.23.3.233 == 0800201DBA3A
```

DHCP に問題があると、デバッグモード出力に警告メッセージやエラーメッセージが表示されることがあります。次の DHCP サーバーのエラーメッセージから解決策を見つけてください。

`ICMP ECHO reply to OFFER candidate:` *ip_address* `disabling`

原因: DHCP サーバーは、IP アドレスをクライアントに提供する前に、ping コマンドを実行してそのアドレスが使用されていないことを確認します。クライアントの回答がある場合には、そのアドレスは使用されています。

対処方法: 構成したアドレスが使用されていないことを確認します。それには、`ping` コマンドを使用します。詳細は、`ping`(1M) のマニュアルページを参照してください。

`No more IP addresses on` *network-address* `network.`

原因: クライアントのネットワークに対応する DHCP ネットワークテーブル中に、使用可能な IP アドレスがありません。

対処方法: DHCP マネージャまたは `pntadm` コマンドを使って、IP アドレスを追加作成します。DHCP デーモンが複数のサブネットを監視している場合は、クライアントが属するサブネットにアドレスを追加します。詳細は、395 ページの「DHCP サービスへの IP アドレスの追加」を参照してください。

`No more IP addresses for` *network-address* `network when you are running the DHCP daemon in BOOTP compatibility mode.`

原因: BOOTP はリース期間を使用しないので、DHCP サーバーは、BOOTP クライアントに割り当てるために設定された BOOTP フラグを持つ空きアドレスを検索します。

対処方法: DHCP マネージャを使用して、BOOTP アドレスを割り当てます。詳細は、388 ページの「DHCP サービスによる BOOTP クライアントのサポート (タスクマップ)」を参照してください。

`Request to access nonexistent per network database:` *database-name* `in datastore:` *datastore.*

原因: DHCP サーバーの構成の際に、サブネットの DHCP ネットワークテーブルが作成されていません。

対処方法: DHCP マネージャまたは `pntadm` コマンドを使って、DHCP ネットワークテーブルと新しい IP アドレスを作成します。詳細は、381 ページの「DHCP ネットワークの追加」を参照してください。

`There is no` *table-name* `dhcp-network table for DHCP client's network.`

原因: DHCP サーバーの構成の際に、サブネットの DHCP ネットワークテーブルが作成されていません。

対処方法: DHCP マネージャまたは `pntadm` コマンドを使って、DHCP ネットワークテーブルと新しい IP アドレスを作成します。詳細は、381 ページの「DHCP ネットワークの追加」を参照してください。

`Client using non_RFC1048 BOOTP cookie.`

原因: ネットワーク上のデバイスが、BOOTP のサポートされていない実装にアクセスしようとしました。

対処方法: このデバイスを構成する必要がない場合は、このメッセージを無視します。このデバイスをサポートする方法については、388 ページの「DHCP サービスによる BOOTP クライアントのサポート (タスクマップ)」を参照してください。

DHCP snoop 出力

下記の `snoop` 出力を見ると、DHCP クライアントシステムと DHCP サーバーシステムの間でパケットが交換されていることがわかります。パケットには、各システムの IP アドレスが示されています。さらに、パケットのパスにルーターやリ

レーエージェントがある場合は、その IP アドレスも含まれています。システム間でパケットの交換が行われない場合は、クライアントシステムからサーバーシステムに全くアクセスできないことがあります。その場合、問題はそれより下のレベルにあります。

snoop 出力を評価するためには、期待する動作がどのようなものかを知っておく必要があります。たとえば、要求が BOOTP リレーエージェントを通るべきかどうかを知っている必要があります。さらに、関係するシステムの MAC アドレスや IP アドレスを知っている必要があります。それによって、それらの値が本来のものかどうかを判別できるからです。また、複数のネットワークインタフェースがある場合は、ネットワークインタフェースのアドレスも知っている必要があります。

次の例は、DHCP 肯定応答メッセージの通常の snoop 出力を示しています。このメッセージは、blue-servr2 上の DHCP サーバーから MAC アドレスが 8:0:20:8e:f3:7e のクライアントに送信されたものです。このメッセージで、サーバーは IP アドレス 192.168.252.6 とホスト名 white-6 をクライアントに割り当てます。さらに、このメッセージには、クライアントに対するいくつもの標準的なネットワークオプションといくつかのベンダー固有のオプションが含まれています。

例 17–5　1 つのパケットに関する snoop 出力の例

```
ETHER:  ----- Ether Header -----
ETHER:
ETHER:  Packet 26 arrived at 14:43:19.14
ETHER:  Packet size = 540 bytes
ETHER:  Destination = 8:0:20:8e:f3:7e, Sun
ETHER:  Source      = 8:0:20:1e:31:c1, Sun
ETHER:  Ethertype = 0800 (IP)
ETHER:
IP:   ----- IP Header -----
IP:
IP:   Version = 4
IP:   Header length = 20 bytes
IP:   Type of service = 0x00
IP:         xxx. .... = 0 (precedence)
IP:         ...0 .... = normal delay
IP:         .... 0... = normal throughput
IP:         .... .0.. = normal reliability
IP:   Total length = 526 bytes
IP:   Identification = 64667
IP:   Flags = 0x4 IP:         .1.. .... = do not fragment
IP:         ..0. .... = last fragment
IP:   Fragment offset = 0 bytes
IP:   Time to live = 254 seconds/hops
IP:   Protocol = 17 (UDP)
IP:   Header checksum = 157a
IP:   Source address = 10.21.0.4, blue-servr2
IP:   Destination address = 192.168.252.6, white-6
IP:   No options
IP:   UDP:  ----- UDP Header -----
UDP:
UDP:  Source port = 67
```

例 17–5　1 つのパケットに関する snoop 出力の例　　(続き)

```
UDP:  Destination port = 68 (BOOTPC)
UDP:  Length = 506
UDP:  Checksum = 5D4C
UDP:
DHCP: ----- Dynamic Host Configuration Protocol -----
DHCP:
DHCP: Hardware address type (htype) =  1 (Ethernet (10Mb))
DHCP: Hardware address length (hlen) = 6 octets
DHCP: Relay agent hops = 0
DHCP: Transaction ID = 0x2e210f17
DHCP: Time since boot = 0 seconds
DHCP: Flags = 0x0000
DHCP: Client address (ciaddr) = 0.0.0.0
DHCP: Your client address (yiaddr) = 192.168.252.6
DHCP: Next server address (siaddr) = 10.21.0.2
DHCP: Relay agent address (giaddr) = 0.0.0.0
DHCP: Client hardware address (chaddr) = 08:00:20:11:E0:1B
DHCP:
DHCP: ----- (Options) field options -----
DHCP:
DHCP: Message type = DHCPACK
DHCP: DHCP Server Identifier = 10.21.0.4
DHCP: Subnet Mask = 255.255.255.0
DHCP: Router at = 192.168.252.1
DHCP: Broadcast Address = 192.168.252.255
DHCP: NISPLUS Domainname = dhcp.test
DHCP: IP Address Lease Time = 3600 seconds
DHCP: UTC Time Offset = -14400 seconds
DHCP: RFC868 Time Servers at = 10.21.0.4
DHCP: DNS Domain Name = sem.example.com
DHCP: DNS Servers at = 10.21.0.1
DHCP: Client Hostname = white-6
DHCP: Vendor-specific Options (166 total octets):
DHCP:   (02) 04 octets  0x8194AE1B (unprintable)
DHCP:   (03) 08 octets  "pacific"
DHCP:   (10) 04 octets  0x8194AE1B (unprintable)
DHCP:   (11) 08 octets  "pacific"
DHCP:   (15) 05 octets  "xterm"
DHCP:   (04) 53 octets  "/export/s2/base.s2s/latest/Solaris_8/Tools/Boot"
DHCP:   (12) 32 octets  "/export/s2/base.s2s/latest"
DHCP:   (07) 27 octets  "/platform/sun4u/kernel/unix"
DHCP:   (08) 07 octets  "EST5EDT"
  0: 0800 208e f37e 0800 201e 31c1 0800 4500    .. .ó~.. .1...E.
 16: 020e fc9b 4000 fe11 157a ac15 0004 c0a8    ....@....z......
 32: fc06 0043 0044 01fa 5d4c 0201 0600 2e21    ...C.D..]L.....!
 48: 0f17 0000 0000 0000 0000 c0a8 fc06 ac15    ................
 64: 0002 0000 0000 0800 2011 e01b 0000 0000    ........ .......
 80: 0000 0000 0000 0000 0000 0000 0000 0000    ................
 96: 0000 0000 0000 0000 0000 0000 0000 0000    ................
112: 0000 0000 0000 0000 0000 0000 0000 0000    ................
128: 0000 0000 0000 0000 0000 0000 0000 0000    ................
144: 0000 0000 0000 0000 0000 0000 0000 0000    ................
160: 0000 0000 0000 0000 0000 0000 0000 0000    ................
176: 0000 0000 0000 0000 0000 0000 0000 0000    ................
192: 0000 0000 0000 0000 0000 0000 0000 0000    ................
208: 0000 0000 0000 0000 0000 0000 0000 0000    ................
```

例 17–5　1 つのパケットに関する snoop 出力の例　　(続き)

```
224: 0000 0000 0000 0000 0000 0000 0000 0000    ................
240: 0000 0000 0000 0000 0000 0000 0000 0000    ................
256: 0000 0000 0000 0000 0000 0000 0000 0000    ................
272: 0000 0000 0000 6382 5363 3501 0536 04ac    ......c.Sc5..6..
288: 1500 0401 04ff ffff 0003 04c0 a8fc 011c    ................
304: 04c0 a8fc ff40 0964 6863 702e 7465 7374    .....@.dhcp.test
320: 3304 0000 0e10 0204 ffff c7c0 0404 ac15    3...............
336: 0004 0f10 736e 742e 6561 7374 2e73 756e    ....sem.example.
352: 2e63 6f6d 0604 ac15 0001 0c07 7768 6974    com.........whit
368: 652d 362b a602 0481 94ae 1b03 0861 746c    e-6+.........pac
384: 616e 7469 630a 0481 94ae 1b0b 0861 746c    ific.........pac
400: 616e 7469 630f 0578 7465 726d 0435 2f65    ific...xterm.5/e
416: 7870 6f72 742f 7332 382f 6261 7365 2e73    xport/sx2/bcvf.s
432: 3238 735f 776f 732f 6c61 7465 7374 2f53    2xs_btf/latest/S
448: 6f6c 6172 6973 5f38 2f54 6f6f 6c73 2f42    olaris_x/Tools/B
464: 6f6f 740c 202f 6578 706f 7274 2f73 3238    oot. /export/s2x
480: 2f62 6173 652e 7332 3873 5f77 6f73 2f6c    /bcvf.s2xs_btf/l
496: 6174 6573 7407 1b2f 706c 6174 666f 726d    atest../platform
512: 2f73 756e 346d 2f6b 6572 6e65 6c2f 756e    /sun4u/kernel/un
528: 6978 0807 4553 5435 4544 54ff              ix..EST5EDT.
```

不正確な DHCP 構成情報に伴う問題

DHCP クライアントが受信するネットワーク構成情報に不正確な情報がある場合は、DHCP サーバーのデータを参照する必要があります。DHCP サーバーが処理するこのクライアント用のマクロのオプション値を確認してください。不正確な情報の例には、間違った NIS ドメイン名やルーター IP アドレスがあります。

正しくない情報の原因がどこにあるのかを特定する際には、次の一般的な指針に従ってください。

- サーバーに定義されているマクロを調べます (410 ページの「DHCP サーバー上で定義されたマクロを表示する方法 (DHCP マネージャ)」を参照)。323 ページの「マクロ処理の順序」の情報を確認し、このクライアントのためにどのマクロが自動的に処理されるかを判別します。
- ネットワークテーブルを調べて、クライアントの IP アドレスに構成マクロとして割り当てられたマクロ (ある場合) を確認します。詳細は、391 ページの「DHCP サービスで IP アドレスを使用して作業する (作業マップ)」を参照してください。
- 複数のマクロで使用されているオプションがないか調べます。オプションに指定したい値が、最後に処理されるマクロに設定されるか確認します。
- 適切なマクロを編集して、正確な値がクライアントに確実に渡されるようにします。詳細は、410 ページの「DHCP マクロの変更」を参照してください。

DHCP クライアント指定のホスト名に関連する問題

ここでは、独自のホスト名を DNS に登録する必要がある DHCP クライアントの問題について説明します。

DHCP クライアントがホスト名を要求しない

クライアントが DHCP クライアントでない場合は、そのクライアントのマニュアルを参照して、ホスト名を要求するために必要なクライアントの構成方法を確認してください。DHCP クライアントの場合は、458 ページの「DHCPv4 クライアントが特定のホスト名を要求できるようにする方法」を参照してください。

要求されたホスト名を DHCP クライアントが受け取らない

次の各項では、クライアントがそのホスト名を取得する際に起こる問題とその解決策について説明します。

問題: クライアントは DHCP サーバーからオファーを受け取るが、サーバーが DNS 更新を行わない。

対処方法: クライアントから 2 つの DHCP サーバーにアクセスできる場合は、両方のサーバーが DNS 更新を行うように構成されている必要があります。DHCP サーバーと DNS サーバーの構成については、371 ページの「DHCP サーバーによる動的 DNS 更新の有効化」を参照してください。

DNS 更新を提供するように DHCP サーバーが構成されているかどうかを判別するには、次のようにします。

1. クライアントの DHCP サーバーの IP アドレスを判別します。クライアントシステムで `snoop` か別のアプリケーションを使ってネットワークパケットを捕捉します。478 ページの「snoop を使用して DHCP ネットワークトラフィックを監視する方法」を参照し、その手順をクライアント (サーバーではなく) で実行します。`snoop` 出力で DHCP Server Identifier を探して、サーバーの IP アドレスを取得します。
2. DHCP サーバーシステムにログインして、システムが、DNS 更新を行うように構成されているか確認します。次のコマンドをスーパーユーザーとして入力します。

   ```
   dhcpconfig -P
   ```

   `UPDATE_TIMEOUT` がサーバーパラメータとして表示される場合、DHCP サーバーは DNS 更新を行うように構成されています。

3. DNS サーバーで /etc/named.conf ファイルを調べます。適切なドメインの zone セクションで allow-update キーワードを探します。サーバーが DHCP サーバーによる DNS 更新を許している場合は、DHCP サーバーの IP アドレスが allow-update キーワードにあります。

問題: クライアントが FQDN オプションを使ってホスト名を指定している。FQDN オプションは DHCP プロトコルに正式には含まれていないため、現在、DHCP ではサポートされていません。

対処方法: そのサーバーで snoop か別のアプリケーションを使ってネットワークパケットを捕捉します。詳細は、478 ページの「snoop を使用して DHCP ネットワークトラフィックを監視する方法」を参照してください。snoop 出力で、クライアントからのパケットにある FQDN オプションを探します。

Hostname オプションを使ってホスト名を指定するようにクライアントの構成を変更します。Hostname はオプションコード 12 です。 詳細は、クライアントのマニュアルを参照してください。

Oracle Solaris クライアントの場合は、458 ページの「DHCPv4 クライアントが特定のホスト名を要求できるようにする方法」 を参照してください。

問題: クライアントにアドレスオファーを行う DHCP サーバーがクライアントの DNS ドメインを知らない。

対処方法: DHCP サーバーで、DNSdmain オプションに有効な値が設定されているか確認します。このクライアントに対して処理されるマクロの DNSdmain オプションに正しい DNS ドメイン名を設定します。DNSdmain は通常、ネットワークマクロに含まれているマクロ内のオプションの値を変更する方法については、410 ページの「DHCP マクロの変更」を参照してください。

問題: クライアントが要求したホスト名が DHCP サーバーが管理していない IP アドレスに対応している。DHCP サーバーは、自らが管理していない IP アドレスに対し DNS 更新を行いません。

対処方法: 次のいずれかの DHCP サーバーメッセージが syslog に書き込まれていないか調べます。

- `There is no` *n.n.n.n* `dhcp-network table for DHCP client's network.`
- `DHCP network record for` *n.n.n.n* `is unavailable, ignoring request.`

別の名前を要求するようにクライアントを構成します。458 ページの「DHCPv4 クライアントが特定のホスト名を要求できるようにする方法」を参照してください。新しく指定する名前には、DHCP サーバーが管理しているアドレスに対応する名前を選択します。DHCP マネージャの「アドレス (Addresses)」タブのアドレスマッピングを見てください。あるいは、どの IP アドレスにも対応していないアドレスを選択することもできます。

問題: クライアントから要求されたホスト名が、使用不可の状態にある IP アドレスに対応している。このアドレスは、ほかの目的で使用されている可能性があります (別のクライアントにリースされているか、別のクライアントにオファー中である)。

対処方法: 次の DHCP サーバーメッセージが `syslog` に書き込まれていないか調べます。 `ICMP ECHO reply to OFFER candidate:` *n.n.n.n.*

異なる IP アドレスに対応する名前を選択するようにクライアントを構成します。あるいは、そのアドレスを使用するクライアントからアドレスを取り返します。

問題: DHCP サーバーからの更新を受け付けるように DNS サーバーが構成されていない。

対処方法: DNS サーバーの `/etc/named.conf` ファイルを調べます。DHCP サーバーのドメインに対する適切な `zone` セクションで `allow-update` キーワードが付いた DHCP サーバーの IP アドレスを探します。IP アドレスが存在しないなら、DNS サーバーは、DHCP サーバーからの更新を受け付けるようには構成されていません。

詳細は、373 ページの「DHCP クライアント用に動的 DNS 更新を有効にする方法」を参照してください。

DHCP サーバーに複数のインタフェースがある場合は、DHCP サーバーのすべてのアドレスからの更新を受け付けるように DNS サーバーを構成する必要がある場合があります。DNS サーバーのデバッグ機能を有効にして、更新が DNS サーバーに届いているか確認します。DNS サーバーが更新要求を受け取っている場合は、デバッグモード出力を見て、更新が行われなかった原因を調べます。DNS デバッグモードについては、`in.named(1M)` のマニュアルページを参照してください。

問題: DNS 更新が、割り当てられた時間内に行われていない可能性がある。構成された時間内に DNS 更新が完了しないと、DHCP サーバーはホスト名をクライアントに返しません。ただし、DNS 更新を完了する試みは続けられます。

対処方法: `nslookup` コマンドを使って、更新が正常に終わっているかを確認します。詳細は、`nslookup(1M)` のマニュアルページを参照してください。

たとえば、DNS ドメインが `hills.example.org` で、DNS サーバーの IP アドレスが `10.76.178.11` であるとします。さらに、クライアントが登録したいホスト名は `cathedral` だとします。`cathedral` が DNS サーバーに登録されているかどうかを知るには、次のコマンドが使用できます。

```
nslookup cathedral.hills.example.org 10.76.178.11
```

更新は正常に行われたが、割り当てられた時間を超えている場合は、タイムアウト値を増やす必要があります。詳細は、373 ページの「DHCP クライアント用に動的 DNS 更新を有効にする方法」を参照してください。この手順では、タイムアウトになる前に DNS サーバーから応答を受け取れるように、タイムアウトの秒数を増やすべきです。

◆ ◆ ◆ 第 18 章

18 DHCP コマンドと DHCP ファイル (リファレンス)

この章では、DHCP コマンドと DHCP ファイルの関連について説明します。ただし、この章にはコマンドの使い方は含まれていません。

この章では、次の内容について説明します。

DHCP のコマンド

次の表に、ネットワーク上で DHCP を管理するために使用できるコマンドを示します。

表 18–1 DHCP で使用されるコマンド

コマンド	説明
/usr/lib/inet/dhcpd	ISC DHCP のみ: ISC DHCP サーバーデーモン。詳細については、dhcpd(8) のマニュアルページを参照してください。
/usr/lib/inet/dhcrelay	ISC DHCP のみ: DHCP および BOOTP 要求を、DHCP サーバーがないネットワーク上のクライアントから別のネットワーク上のサーバーに中継する手段を有効にします。詳細については、dhcrelay(8) のマニュアルページを参照してください。
/usr/lib/inet/in.dhcpd	DHCP サーバーデーモン。デーモンはシステムの起動時に起動されます。したがって、サーバーデーモンを直接起動すべきではありません。デーモンの起動や停止には、DHCP マネージャー、svcadm コマンド、または dhcpconfig を使用できます。問題をトラブルシュートするためにデーモンをデバッグモードで実行する場合にのみデーモンを直接起動します。詳細については、in.dhcpd(1M) のマニュアルページを参照してください。

表 18–1 DHCP で使用されるコマンド (続き)

コマンド	説明
/usr/sadm/admin/bin/dhcpmgr	DHCP マネージャー。DHCP サービスの構成や管理に使用するグラフィカルユーザーインタフェース (GUI) ツールです。DHCP マネージャは、推奨 DHCP 管理ツールです。詳細については、dhcpmgr(1M) のマニュアルページを参照してください。
/usr/sbin/dhcpagent	DHCP クライアントデーモン。DHCP プロトコルのクライアント側を実装します。詳細については、dhcpagent(1M) のマニュアルページを参照してください。
/usr/sbin/dhcpconfig	DHCP サーバーや BOOTP リレーエージェントの構成や構成解除を行うために使用されます。さらに、データストアを別のデータストアへ変換したり、DHCP 構成データのインポートやエクスポートを行うときにも使用します。詳細については、dhcpconfig(1M) のマニュアルページを参照してください。
/usr/sbin/dhcpinfo	Oracle Solaris クライアントシステムのシステム起動スクリプトの中で、DHCP クライアントデーモン (dhcpagent) からホスト名などの情報を取得するときに使用します。また、スクリプトやコマンド行で dhcpinfo を使用して、特定のパラメータ値を取得することもできます。詳細については、dhcpinfo(1) のマニュアルページを参照してください。
/usr/sbin/dhtadm	dhcptab テーブル内のオプションやマクロを変更するために使用されます。このコマンドは、DHCP 情報を自動的に変更するために作成するスクリプトでもっとも役立ちます。dhtadm に -P オプションを指定し、その結果を grep コマンドに渡すと、dhcptab テーブル内の特定のオプション値をすばやく検索できます。詳細については、dhtadm(1M) のマニュアルページを参照してください。
/usr/sbin/ifconfig	IP アドレスをネットワークインタフェースに割り当てたり、ネットワークインタフェースパラメータを構成したりする場合 (あるいはその両方の場合)、システムブート時に使用します。 DHCP クライアントでは、ifconfig によって DHCP が起動し、ネットワークインタフェースの構成に必要なパラメータ (IP アドレスを含む) が取得されます。詳細は、ifconfig(1M) のマニュアルページを参照してください。
/usr/sbin/omshell	ISC DHCP のみ: Object Management API (OMAPI) を使用して ISC DHCP サーバーの状態を照会および変更する手段を提供します。詳細については、omshell(1) のマニュアルページを参照してください。
/usr/sbin/pntadm	DHCP ネットワークテーブルを変更するときに使用され、このテーブルでは、クライアント ID と IP アドレスが対応付けられ、オプションとして構成情報と IP アドレスが関連付けられます。詳細は、pntadm(1M) のマニュアルページを参照してください。
/usr/sbin/snoop	ネットワーク経由で渡されるパケットの内容を取得および表示するときに使用されます。snoop は、DHCP サービスの問題をトラブルシューティングするときに役立ちます。詳細については、snoop(1M) のマニュアルページを参照してください。

スクリプトにおける DHCP コマンドの実行

dhcpconfig、dhtadm、pntadm コマンドは、スクリプト中での使用に適しています。特に、pntadm コマンドは大量の IP アドレスエントリを DHCP ネットワークテーブルに作成するときに便利です。次のサンプルスクリプトでは、バッチモードで pntadm を使って、IP アドレスを作成しています。

例 18–1　addclient.ksh スクリプトで pntadm コマンドを使用する

```
#! /usr/bin/ksh
#
# This script utilizes the pntadm batch facility to add client entries
# to a DHCP network table. It assumes that the user has the rights to
# run pntadm to add entries to DHCP network tables.

#
# Based on the switch setting, query the netmasks table for a netmask.
# Accepts one argument, a dotted IP address.
#
get_netmask()
{
    MTMP=‘getent netmasks ${1} | awk ’{ print $2 }’‘
    if [ ! -z "${MTMP}" ]
    then
        print - ${MTMP}
    fi
}

#
# Based on the network specification, determine whether or not network is
# subnetted or supernetted.
# Given a dotted IP network number, convert it to the default class
# network.(used to detect subnetting). Requires one argument, the
# network number. (e.g. 10.0.0.0) Echos the default network and default
# mask for success, null if error.
#
get_default_class()
{
    NN01=${1%%.*}
    tmp=${1#*.}
    NN02=${tmp%%.*}
    tmp=${tmp#*.}
    NN03=${tmp%%.*}
    tmp=${tmp#*.}
    NN04=${tmp%%.*}
    RETNET=""
    RETMASK=""

    typeset -i16 ONE=10#${1%%.*}
    typeset -i10 X=$((${ONE}&16#f0))
    if [ ${X} -eq 224 ]
    then
        # Multicast
        typeset -i10 TMP=$((${ONE}&16#f0))
        RETNET="${TMP}.0.0.0"
        RETMASK="240.0.0.0"
    fi
```

例 18–1 addclient.ksh スクリプトで pntadm コマンドを使用する (続き)

```
    typeset -i10 X=$((${ONE}&16#80))
    if [ -z "${RETNET}" -a ${X} -eq 0 ]
    then
        # Class A
        RETNET="${NN01}.0.0.0"
        RETMASK="255.0.0.0"
    fi
    typeset -i10 X=$((${ONE}&16#c0))
    if [ -z "${RETNET}" -a ${X} -eq 128 ]
    then
        # Class B
        RETNET="${NN01}.${NN02}.0.0"
        RETMASK="255.255.0.0"
    fi
    typeset -i10 X=$((${ONE}&16#e0))
    if [ -z "${RETNET}" -a ${X} -eq 192 ]
    then
        # Class C
        RETNET="${NN01}.${NN02}.${NN03}.0"
        RETMASK="255.255.255.0"
    fi
    print - ${RETNET} ${RETMASK}
    unset NNO1 NNO2 NNO3 NNO4 RETNET RETMASK X ONE
}

#
# Given a dotted form of an IP address, convert it to its hex equivalent.
#
convert_dotted_to_hex()
{
    typeset -i10 one=${1%%.*}
    typeset -i16 one=${one}
    typeset -Z2 one=${one}
    tmp=${1#*.}

    typeset -i10 two=${tmp%%.*}
    typeset -i16 two=${two}
    typeset -Z2 two=${two}
    tmp=${tmp#*.}

    typeset -i10 three=${tmp%%.*}
    typeset -i16 three=${three}
    typeset -Z2 three=${three}
    tmp=${tmp#*.}

    typeset -i10 four=${tmp%%.*}
    typeset -i16 four=${four}
    typeset -Z2 four=${four}

     hex=‘print - ${one}${two}${three}${four} | sed -e ’s/#/0/g’‘
     print - 16#${hex}
     unset one two three four tmp
}

#
# Generate an IP address given the network address, mask, increment.
```

例 18–1 addclient.ksh スクリプトで pntadm コマンドを使用する (続き)

```
#
get_addr()
{
    typeset -i16 net=‘convert_dotted_to_hex ${1}‘
    typeset -i16 mask=‘convert_dotted_to_hex ${2}‘
    typeset -i16 incr=10#${3}

    # Maximum legal value - invert the mask, add to net.
    typeset -i16 mhosts=~${mask}
    typeset -i16 maxnet=${net}+${mhosts}

    # Add the incr value.
    let net=${net}+${incr}

    if [ $((${net} < ${maxnet})) -eq 1 ]
    then
        typeset -i16 a=${net}\&16#ff000000
        typeset -i10 a="${a}>>24"

        typeset -i16 b=${net}\&16#ff0000
        typeset -i10 b="${b}>>16"

        typeset -i16 c=${net}\&16#ff00
        typeset -i10 c="${c}>>8"

        typeset -i10 d=${net}\&16#ff
        print - "${a}.${b}.${c}.${d}"
    fi
    unset net mask incr mhosts maxnet a b c d
}

# Given a network address and client address, return the index.
client_index()
{
    typeset -i NNO1=${1%%.*}
    tmp=${1#*.}
    typeset -i NNO2=${tmp%%.*}
    tmp=${tmp#*.}
    typeset -i NNO3=${tmp%%.*}
    tmp=${tmp#*.}
    typeset -i NNO4=${tmp%%.*}

    typeset -i16 NNF1
    let NNF1=${NNO1}
    typeset -i16 NNF2
    let NNF2=${NNO2}
    typeset -i16 NNF3
    let NNF3=${NNO3}
    typeset -i16 NNF4
    let NNF4=${NNO4}
    typeset +i16 NNF1
    typeset +i16 NNF2
    typeset +i16 NNF3
    typeset +i16 NNF4
    NNF1=${NNF1#16\#}
    NNF2=${NNF2#16\#}
```

例 18–1 addclient.ksh スクリプトで pntadm コマンドを使用する (続き)

```
NNF3=${NNF3#16\#}
NNF4=${NNF4#16\#}
if [ ${#NNF1} -eq 1 ]
then
    NNF1="0${NNF1}"
fi
if [ ${#NNF2} -eq 1 ]
then
    NNF2="0${NNF2}"
fi
if [ ${#NNF3} -eq 1 ]
then
    NNF3="0${NNF3}"
fi
if [ ${#NNF4} -eq 1 ]
then
    NNF4="0${NNF4}"
fi
typeset -i16 NN
let NN=16#${NNF1}${NNF2}${NNF3}${NNF4}
unset NNF1 NNF2 NNF3 NNF4

typeset -i NNO1=${2%%.*}
tmp=${2#*.}
typeset -i NNO2=${tmp%%.*}
tmp=${tmp#*.}
typeset -i NNO3=${tmp%%.*}
tmp=${tmp#*.}
typeset -i NNO4=${tmp%%.*}
typeset -i16 NNF1
let NNF1=${NNO1}
typeset -i16 NNF2
let NNF2=${NNO2}
typeset -i16 NNF3
let NNF3=${NNO3}
typeset -i16 NNF4
let NNF4=${NNO4}
typeset +i16 NNF1
typeset +i16 NNF2
typeset +i16 NNF3
typeset +i16 NNF4
NNF1=${NNF1#16\#}
NNF2=${NNF2#16\#}
NNF3=${NNF3#16\#}
NNF4=${NNF4#16\#}
if [ ${#NNF1} -eq 1 ]
then
    NNF1="0${NNF1}"
fi
if [ ${#NNF2} -eq 1 ]
then
    NNF2="0${NNF2}"
fi
if [ ${#NNF3} -eq 1 ]
then
    NNF3="0${NNF3}"
```

例 18–1 addclient.kshスクリプトでpntadmコマンドを使用する (続き)

```
    fi
    if [ ${#NNF4} -eq 1 ]
    then
        NNF4="0${NNF4}"
    fi
    typeset -i16 NC
    let NC=16#${NNF1}${NNF2}${NNF3}${NNF4}
    typeset -i10 ANS
    let ANS=${NC}-${NN}
    print - $ANS
}

#
# Check usage.
#
if [ "$#" != 3 ]
then
    print "This script is used to add client entries to a DHCP network"
    print "table by utilizing the pntadm batch facilty.\n"
    print "usage: $0 network start_ip entries\n"
    print "where: network is the IP address of the network"
        print "       start_ip is the starting IP address \n"
        print "       entries is the number of the entries to add\n"
    print "example: $0 10.148.174.0 10.148.174.1 254\n"
    return
fi

#
# Use input arguments to set script variables.
#
NETWORK=$1
START_IP=$2
typeset -i STRTNUM=‘client_index ${NETWORK} ${START_IP}‘
let ENDNUM=${STRTNUM}+$3
let ENTRYNUM=${STRTNUM}
BATCHFILE=/tmp/batchfile.$$
MACRO=‘uname -n‘

#
# Check if mask in netmasks table. First try
# for network address as given, in case VLSM
# is in use.
#
NETMASK=‘get_netmask ${NETWORK}‘
if [ -z "${NETMASK}" ]
then
    get_default_class ${NETWORK} | read DEFNET DEFMASK
    # use the default.
    if [ "${DEFNET}" != "${NETWORK}" ]
    then
        # likely subnetted/supernetted.
        print - "\n\n###\tWarning\t###\n"
        print - "Network ${NETWORK} is netmasked, but no entry was found  \n
              in the ’netmasks’ table; please update the ’netmasks’  \n
              table in the appropriate nameservice before continuing. \n
              (See /etc/nsswitch.conf.) \n" >&2
```

例 18–1 addclient.ksh スクリプトで pntadm コマンドを使用する (続き)

```
        return 1
    else
        # use the default.
        NETMASK="${DEFMASK}"
    fi
fi

#
# Create a batch file.
#
print -n "Creating batch file "
while [ ${ENTRYNUM} -lt ${ENDNUM} ]
do
    if [ $((${ENTRYNUM}-${STRTNUM}))%50 -eq 0 ]
    then
        print -n "."
    fi

    CLIENTIP=‘get_addr ${NETWORK} ${NETMASK} ${ENTRYNUM}‘
    print "pntadm -A ${CLIENTIP} -m ${MACRO} ${NETWORK}" >> ${BATCHFILE}
    let ENTRYNUM=${ENTRYNUM}+1
done
print " done.\n"

#
# Run pntadm in batch mode and redirect output to a temporary file.
# Progress can be monitored by using the output file.
#
print "Batch processing output redirected to ${BATCHFILE}"
print "Batch processing started."

pntadm -B ${BATCHFILE} -v > /tmp/batch.out 2 >&1

print "Batch processing completed."
```

DHCP サービスによって使用されるファイル

次の表に、DHCP に関連するファイルを示します。

表 18–2 DHCP デーモンや DHCP コマンドで使用されるファイル

ファイルまたはテーブル名	説明
dhcptab	レガシーの Sun DHCP のみ: DHCP 構成情報のテーブルを表す総称的な用語。構成情報は割り当てられた値と一緒にオプションとして記録され、さらにマクロとしてグループ化されます。dhcptab テーブルの名前と場所は、DHCP 情報用に使用するデータストアによって決まります。詳細は、dhcptab(4) のマニュアルページを参照してください。

表 18–2 DHCP デーモンや DHCP コマンドで使用されるファイル (続き)

ファイルまたはテーブル名	説明
DHCP ネットワークテーブル	レガシーの Sun DHCP のみ: IP アドレスをクライアント ID および構成オプションにマップします。DHCP ネットワークテーブルの名前は、`10.21.32.0` など、ネットワークの IP アドレスに基づいて付けられます。`dhcp_network` というファイルはありません。DHCP ネットワークテーブルの名前と場所は、DHCP 情報用に使用するデータストアによって決まります。詳細は、`dhcp_network(4)` のマニュアルページを参照してください。
`/etc/dhcp/eventhook`	レガシーの Sun DHCP のみ: `dhcpagent` デーモンが自動的に実行できるスクリプトまたは実行可能ファイル。詳細については、`dhcpagent(1M)` のマニュアルページを参照してください。
`/etc/inet/dhcpd4.conf` `/etc/inet/dhcpd6.conf`	ISC DHCP のみ: ISC DHCP サーバー `dhcpd` 用の構成情報を含みます。詳細については、`dhcpd.conf(5)` のマニュアルページを参照してください。
`/etc/inet/dhcpsvc.conf`	レガシーの Sun DHCP のみ: DHCP デーモンの起動オプションと、データストア情報を格納しています。このファイルを手動で編集してはいけません。起動オプションの変更には `dhcpconfig` コマンドを使用します。詳細は、`dhcpsvc.conf(4)` のマニュアルページを参照してください。
`nsswitch.conf`	ネームサービスデータベースの場所と、それらのデータベースをどのような順序で検索してさまざまな情報を入手するかを指定します。`nsswitch.conf` ファイルは、DHCP サーバーを構成する際に正確な構成情報を入手するために使用されます。このファイルは、`/etc` ディレクトリに存在します。詳細は、`nsswitch.conf(4)` のマニュアルページを参照してください。
`resolv.conf`	DNS クエリーを解決するための情報が含まれています。DHCP サーバーの構成中に、このファイルで、DNS ドメインと DNS サーバーに関する情報が調べられます。このファイルは、`/etc` ディレクトリに存在します。詳細は、`resolv.conf(4)` のマニュアルページを参照してください。
`dhcp.`*interface*	ファイル名 `dhcp.`*interface* で指定されたクライアントネットワークインタフェースで DHCP が使用されることを示します。たとえば、`dhcp.qe0` という名前のファイルが存在する場合、DHCP は `qe0` インタフェースで使用されることを表します。`dhcp.`*interface* ファイルには、`ifconfig` コマンド (クライアントでの DHCP 起動に使用) にオプションとして渡されるコマンドが含まれていることがあります。このファイルは、DHCP クライアントシステムの `/etc` ディレクトリにあります。特定のマニュアルページはありません。`dhcp(5)` を参照してください。

表 18–2 DHCP デーモンや DHCP コマンドで使用されるファイル (続き)

ファイルまたはテーブル名	説明
/etc/dhcp/*interface*.dhc /etc/dhcp/*interface*.dh6	DHCP から取得した特定のネットワークインタフェースの構成パラメータが含まれています。DHCPv4 の場合、ファイル名は dhc で終わります。DHCPv6 の場合、ファイル名は dh6 で終わります。インタフェースの IP アドレスのリースが停止されると、このクライアントは、/etc/dhcp/*interface*.dhc にある現在の構成情報をキャッシュします。たとえば、DHCP が qe0 インタフェースで使用されている場合、dhcpagent は、構成情報を /etc/dhcp/qe0.dhc にキャッシュします。DHCP が次にこのインタフェースで起動するときに、リースの有効期限内であれば、このクライアントはキャッシュされた情報を使用するように要求します。DHCP サーバーがこの要求を拒否すると、クライアントは標準の DHCP リースネゴシエーション手順を開始します。
/etc/default/dhcpagent	dhcpagent クライアントデーモンのパラメータ値を設定します。パラメータについては、/etc/default/dhcpagent ファイルか、dhcpagent(1M) のマニュアルページを参照してください。
/etc/dhcp/inittab /etc/dhcp/inittab6	レガシーの Sun DHCP のみ: データ型などの DHCP オプションコードのさまざまな要素を定義し、ニーモニックラベルを割り当てます。ファイルの構文については、dhcp_inittab(4) のマニュアルページを参照してください。/etc/dhcp/inittab6 は DHCPv6 クライアントによって使用されます。 クライアント側では、/etc/dhcp/inittab ファイル内の情報は、情報を判読するユーザーに意味のある情報を提供するために dhcpinfo コマンドによって使用されます。DHCP サーバーシステムでは、DHCP デーモンと管理ツールがこのファイルから DHCP オプション情報を入手します。 以前のリリースで使用されていた /etc/dhcp/dhcptags ファイルは /etc/dhcp/inittab ファイルで置き換えられている。
/var/db/isc-dhcp/dhcp4.leases /var/db/isc-dhcp/dhcp4.leases- /var/db/isc-dhcp/dhcp6.leases /var/db/isc-dhcp/dhcp6.leases-	ISC DHCP のみ: DHCPv4 および DHCPv6 サーバー用のリースを一覧表示します。ファイル名の末尾に「-」が付いたファイルは以前のコピーです。

DHCP のオプション情報

従来、DHCP のオプション情報は、サーバーの dhcptab テーブルやクライアントの dhcptags ファイル、さらにさまざまなプログラムの内部テーブルなど、複数の場所に格納されてきました。そのため、Solaris 8 リリース以降では、すべてのオプション情報が /etc/dhcp/inittab ファイルに統合されています。ファイルの詳細については、dhcp_inittab(4) のマニュアルページを参照してください。

DHCP クライアントでは、dhcptags ファイルの代わりに DHCP の inittab ファイルを使用します。クライアントは、DHCP パケットの一部としてオプションコードを受

け取ると、その情報をこのファイルから取得します。DHCP サーバーの in.dhcpd、snoop、dhcpmgr プログラムでもこの inittab ファイルを使用します。

サイトが影響を受けるかどうかの判別

DHCP を使用するほとんどのサイトは、/etc/dhcp/inittab ファイルに切り替えても影響を受けません。影響を受けるのは、次の条件がすべて当てはまるサイトだけです。

- Solaris 8 よりも前の Oracle Solaris リリースをアップグレードする予定がある。
- 以前に新しい DHCP オプションを作成したことがある。
- /etc/dhcp/dhcptags ファイルを変更したことがあり、その変更を保持したい。

アップグレードを行うと、dhcptags ファイルが変更されたために DHCP inittab ファイルを変更する必要があることを示すメッセージがアップグレードログに書き込まれます。

dhcptags ファイルと inittab ファイルの違い

inittab ファイルには、dhcptags ファイルよりも多くの情報が含まれています。さらに、inittab ファイルでは、異なる構文が使用されます。

次に、dhcptags のエントリ例を示します。

```
33 StaticRt - IPList Static_Routes
```

33 は DHCP パケットで渡される数値コードです。StaticRt はオプション名です。IPList は、StaticRt のデータタイプが一連の IP アドレスでなければならないことを示します。Static_Routes は記述名です。

inittab ファイルは、これらのオプションを 1 行で表した複数のレコードから構成されています。形式は、dhcptab のシンボルを定義する形式と似ています。次の表に、inittab ファイルの構文について説明します。

オプション	説明
option-name	オプションの名前。オプション名は、そのオプションのカテゴリ内部で一意である必要があります。また、Standard、Site、Vendor のカテゴリにある、ほかのオプション名と重複できません。たとえば、同じ名前を持つ Site オプションを 2 つ持つことはできず、Standard のオプションと同じ名前の Site のオプションは作成できません。
category	オプションが所属する名前空間を特定します。次のいずれかである必要があります。 Standard、Site、Vendor、Field、または Internal。

code オプションがネットワーク経由で送信されたときにそのオプションを特定します。多くの場合、カテゴリがなくてもコードはオプションを一意に特定します。ただし、Field や Internal などの内部カテゴリの場合は、コードがほかの目的に使用される場合があります。コードは、広域的に一意でない場合があります。コードは、オプションのカテゴリ内部では一意であることが必要で、Standard と Site のフィールドにあるコードと重複することはできません。

type このオプションと関連するデータを記述します。有効なタイプは、IP、ASCII、Octet、Boolean、Unumber8、Unumber16、Unumber32、Unumber64、Snumber8、Snumber16、Snumber32、および Snumber64 です。数値の場合、先頭の U や S は、数値が符号なしか符号ありかを示します。終わりの数字は、数値にいくつのビットが含まれているかを表します。たとえば、Unumber8 は無符号の 8 ビット数値を表します。タイプには、大文字小文字の区別はありません。

granularity このオプションの値を構成するデータ単位数を記述します。

maximum このオプションに指定可能な値の個数を記述します。0 は、無限大の数を表します。

consumers この情報を使用できるプログラムを記述します。これには次の `sdmi` を指定します。

```
s    snoop

d    in.dhcpd

m    dhcpmgr

i    dhcpinfo
```

次に `inittab` エントリの例を示します。

```
StaticRt - Standard, 33, IP, 2, 0, sdmi
```

このエントリは、オプションが `StaticRt` という名前で、Standard カテゴリに属し、オプションコード 33 であることを示します。これで表されるデータは、理論上無数の IP アドレスの組です。なぜなら、タイプが `IP`、データ単位が 2、最大が無限 (`0`) だからです。このオプションを利用するのは `sdmi`: `snoop`、`in.dhcpd`、`dhcpmgr`、`dhcpinfo` です。

dhcptags エントリの inittab エントリへの変換

以前に `dhcptags` ファイルにエントリを追加している場合は、対応するエントリを新しい `inittab` ファイルにも追加する必要があります。ただし、以前に追加したオプ

ションを使用しない場合は不要です。次の例では、dhcptags エントリの例を inittab フォーマットで表す方法を示しています。

ネットワークに接続されたファックスについて、次の dhcptags エントリを追加したと想定してください。

```
128 FaxMchn - IP Fax_Machine
```

コード 128 は、オプションが Site カテゴリに属していなければならないことを示します。オプション名は FaxMchn、データタイプは IP です。

対応する inittab エントリは次のとおりです。

```
FaxMchn SITE, 128, IP, 1, 1, sdmi
```

データ単位数が 1、指定可能な値の数が 1 なので、このオプションには 1 つの IP アドレスを指定することを表しています。

パート IV

IP セキュリティー

このパートでは、ネットワークセキュリティーについて説明します。IP セキュリティーアーキテクチャー (IPsec) は、ネットワークをパケットレベルで保護します。インターネット鍵管理 (IKE) は、IPsec の鍵を管理します。Oracle Solaris の IP フィルタ機能はファイアウォールを提供します。

◆ ◆ ◆ 第 19 章

IP セキュリティーアーキテクチャー (概要)

IP セキュリティーアーキテクチャー (IPsec) は、IPv4 および IPv6 ネットワークパケットで IP データグラムを暗号化して保護します。

この章では、次の内容について説明します。

IPsec をネットワークに実装するには、第 20 章「IPsec の構成 (タスク)」を参照してください。参考情報については、第 21 章「IP セキュリティーアーキテクチャー (リファレンス)」を参照してください。

IPsec の新機能

Solaris 10 4/09: このリリース以降、サービス管理機能 (SMF) は IPsec を一連のサービスとして管理します。

デフォルトでは、システムのブート時に次の 2 つの IPsec サービスが有効になります。

- `svc:/network/ipsec/policy:default`
- `svc:/network/ipsec/ipsecalgs:default`

デフォルトでは、システムのブート時に鍵管理サービスは無効になっています。

- `svc:/network/ipsec/manual-key:default`
- `svc:/network/ipsec/ike:default`

SMF の下で IPsec ポリシーを有効にするには、次の手順を実行します。

1. `ipsecinit.conf` ファイルに IPsec ポリシーエントリを追加します。
2. Internet Key Exchange (IKE) を構成するか、鍵を手動で構成します。
3. IPsec ポリシーサービスをリフレッシュします。
4. 鍵管理サービスを有効にします。

SMF については、『Oracle Solaris の管理: 基本管理』の第 18 章「サービスの管理 (概要)」を参照してください。`smf(5)` および `svcadm(1M)` のマニュアルページも参照してください。

このリリース以降では、`ipsecconf` コマンドと `ipseckey` コマンドに `-c` オプションが追加されており、それぞれの構成ファイルの構文をチェックできます。また、IPsec と IKE を管理するための Network IPsec Management 権利プロファイルが用意されています。

Solaris 10 7/07: このリリース以降、IPsec はトンネルモードのトンネルを完全に実装し、トンネルをサポートするユーティリティーは変更されます。

- IPsec は、仮想プライベートネットワーク (VPN) 用のトンネルモードのトンネルを実装します。トンネルモードでの IPsec は、単一の NAT の背後にある複数のクライアントをサポートします。トンネルモードでの IPsec は、ほかのベンダーによって実装された IP 内 IP トンネルと相互運用できます。IPsec は、引き続きトランスポートモードのトンネルをサポートするので、以前の Solaris リリースとの互換性があります。
- トンネルを作成するための構文が簡素化されます。IPsec ポリシーを管理するために、`ipsecconf` コマンドが拡張されました。IPsec ポリシーの管理に `ifconfig` コマンドは推奨されなくなりました。
- このリリース以降、`/etc/ipnodes` ファイルは削除されます。ネットワークの IPv6 アドレスを構成するには、`/etc/hosts` ファイルを使用してください。

Solaris 10 1/06: このリリース以降では、IKE は完全に NAT-Traversal サポート (RFC 3947 と RFC 3948 を参照) に準拠します。IKE 操作は暗号化フレームワークから PKCS #11 ライブラリを使用し、パフォーマンスを向上させます。

この暗号化フレームワークは、メタスロットを使用するアプリケーションにソフトトークンキーストアを提供します。IKE がメタスロットを使用するとき、キーを格納する場所として、ディスク、接続したボード、またはソフトトークンキーストアを選択できます。

- ソフトトークンキーストアを使用する方法については、`cryptoadm(1M)` のマニュアルページを参照してください。
- Oracle Solaris の新機能の完全な一覧については、『Oracle Solaris 10 1/13 の新機能』を参照してください。

IPsec とは

IPsec は、パケットの認証または暗号化、もしくはこの両方を実行することで、IP パケットを保護します。IPsec は IP モジュールの内側で実行されます。したがって、インターネットアプリケーションは、IPsec を使用するために自分自身を構成することなく、IPsec を利用できます。正しく使用すれば、IPsec は、ネットワークトラフィックの保護に有効なツールとなります。

IPsec の保護に関連する主要コンポーネントは次のとおりです。

- セキュリティープロトコル – IP データグラムの保護メカニズムです。認証ヘッダー (AH) は IP パケットのハッシュを含み、完全性を保証します。データグラムの内容は暗号化されませんが、パケットの内容が変更されていないことが受信側に保証されます。また、パケットが送信側によって送られたことも保証されます。カプセル化セキュリティーペイロード (ESP)は、IP データを暗号化し、パケット転送中に内容が分からないようにします。ESP は、認証アルゴリズムオプションによってデータの完全性も保証します。
- セキュリティーアソシエーション **(SA)** – ネットワークトラフィックの特定のフローに適用される暗号化パラメータと IP セキュリティープロトコル。各 SA は、セキュリティーパラメータインデックス (SPI) と呼ばれる一意の参照を持ちます。
- セキュリティーアソシエーションデータベース **(SADB)** – IP 宛先アドレスと索引番号にセキュリティープロトコルを関連付けるデータベースです。この索引番号は、セキュリティーパラメータインデックス (SPI)と呼ばれます。これらの 3 つの要素 (セキュリティープロトコル、宛先アドレスおよび SPI) は、正当な IPsec パケットを一意に識別します。セキュリティーアソシエーションデータベースにより、パケットの宛先に届いた保護対象のパケットは確実に受信側に認識されます。また、受信側は、このデータベースの情報を使用して、通信を復号化し、パケットが変更されていないことを確認し、パケットを再度組み立て、そのパケットを最終的な宛先に届けます。
- 鍵管理 – 暗号化アルゴリズムおよび SPI 用の鍵の生成と配布です。
- セキュリティーメカニズム – IP データグラム内のデータを保護する認証アルゴリズムと暗号化アルゴリズムです。

- セキュリティーポリシーデータベース **(SPD)** – パケットに適用される保護レベルを指定するデータベースです。SPD は、IP トラフィックをフィルタリングし、パケットの処理方法を決定します。パケットは破棄したり、問題ない場合は、通過させたりできます。また、IPsec で保護することも可能です。アウトバウンドパケットの場合、SPD と SADB が適用する保護レベルを決定します。インバウンドパケットの場合、パケットの保護レベルを許容できるかどうかの決定に SPD が役立ちます。パケットが IPsec によって保護されている場合は、パケットを復号化し、確認したあとに、SPD が参照されます。

IPsec は、セキュリティーメカニズムを IP 宛先アドレスに転送される IP データグラムに適用します。受信側ユーザーは、SADB の情報を使用して、到着パケットが正当なことを確認し、それらを復号化します。アプリケーションで IPsec を呼び出すと、ソケット単位レベルでも IP データグラムにセキュリティーメカニズムが適用されます。

ポートのソケットが接続され、そのあとで IPsec ポリシーがそのポートに適用された場合、そのソケットを使用するトラフィックは IPsec によって保護されません。当然、IPsec ポリシーがポートに適用されたあとにポート上で開かれたソケットは、IPsec ポリシーによって保護されます。

IPsec RFC

インターネットエンジニアリングタスクフォース (IETF) は、IP 層のセキュリティーアーキテクチャーを説明するいくつかの RFC (Requests for Comments) を公表しています。すべての RFC の著作権は、インターネット協会が有しています。RFC へのリンクについては、`http://www.ietf.org/`を参照してください。次の RFC リストは、比較的一般的な IP セキュリティーの参考文献です。

- RFC 2411、『IP Security Document Roadmap』、1998 年 11 月
- RFC 2401、『Security Architecture for the Internet Protocol』、1998 年 11 月
- RFC 2402、『IP Authentication Header』、1998 年 11 月
- RFC 2406、『IP Encapsulating Security Payload (ESP)』、1998 年 11 月
- RFC 2408、『Internet Security Association and Key Management Protocol (ISAKMP)』、1998 年 11 月
- RFC 2407、『The Internet IP Security Domain of Interpretation for ISAKMP』、1998 年 11 月
- RFC 2409、『The Internet Key Exchange (IKE)』、1998 年 11 月
- RFC 3554、『On the Use of Stream Control Transmission Protocol (SCTP) with IPsec』、2003 年 7 月、[Oracle Solaris 10 リリースでは実装せず]

IPsec の用語

IPsec RFC は、IPsec をシステムに実装する際に分かっていると便利な用語を多数定義しています。次の例は、IPsec の用語、それらの一般的に使用されている用語を示し、各用語を定義しています。鍵のネゴシエーションで使用する用語の一覧は、表 22–1 を参照してください。

表 19–1　IPsec の用語、略語、および使用方法

IPsec の用語	略語	定義
セキュリティーアソシエーション	SA	ネットワークトラフィックの特定のフローに適用される暗号化パラメータと IP セキュリティープロトコル。SA は、セキュリティープロトコル、一意のセキュリティーパラメータインデックス (SPI)、および IP 宛先の 3 つで定義されます。
セキュリティーアソシエーションデータベース	SADB	アクティブなセキュリティーアソシエーションをすべて含むデータベース。
セキュリティーパラメータインデックス	SPI	セキュリティーアソシエーションの索引値。SPI は、同じ IP 宛先およびセキュリティープロトコルを持つ SA を区別する 32 ビットの値です。
セキュリティーポリシーデータベース	SPD	アウトバウンドパケットとインバウンドパケットの保護レベルが指定どおりかを判断するデータベース。
鍵交換		非対称暗号化アルゴリズムを使用して鍵を生成する処理。主な手法には、RSA と Diffie-Hellman の 2 つがあります。
Diffie-Hellman	DH	鍵生成と鍵認証を可能にする鍵交換アルゴリズム。しばしば「認証された鍵交換」と呼ばれます。
RSA	RSA	鍵生成と鍵配布を可能にする鍵交換アルゴリズム。このプロトコル名は、作成者の Rivest、Shamir、Adleman の三氏に因んでいます。
インターネットセキュリティーアソシエーションおよび鍵管理プロトコル	ISAKMP	SA 属性の形式を設定し、SA のネゴシエーション、変更、削除を行うための共通フレームワーク。ISAKMP は、IKE 交換を処理するための IETF 標準です。

IPsec パケットのフロー

図 19–1 は、IPsec がアウトバウンドパケットで呼び出されたときに、IP アドレスを持つパケットが IP データグラムの一部としてどのように処理されるかを示しています。フロー図は、認証ヘッダー (AH) とカプセル化されたセキュリティーペイロード (ESP) エンティティーがどこでパケットに適用されるかを示しています。これらのエンティティーの適用方法とアルゴリズムの選択方法については、これ以降のセクションで説明します。

図 19–2 は、IPsec インバウンドプロセスを示しています。

図 19–1 アウトバウンドパケットプロセスに適用された IPsec

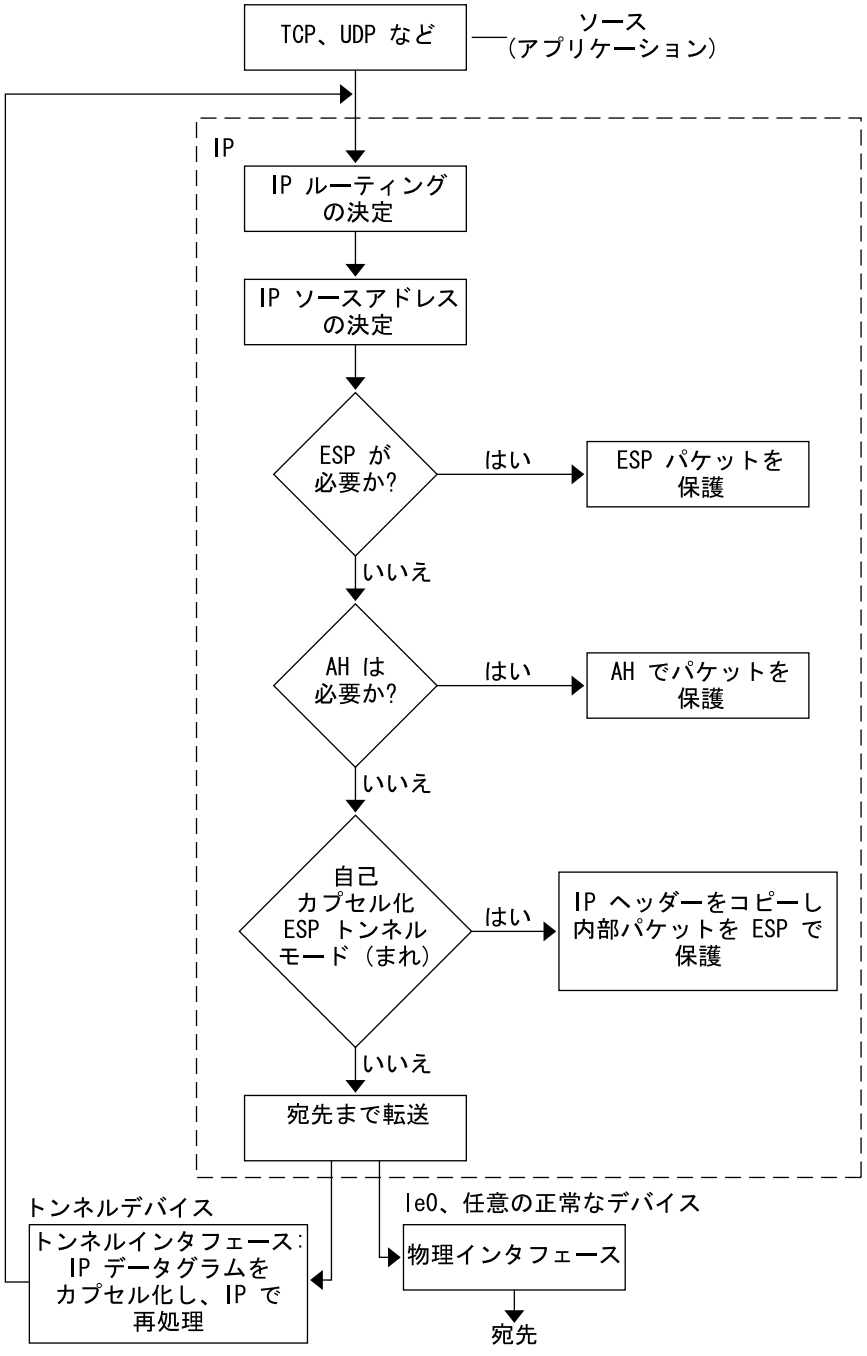
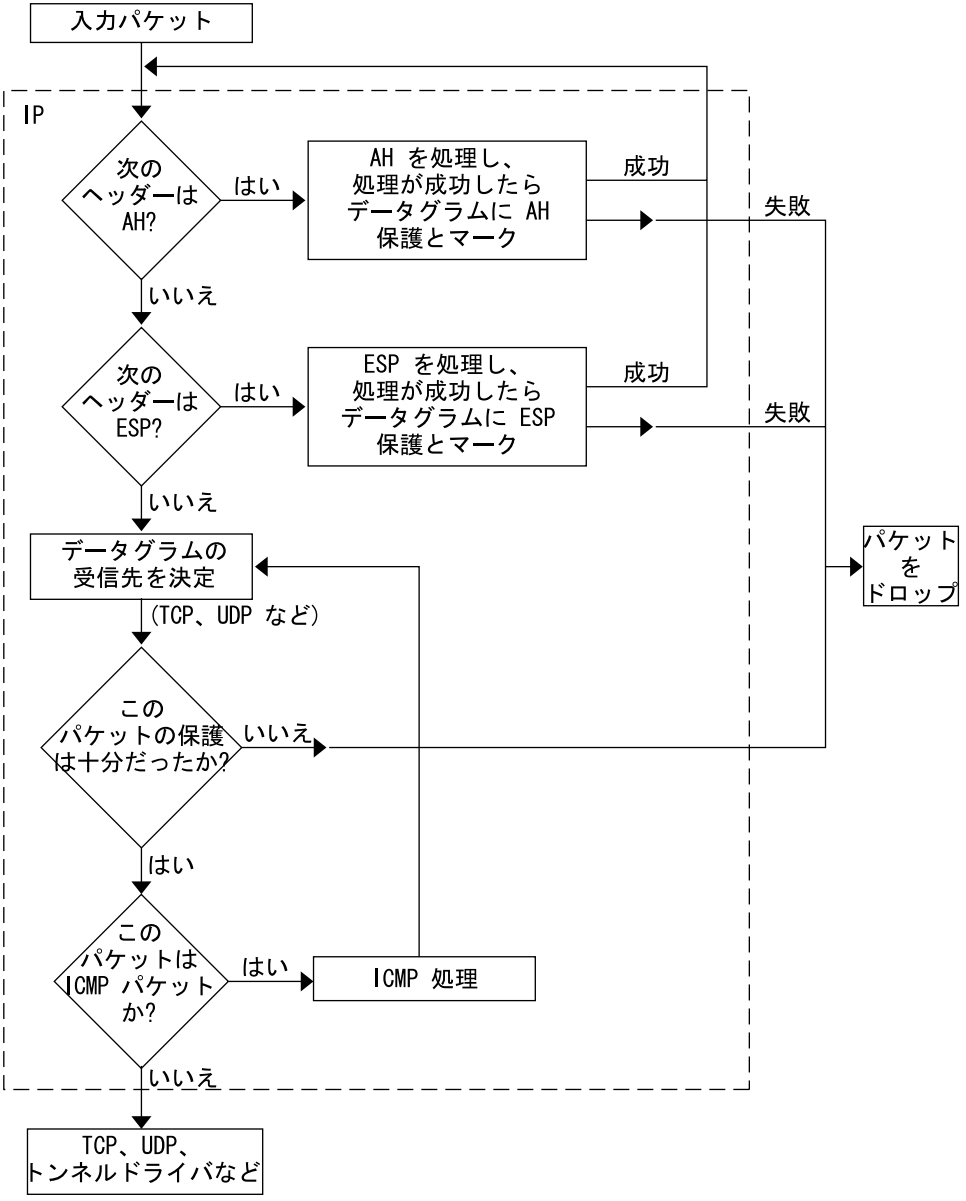


図 19–2　IPsec を入力パケットプロセスに適用



IPsec セキュリティーアソシエーション

IPsec の セキュリティーアソシエーション (SA) は、通信するホストが認識するセキュリティープロパティーを示します。1 つの SA は、1 方向のデータを保護します。つまり、1 つのホストかグループ (マルチキャスト) アドレスのどちらかです。大部分の通信がピアツーピアかクライアントサーバーなので、両方向のトラフィックの安全性を確保するために 2 つの SA が必要です。

次の 3 つの要素は、IPsec SA を一意に識別します。

- セキュリティープロトコル (AH または ESP)
- 宛先 IP アドレス
- セキュリティーパラメータインデックス (SPI)

任意の 32 ビット値の SPI は、AH パケットまたは ESP パケットで転送されます。AH および ESP によって保護される範囲については、 `ipsecah(7P)` と `ipsecesp(7P)` のマニュアルページを参照してください。完全性チェックサム値を使用して、パケットを認証します。認証が失敗すると、パケットがドロップされます。

SA は、セキュリティーアソシエーションデータベース (SADB) に格納されます。ソケットベースの管理インタフェース `PF_KEY` により、特権を持つアプリケーションでそのデータベースを管理できます。たとえば、IKE アプリケーションと `ipseckeys` コマンドは `PF_KEY` ソケットインタフェースを使用します。

- IPsec SADB のより完全な説明については、589 ページの「IPsec のセキュリティーアソシエーションデータベース」を参照してください。
- SADB の管理については、`pf_key(7P)` のマニュアルページを参照してください。

IPsec での鍵管理

セキュリティーアソシエーション (SA) は、認証および暗号化で使用するキー作成素材を必要とします。このキーイング素材の管理を鍵管理と呼びます。IKE (インターネット鍵交換) プロトコルにより、鍵管理が自動的に行われます。また、`ipseckey` コマンドを指定して、鍵管理を手動で行うこともできます。

IPv4 と IPv6 パケットの SA は、どちらの鍵管理方法も使用できます。手動で鍵管理を行う決定的な理由がないかぎり、IKE をお勧めします。

Oracle Solaris のサービス管理機能 (SMF) 機能は、次の IPsec 用鍵管理サービスを提供します。

- `svc:/network/ipsec/ike:default` サービス – 自動鍵管理のための SMF サービスです。`ike` サービスは `in.iked` デーモンを実行して自動鍵管理を提供します。IKE については、第 22 章「インターネット鍵交換 (概要)」を参照してください。`in.iked` デーモンの詳細については、`in.iked(1M)` のマニュアルページを参照してください。`ike` サービスについては、649 ページの「IKE サービス」を参照してください。
- `svc:/network/ipsec/manual-key:default` サービス – 手動での鍵管理のための SMF サービスです。`manual-key` サービスは `ipseckey` コマンドを各種オプションで実行して、鍵を手動で管理します。`ipseckey` コマンドについては、589 ページの「IPsec の SA を生成するためのユーティリティー」を参照してください。`ipseckey` コマンドオプションの詳細な説明については、`ipseckey(1M)` のマニュアルページを参照してください。

Solaris 10 4/09 リリースより前のリリースでは、`in.iked` コマンドと `ipseckey` コマンドで鍵情報を管理します。

- `in.iked` デーモンは、自動的な鍵管理を提供します。IKE については、第 22 章「インターネット鍵交換 (概要)」を参照してください。`in.iked` デーモンの詳細については、`in.iked(1M)` のマニュアルページを参照してください。
- `ipseckey` コマンドを使用すると、手動でキーを管理できます。このコマンドの説明については、589 ページの「IPsec の SA を生成するためのユーティリティー」を参照してください。`ipseckey` コマンドオプションの詳細な説明については、`ipseckey(1M)` のマニュアルページを参照してください。

IPsec の保護メカニズム

IPsec は、データを保護するために次の 2 つのセキュリティープロトコルを提供しています。

- 認証ヘッダー (AH)
- カプセル化セキュリティーペイロード (ESP)

AH は、認証アルゴリズムでデータを保護します。ESP は、暗号化アルゴリズムでデータを保護します。ESP は認証メカニズムと組み合わせて使用可能であり、またそうすべきです。NAT をトラバースしない場合は、ESP と AH を組み合わせることができます。それ以外の場合、ESP で認証アルゴリズムと暗号化メカニズムを使用できます。

認証ヘッダー

認証ヘッダーは、IP データグラムに対するデータ認証、強力な完全性、再送保護を供給します。AH では大部分の IP データグラムを保護します。次の図に示されているように、AH は IP ヘッダーとトランスポートヘッダーの間に挿入されます。

トランスポートヘッダーは、TCP、UDP、SCTP、または ICMP のいずれかです。トンネルを使用している場合は、トランスポートヘッダーがこれ以外の IP ヘッダーである場合もあります。

カプセル化セキュリティーペイロード

カプセル化セキュリティーペイロード (ESP)モジュールは、ESP がカプセル化した対象の機密性を守ります。また、AH が提供するサービスも提供します。ただし、保護される対象は、データグラムのうち ESP がカプセル化した部分だけです。ESP は、保護されたパケットの完全性を保証するオプションの認証サービスを提供します。ESP は暗号化対応技術を使用するため、ESP を提供するシステムは輸出入管理法の対象となります。

ESP はデータをカプセル化します。したがって、次の図に示されているように、ESP が保護するのはデータグラム内の EPS の開始点以降のデータのみです。

暗号化部分

TCP パケットでは、ESP は TCP ヘッダーとそのデータだけをカプセル化します。パケットが IP 内 IP データグラムの場合、ESP は内部 IP データグラムを保護します。ソケット別ポリシーでは、「自己カプセル化」ができるため、必要に応じて ESP では IP オプションをカプセル化できます。

自己カプセル化が設定されている場合は、IP 内 IP データグラムを構築するために IP ヘッダーのコピーが作成されます。たとえば、TCP ソケットに自己カプセル化が設定されていない場合、データグラムは次の形式で送信されます。

```
[ IP(a -> b) options + TCP + data ]
```

TCP ソケットに自己カプセル化が設定されている場合、データグラムは次の形式で送信されます。

```
[ IP(a -> b) + ESP [ IP(a -> b) options + TCP + data ] ]
```

さらに詳しくは、518 ページの「IPsec のトランスポートモードとトンネルモード」を参照してください。

AH と ESP を使用する場合のセキュリティー上の考慮事項

次の表では、AH と ESP が提供する保護を比較しています。

表 19–2 IPsec で AH と ESP が提供する保護

プロトコル	パケットの範囲	保護	対象となる攻撃
AH	IP ヘッダーからトランスポートヘッダーまでのパケットを保護	強力な完全性およびデータ認証を提供します。 ■ 送信側が送ったものとまったく同じものを受信側が受け取ることを保証する ■ AH がリプレー保護を有効にしていない場合は、リプレー攻撃を受けやすい	リプレー、カットアンドペースト
ESP	データグラムの ESP 開始後のパケットを保護	暗号化オプションで、IP ペイロードを暗号化します。機密性を確保します	盗聴
		認証オプションで、AH と同じペイロード保護を提供します	リプレー、カットアンドペースト
		両方のオプションで、強力な完全性、データ認証、および機密性を提供します	リプレー、カットアンドペースト、盗聴

IPsec の認証アルゴリズムと暗号化アルゴリズム

IPsec セキュリティープロトコルは、認証と暗号化という 2 種類のアルゴリズムを提供しています。AH モジュールは、認証アルゴリズムを使用します。ESP モジュールは、暗号化アルゴリズムと認証アルゴリズムを使用します。`ipsecalgs` コマンドを使用すると、システムのアルゴリズムとプロパティーの一覧を取得できます。詳細は、`ipsecalgs(1M)` のマニュアルページを参照してください。`getipsecalgbyname(3NSL)` のマニュアルページで説明されている機能を使用して、アルゴリズムのプロパティーを検索することもできます。

IPsec は、暗号化フレームワークを使用してアルゴリズムにアクセスします。暗号化フレームワークは、その他のサービスに加えて、アルゴリズムの集中リポジトリを提供します。このフレームワークによって、IPsec は、高性能な暗号ハードウェアアクセラレータを利用できます。

詳細については、次を参照してください。

- 『Solaris のシステム管理: セキュリティーサービス』の第 13 章「Oracle Solaris の暗号化フレームワーク (概要)」
- 『Oracle Solaris 10 セキュリティー開発者ガイド』の第 8 章「Oracle Solaris 暗号化フレームワークの紹介」

IPsec での認証アルゴリズム

認証アルゴリズムは、データとキーを基に整合性チェックサムの値、つまり、「ダイジェスト」を生成します。AH モジュールは、認証アルゴリズムを使用します。ESP モジュールも、認証アルゴリズムを使用します。

IPsec での暗号化アルゴリズム

暗号化アルゴリズムは、キーでデータを暗号化します。IPsec の ESP モジュールは、暗号化アルゴリズムを使用します。暗号化アルゴリズムでは、「ブロックサイズ」ごとにデータを処理します。

デフォルトで使用される暗号化アルゴリズムは、Oracle Solaris のリリースによって異なります。

Solaris 10 7/07 リリース以降では、Solaris Encryption Kit の内容は Solaris インストールメディアによってインストールされます。このリリースでは、SHA2 認証アルゴリズム sha256、sha384、および sha512 が追加されています。SHA2 の実装は RFC 4868 仕様に適合しています。このリリースでは、Diffie-Hellman グループ 2048 ビット (グループ 14)、3072 ビット (グループ 15)、および 4096 ビット (グループ 16) も追加されています。ただし、CoolThreads テクノロジを備えた Oracle Sun システムでは、2048 ビットグループだけが高速化されます。

注意 – Solaris 10 7/07 リリース以降では、システムに Solaris Encryption Kit を追加しないでください。このキットはシステムにおける暗号化のパッチレベルを低下させます。このキットはシステム上の暗号化と互換性がありません。

IPsec の保護ポリシー

IPsec の保護ポリシーは、どのセキュリティーメカニズムも使用できます。IPsec ポリシーは、次のレベルで適用できます。

- システム規模レベル
- ソケット単位レベル

IPsec は、システム共通ポリシーをアウトバウンドデータグラムとインバウンドデータグラムに適用します。アウトバウンドデータグラムは、保護付きまたは保護なしで送信されます。保護が適用されると、特定アルゴリズムか汎用アルゴリズム

のどちらかになります。システムで認識されるデータがあるため、アウトバウンドデータグラムにはその他の規則も適用できます。インバウンドデータグラムの処理は、受理されるか拒絶されるかのどちらかです。インバウンドデータグラムの受理か拒絶を決定する基準はいくつかありますが、場合によってはその基準が重複したり競合することがあります。競合の解決に当たっては、どの規則の構文解析を最初に行うかが決定されます。ポリシーのエントリによって、そのトラフィックがすべてのほかのポリシーを省略すると指示されている場合を除いて、トラフィックは自動的に受理されます。

データグラムを保護する通常のポリシーを省略することもできます。それには、システム規模ポリシーに例外を指定するか、ソケット単位ポリシーで省略を要求します。システム内トラフィックの場合、ポリシーは実施されますが、実際のセキュリティーメカニズムは適用されません。その代わりに、イントラシステム内パケットのアウトバウンドポリシーが、セキュリティー機能の適用されたインバウンドパケットになります。

`ipsecinit.conf` ファイルと `ipsecconf` コマンドを使用して、IPsec ポリシーを構成します。詳細と例については、`ipsecconf(1M)` のマニュアルページを参照してください。

IPsec のトランスポートモードとトンネルモード

IPsec 規格では、IPsec の動作モードとして「トランスポートモード」と「トンネルモード」という 2 つの異なるモードが定義されています。これらのモードは、パケットの符号化には影響を与えません。各モードで、パケットは AH または ESP、あるいはその両方によって保護されます。内側のパケットが IP パケットである場合に、モードによってポリシーの適用方法が次のように異なります。

- トランスポートモードでは、外側のヘッダーによって、内側の IP パケットを保護する IPsec ポリシーが決まります。
- トンネルモードでは、内側の IP パケットによって、その内容を保護する IPsec ポリシーが決まります。

トランスポートモードでは、外側のヘッダー、次のヘッダー、および次のヘッダーでサポートされるすべてのポートを使用して、IPsec ポリシーを決定できます。実際、IPsec は 2 つの IP アドレスの間で異なるトランスポートモードポリシーを適用でき、ポート単位まで細かく設定できます。たとえば、次のヘッダーが TCP であれば、ポートをサポートするので、外側の IP アドレスの TCP ポートに対して IPsec ポリシーを設定できます。同様に、次のヘッダーが IP ヘッダーであれば、外側のヘッダーと内側の IP ヘッダーを使用して IPsec ポリシーを決定できます。

トンネルモードは IP 内 IP データグラムに対してのみ機能します。トンネルモードのトンネリングは、自宅のコンピュータから中央コンピュータに接続する場合に役立ちます。トンネルモードでは、IPsec ポリシーは内側の IP データグラムの内容に適用

されます。内側の IP アドレスごとに異なる IPsec ポリシーを適用できます。つまり、内側の IP ヘッダー、その次のヘッダー、および次のヘッダーでサポートされるポートを使用して、ポリシーを適用することができます。トランスポートモードとは異なり、トンネルモードでは、外側の IP ヘッダーによって内側の IP データグラムのポリシーが決まることはありません。

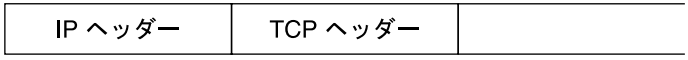
したがって、トンネルモードでは、ルーターの背後にある LAN のサブネットや、そのようなサブネットのポートに対して、IPsec ポリシーを指定することができます。これらのサブネット上の特定の IP アドレス (つまり、ホスト) に対しても、IPsec ポリシーを指定することができます。これらのホストのポートに対しても、固有の IPsec ポリシーを適用できます。ただし、トンネルを経由して動的ルーティングプロトコルが実行されている場合は、サブネットやアドレスは選択しないでください。ピアネットワークでのネットワークトポロジのビューが変化する可能性があるためです。そのような変化があると、静的な IPsec ポリシーが無効になります。静的ルートの構成を含むトンネリング手順の例については、550 ページの「IPsec による VPN の保護 (タスクマップ)」を参照してください。

Oracle Solaris では、IP トンネルネットワークインタフェースにのみトンネルモードを適用できます。`ipsecconf` コマンドには、IP トンネルネットワークインタフェースを選択するための `tunnel` キーワードが用意されています。規則内に `tunnel` キーワードが含まれている場合は、その規則に指定されているすべてのセレクタが内側のパケットに適用されます。

トランスポートモードでは、ESP または AH、あるいはその両方を使用してデータグラムを保護できます。

次の図は、IP ヘッダーと保護されていない TCP パケットを示します。

図 19–3　TCP 情報を伝送する保護されていない IP パケット



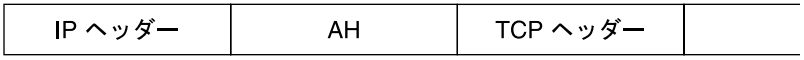
トランスポートモードで、ESP は次の図のようにデータを保護します。網かけされた領域は、パケットの暗号化された部分を示します。

図 19–4　TCP 情報を伝送する保護された IP パケット

暗号化部分

トランスポートモードで、AH は次の図のようにデータを保護します。

図 19–5 認証ヘッダーで保護されたパケット



AH 保護では、トランスポートモードの場合でも、IP ヘッダーの大部分が保護されます。

トンネルモードでは、データグラム全体が IPsec ヘッダーの保護下にあります。図 19–3 のデータグラムは、トンネルモードでは外側の IPsec ヘッダー (この例では ESP) によって保護され、次の図のようになります。

図 19–6 トンネルモードで保護された IPsec パケット

暗号化部分

`ipsecconf` コマンドには、トンネルをトンネルモードまたはトランスポートモードで設定するためのキーワードが用意されています。

- ソケットごとのポリシーの詳細については、`ipsec(7P)` のマニュアルページを参照してください。
- ソケットごとのポリシーの例については、533 ページの「IPsec を使って Web 以外のトラフィックから Web サーバーを保護する方法」を参照してください。
- トンネルの詳細については、`ipsecconf(1M)` のマニュアルページを参照してください。
- トンネル構成の例については、553 ページの「IPv4 を使用するトンネルモードの IPsec トンネルで VPN を保護する方法」を参照してください。

仮想プライベートネットワークと IPsec

構成したトンネルは、ポイントツーポイントインタフェースです。トンネルによって、IP パケットを別の IP パケット内にカプセル化できます。トンネルの構成には、トンネルソースとトンネル宛先が必要です。詳細は、`tun(7M)` のマニュアルページと、IPv6 サポート用のトンネルの構成を参照してください。

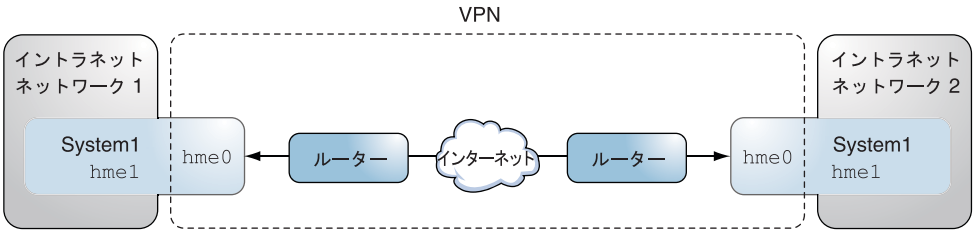
トンネルは、IP への物理インタフェースのようなものを作成します。この物理的リンクの完全性は、基本になるセキュリティープロトコルによって異なります。セキュリティーアソシエーション (SA) を確実に行えば、信頼性の高いトンネルになります。トンネルのデータパケットのソースはトンネル宛先で指定したピアでなけれ

ばなりません。この信頼関係があるかぎり、インタフェース別 IP 送信を利用して仮想プライベートネットワーク (VPN)を作成できます。

VPN に IPsec 保護を追加できます。IPsec が接続の安全性を確保します。たとえば、分離したネットワークを持つ複数のオフィスを VPN テクノロジを使用して接続している組織は、IPsec を追加して 2 つのオフィス間のトラフィックをセキュリティー保護できます。

次の図は、ネットワークシステムに配備した IPsec で、2 つのオフィスが VPN を形成する方法を示しています。

図 19–7　仮想プライベートネットワーク



設定手順の詳細な例については、553 ページの「IPv4 を使用するトンネルモードの IPsec トンネルで VPN を保護する方法」を参照してください。

IPv6 アドレスを使用する同様の例については、564 ページの「IPv6 を使用するトンネルモードの IPsec トンネルで VPN を保護する方法」を参照してください。

IPsec と NAT 越え

IKE は、NAT ボックスを通して IPsec SA とネゴシエートできます。この機能により、システムは、システムが NAT デバイスの背後にある場合も、リモートネットワークから安全に接続を行うことができます。たとえば、自宅で働く社員や会議場からログオンする社員も IPsec で自分のトラフィックを保護できます。

NAT は、Network Address Translation (ネットワークアドレス変換) の略語です。NAT ボックスは、プライベートな内部アドレスを一意のインターネットアドレスに変換します。NAT は、ホテルなどのインターネットへの公共のアクセスポイントでは非常によく使用されています。詳細は、667 ページの「IP フィルタの NAT 機能の使用」を参照してください。

NAT ボックスが通信システム間にある場合に IKE を使用する機能は、NAT traversal、または NAT-T と呼ばれます。Oracle Solaris 10 リリースでは、NAT-T には次の制限があります。

- NAT-T は、Sun Crypto Accelerator 4000 ボードが提供する IPsec ESP の高速化を利用できません。ただし、Sun Crypto Accelerator 4000 ボードによる IKE の高速化は可能です。
- AH プロトコルは不変の IP ヘッダーに依存しますので、AH を NAT-T と連係させることはできません。NAT-T を使用する場合は、ESPプロトコルが使用します。
- NAT ボックスには特別な処理規則はありません。特別な IPsec 処理規則を持つ NAT ボックスは、NAT-T の実装の障害となる場合があります。
- NAT-T が機能するのは、IKE イニシエータが NAT ボックスの背後にあるシステムの場合だけです。ボックスが、ボックスの背後の適切なシステム各自に IKE パケットを転送するようにプログラムされていない場合は、IKE の応答者が NAT ボックスの背後にいることはできません。

次の RFC は、NAT 機能と NAT-T の制限事項について説明しています。RFC のコピーは `http://www.rfc-editor.org` から取得できます。

- RFC 3022、『Traditional IP Network Address Translator (Traditional NAT)』、2001 年 1 月
- RFC 3715、『IPsec-Network Address Translation (NAT) Compatibility Requirements』、2004 年 3 月
- RFC 3947、『Negotiation of NAT-Traversal in the IKE』、2005 年 1 月
- RFC 3948、『UDP Encapsulation of IPsec Packets』、2005 年 1 月

NAT を通して IPsec を使用するには、633 ページの「移動体システム用の IKE の構成 (タスクマップ)」を参照してください。

IPsec と SCTP

Oracle Solaris は SCTP (Streams Control Transmission Protocol) をサポートしています。SCTP プロトコルと SCTP ポート番号を使用した IPsec ポリシーの指定はサポートされていますが、頑丈ではありません。RFC 3554 に指定されている SCTP の IPsec 拡張は、まだ実装されていません。これらの制限事項によって SCTP 向けの IPsec ポリシーの作成が複雑になる場合もあります。

SCTP は、単独の SCTP アソシエーションのコンテキストで、複数の発信元アドレスと宛先アドレスを利用できます。1 つの発信元アドレスまたは 1 つの宛先アドレスに IPsec ポリシーを適用すると、SCTP がそのアソシエーションの発信元アドレスまたは宛先アドレスを切り替えたときに、通信が失敗する恐れがあります。IPsec ポリシーは、元のアドレスしか認識しません。SCTP については、RFC および40 ページの「SCTP プロトコル」を参照してください。

IPsec と Oracle Solaris ゾーン

共有 IP ゾーンについては、IPsec の構成は大域ゾーンから行います。IPsec ポリシー構成ファイル `ipsecinit.conf` は、大域ゾーンだけに存在します。このファイルには、大域ゾーンに適用するエントリだけでなく、非大域ゾーンに適用するエントリも含めることができます。

排他的 IP ゾーンについては、IPsec は非大域ゾーンごとに構成されます。

IPsec をゾーンで使用する方法については、527 ページの「IPsec によるトラフィックの保護 (タスクマップ)」を参照してください。ゾーンについては、『System Administration Guide: Oracle Solaris Containers-Resource Management and Oracle Solaris Zones』の第 16 章「Introduction to Solaris Zones」を参照してください。

IPsec と論理ドメイン

IPsec は論理ドメインで動作します。論理ドメインは、IPsec を含む Oracle Solaris バージョン (Oracle Solaris 10 リリースなど) を実行している必要があります。

論理ドメインを作成するには、Oracle VM Server for SPARC (以前の名称は Logical Domains) を使用する必要があります。論理ドメインを構成する方法については、『Oracle VM Server for SPARC 2.2 管理ガイド』または『Oracle VM Server for SPARC 2.0 管理ガイド』を参照してください。

IPsec ユーティリティーおよび IPsec ファイル

表 19–3 は、IPsec を構成および管理するために使用するファイル、コマンド、およびサービス識別子について説明しています。完全性を期すために、鍵管理ファイルとコマンドも含めました。

Solaris 10 4/09 リリース以降では、IPsec は SMF によって管理されます。サービス識別子の詳細については、『Oracle Solaris の管理: 基本管理』の第 18 章「サービスの管理 (概要)」を参照してください。

- IPsec をネットワークに実装する手順については、527 ページの「IPsec によるトラフィックの保護 (タスクマップ)」を参照してください。
- IPsec ユーティリティーとファイルの詳細については、第 21 章「IP セキュリティーアーキテクチャー (リファレンス)」を参照してください。

表 19–3 選択される IPsec ユーティリティーとファイルのリスト

IPsec ユーティリティー、ファイル、またはサービス	説明	マニュアルページ
svc:/network/ipsec/ipsecalgs	現在のリリースでは、IPsec アルゴリズムを管理する SMF サービス。	ipsecalgs(1M)
svc:/network/ipsec/manual-key	現在のリリースでは、手動での鍵付き IPsec SA を管理する SMF サービス。	ipseckey(1M)
svc:/network/ipsec/policy	現在のリリースでは、IPsec ポリシーを管理する SMF。	smf(5)、ipsecconf(1M)
svc:/network/ipsec/ike	現在のリリースでは、IKE を使用した IPsec SA の自動管理のための SMF サービス。	smf(5)、in.iked(1M)
/etc/inet/ipsecinit.conf ファイル	IPsec ポリシーファイル。Solaris 10 4/09 リリースより前のリリースでは、このファイルが存在する場合、IPsec はブート時にアクティブになります。 現在のリリースでは、SMF policy サービスはシステムのブート時にこのファイルを使用して IPsec ポリシーを構成します。	ipsecconf(1M)
ipsecconf コマンド	IPsec ポリシーコマンド。現在の IPsec ポリシーの表示および変更や、テストを行うときに役立ちます。Solaris 10 4/09 リリースより前のリリースでは、ブートスクリプトは ipsecconf を使用して /etc/inet/ipsecinit.conf ファイルを読み込み、IPsec を有効にします。 現在のリリースでは、ipsecconf は、システムのブート時に IPsec ポリシーを構成するために SMF policy サービスで使用されます。	ipsecconf(1M)
PF_KEY ソケットインタフェース	SA データベース (SADB) のインタフェース。手動と自動の鍵管理を処理します。	pf_key(7P)
ipseckey コマンド	IPsec SA キー作成コマンド。ipseckey は、PF_KEY インタフェースに対するコマンド行フロントエンドです。ipseckey は、SA を作成、破棄、または修正できます。	ipseckey(1M)
/etc/inet/secret/ipseckeys ファイル	手動で鍵が設定された SA を含みます。Solaris 10 4/09 リリースより前のリリースでは、ipsecinit.conf ファイルが存在する場合、ブート時に ipseckeys ファイルが自動的に読み込まれます。 現在のリリースでは、ipseckeys は、システムのブート時に SA を手動で構成するために SMF manual-key サービスで使用されます。	

表 19–3 選択される IPsec ユーティリティーとファイルのリスト (続き)

IPsec ユーティリティー、ファイル、または サービス	説明	マニュアルページ
ipsecalgs コマンド	IPsec アルゴリズムコマンド。IPsec アルゴリズムとそのプロパティーの一覧を参照および変更するときに役立ちます。 現在のリリースでは、システムのブート時に既知の IPsec アルゴリズムをカーネルと同期するために SMF ipsecalgs サービスで使用されます。	ipsecalgs(1M)
/etc/inet/ipsecalgs ファイル	構成されている IPsecプロトコルとアルゴリズム定義を含みます。このファイルは、ipsecalgs コマンドによって管理されます。手動では絶対に編集しないでください。	
/etc/inet/ike/config ファイル	IKE の構成とポリシーファイル。デフォルトでは、このファイルはありません。Solaris 10 4/09 リリースより前のリリースでは、このファイルが存在する場合、IKE デーモン in.iked は自動鍵管理を提供します。/etc/inet/ike/config ファイル内の規則およびグローバルパラメータに基づいて鍵管理が行われます。600 ページの「IKE ユーティリティーおよび IKE ファイル」を参照してください。 現在のリリースでは、このファイルが存在する場合、svc:/network/ipsec/ike サービスは IKE デーモン in.iked を起動して自動鍵管理を提供します。	ike.config(4)

Oracle Solaris 10 リリースでの IPsec の変更点

Oracle Solaris の新機能の完全な一覧については、『Oracle Solaris 10 1/13 の新機能』を参照してください。Solaris 9 リリースから、IPsec には次の機能が追加されました。

- Sun Crypto Accelerator 4000 ボードを接続すると、ボードは、ボードの Ethernet インタフェースを使用するパケットの IPsec SA を自動的にキャッシュします。ボードは、IPsec SA の処理速度も速めます。
- IPsec は、IPv6 ネットワーク上の IKE での自動鍵管理を利用できます。詳細については、第 22 章「インターネット鍵交換 (概要)」を参照してください。

  IKE の新機能については、601 ページの「Oracle Solaris 10 リリースにおける IKE の変更」を参照してください。
- ipseckey コマンドのパーサは、より明確なヘルプを提供します。ipseckey monitor コマンドは、各イベントにタイムスタンプをつけます。詳細については、ipseckey(1M) のマニュアルページを参照してください。
- IPsec アルゴリズムが中央の格納場所である Oracle Solaris の暗号フレームワーク機能から実行されます。ipsecalgs(1M) のマニュアルページは、使用可能なアルゴリズムの特徴を説明しています。アルゴリズムは、実行するアーキテクチャーに対して最適化されます。暗号化フレームワークの説明については、『Solaris のシ

ステム管理: セキュリティーサービス』の第 13 章「Oracle Solaris の暗号化フレームワーク (概要)」を参照してください。

- IPsec は、大域ゾーンで動作します。IPsec ポリシーは、非大域ゾーンに対して大域ゾーンで管理されます。非大域ゾーンのキー作成素材は、大域ゾーンで作成され、手動で管理されます。IKE は、非大域ゾーンのキーの生成には使用できません。ゾーンの詳細については、『System Administration Guide: Oracle Solaris Containers-Resource Management and Oracle Solaris Zones』の第 16 章「Introduction to Solaris Zones」を参照してください。
- IPsec ポリシーは、SCTP (Streams Control Transmission Protocol) と SCTP ポート番号と組み合わせて使用できます。ただし、実装は不完全です。RFC 3554 に指定されている SCTP の IPsec 拡張は、まだ実装されていません。これらの制限事項によって SCTP 向けの IPsec ポリシーの作成が複雑になる場合もあります。詳細は、RFC を参照してください。また、522 ページの「IPsec と SCTP」および 40 ページの「SCTP プロトコル」をお読みください。
- IPsec と IKE は、NAT ボックスの背後で発生したトラフィックを保護します。詳細と制限については、521 ページの「IPsec と NAT 越え」を参照してください。手順については、633 ページの「移動体システム用の IKE の構成 (タスクマップ)」を参照してください。

◆ ◆ ◆ 第 20 章

IPsecの構成(タスク)

この章では、ネットワークにIPsecを実装する手順について説明します。この手順は、次のタスクマップで説明されています。

- 527ページの「IPsecによるトラフィックの保護(タスクマップ)」
- 550ページの「IPsecによるVPNの保護(タスクマップ)」

IPsecの概要については、第19章「IPセキュリティーアーキテクチャー(概要)」を参照してください。IPsecの参考情報については、第21章「IPセキュリティーアーキテクチャー(リファレンス)」を参照してください。

IPsecによるトラフィックの保護(タスクマップ)

次のタスクマップに、1台以上のシステム間でIPsecを設定する手順を示します。`ipsecconf`(1M)、`ipseckey`(1M)、および`ifconfig`(1M)のマニュアルページも、それぞれの「例」で役立つ手順を説明しています。

タスク	説明	参照先
システム間のトラフィックを保護します。	あるシステムから別のシステムへのパケットを保護します。	529ページの「IPsecで2つのシステム間のトラフィックを保護するには」
IPsecポリシーによるWebサーバーを保護します。	Web以外のトラフィックにIPsecの使用を求めます。Webクライアントは、IPsecチェックをバイパスする特定のポートによって識別されます。	533ページの「IPsecを使ってWeb以外のトラフィックからWebサーバーを保護する方法」
IPsecポリシーの表示	現在実施されているIPsecポリシーを、ポリシーの実行順序に従って表示します。	536ページの「IPsecポリシーを表示するには」

タスク	説明	参照先
乱数を生成します。	手動で作成するセキュリティーアソシエーションのためにキー作成素材用の乱数を生成します。	537 ページの「Oracle Solaris System で乱数を生成する方法」 『Solaris のシステム管理: セキュリティーサービス』の「pktool コマンドを使用して対称鍵を生成する方法」
手動によるセキュリティーアソシエーションの作成または置き換えを行います。	次のようなセキュリティーアソシエーション向けのローデータを提供します。 ■ IPsec アルゴリズム名とキー作成素材 ■ セキュリティーパラメータインデックス (SPI) ■ 発信元と着信先の IP アドレス、およびその他のパラメータ	538 ページの「IPsec セキュリティーアソシエーションを手動で作成する方法」
IPsec がパケットを保護しているかどうかを検査します。	snoop の出力を調べ、IP データグラムがどのように保護されているかを示すヘッダーをチェックします。	543 ページの「IPsec によってパケットが保護されていることを確認する方法」
(任意) ネットワークセキュリティーの役割を作成します。	セキュリティーネットワークを設定できるが、スーパーユーザーより権限が少ない役割を作成します。	545 ページの「ネットワークセキュリティーの役割を構成する方法」
IPsec と鍵情報を一連の SMF サービスとして管理します。	サービスの有効化、無効化、リフレッシュ、および再起動を行うコマンドを、いつどのように使用するかについて説明します。サービスのプロパティー値を変更するコマンドについても説明します。	546 ページの「IKE および IPsec サービスを管理する方法」
セキュアな仮想プライベートネットワーク (VPN) を設定します。	インターネット経由の 2 つのシステム間で IPsec を設定します。	550 ページの「IPsec による VPN の保護 (タスクマップ)」

IPsec によるトラフィックの保護

このセクションでは、2 つのシステム間のトラフィックを保護する手順と、Web サーバーを保護する手順について説明します。VPN を保護する手順については、550 ページの「IPsec による VPN の保護 (タスクマップ)」を参照してください。追加手順では、鍵情報とセキュリティーアソシエーションの提供、IPsec が構成どおり動作しているかどうかの確認を行います。

次の情報は、すべての IPsec 構成タスクで使用されます。

- **IPsec とゾーン –** 共有 IP 非大域ゾーンの IPsec ポリシーと鍵を管理するには、大域ゾーンで IPsec ポリシーファイルを作成し、大域ゾーンで IPsec 構成コマンドを実行します。構成中の非大域ゾーンに対応する発信元アドレスを使用してください。大域ゾーンでは、大域ゾーンの IPsec ポリシーとキーも構成できます。排他的 IP ゾーンについては、非大域ゾーンで IPsec ポリシーを構成します。Solaris 10 7/07 リリース以降では、非大域ゾーンのキーの管理に IKE を使用できます。
- **IPsec と RBAC –** IPsec を管理する役割を使用する方法については、『Solaris のシステム管理: セキュリティーサービス』の第 9 章「役割に基づくアクセス制御の使用 (タスク)」を参照してください。例については、545 ページの「ネットワークセキュリティーの役割を構成する方法」を参照してください。
- **IPsec と SCTP –** IPsec は、Streams Control Transmission Protocol (SCTP) アソシエーションを保護するのに使用できますが、注意が必要です。詳細は、522 ページの「IPsec と SCTP」を参照してください。

▼ IPsec で 2 つのシステム間のトラフィックを保護するには

この手順では、次の設定がすでになされているものとします。

- 2 つのシステムが `enigma` および `partym` と名付けられている。
- 各システムには、2 つのアドレス (IPv4 アドレスと IPv6 アドレス) がある。
- 各システムには、AES アルゴリズムを使用した ESP 暗号化 (128 ビットのキーが必要) と、SHA1 メッセージ要約を使用した ESP 認証 (160 ビットのキーが必要) が必要です。
- 各システムは、共有セキュリティーアソシエーションを使用します。

  共有セキュリティーアソシエーションでは、2 つのシステムを保護するのに必要なのは 1 組だけの SA です。

始める前に　システムまたは共有 IP ゾーンの IPsec ポリシーの構成は、大域ゾーンで行う必要があります。排他的 IP ゾーンについては、非大域ゾーンで IPsec ポリシーを構成します。

1. システムコンソール上で、**Primary Administrator** の役割を引き受けるか、スーパーユーザーになります。

   Primary Administrator 役割には、Primary Administrator プロファイルが含まれます。役割を作成してユーザーに役割を割り当てるには、『Oracle Solaris の管理: 基本管理』の第 2 章「Solaris 管理コンソールの操作 (タスク)」を参照してください。

注 – リモートログインすると、セキュリティー上重要なトラフィックが盗聴される可能性があります。何らかの方法でリモートログインを保護していても、システムのセキュリティーがリモートログインセッションレベルに低下します。セキュリティー保護されたリモートログインには、`ssh` コマンドを使用してください。例については、例 20–1 を参照してください。

2 各システムでホストエントリを確認します。

現在のリリースでは、`/etc/inet/hosts` ファイルにホストエントリを追加します。

Solaris 10 7/07 リリースより前のリリースを実行しているシステムでは、`/etc/inet/ipnodes` ファイルに IPv4 と IPv6 のエントリを追加します。次のように、1 つのシステムのエントリは連続してそのファイルに入力します。システムの構成ファイルについては、241 ページの「TCP/IP 構成ファイル」および第 11 章「IPv6 の詳細 (リファレンス)」を参照してください。

IPv4 アドレスだけを持つシステムどうしを接続する場合は、`/etc/inet/hosts` ファイルに変更を加えます。この例で接続するシステムはどちらも、以前の Solaris リリースを実行しており、IPv6 アドレスを使用しています。

a. **enigma** という名前のシステムでは、**hosts** または **ipnodes** ファイルに次のように入力します。

```
# Secure communication with partym
192.168.13.213 partym
2001::eeee:3333:3333 partym
```

b. **partym** という名前のシステムでは、**hosts** または **ipnodes** ファイルに次のように入力します。

```
# Secure communication with enigma
192.168.116.16 enigma
2001::aaaa:6666:6666 enigma
```

記号名にネームサービスを使用するのは安全ではありません。

3 各システムで **IPsec** ポリシーファイルを作成します。

ファイル名は `/etc/inet/ipsecinit.conf` です。例は、`/etc/inet/ipsecinit.sample` ファイルを参照してください。

4 **IPsec** ポリシーエントリを **ipsecinit.conf** ファイルに追加します。

a. **enigma** システムで、次のポリシーを追加します。

```
{laddr enigma raddr partym} ipsec {encr_algs aes encr_auth_algs sha1 sa shared}
```

b. **partym** システムで、同じポリシーを追加します。

```
{laddr partym raddr enigma} ipsec {encr_algs aes encr_auth_algs sha1 sa shared}
```

IPsec ポリシーエントリの構文については、ipsecconf(1M)のマニュアルページを参照してください。

5 各システムで、2 つのシステム間に IPsec SA ペアを追加します。

インターネットキー交換 (IKE) を構成すると、SA が自動的に生成されます。SA は手動でも追加できます。

注 – キーの生成や保守を手動で行う必要が特にない場合は、IKE を使用すべきです。IKE 鍵管理では、手動での鍵管理よりも強力なセキュリティー効果が得られます。

- 603 ページの「IKE の構成 (タスクマップ)」の構成手順のいずれかに従って、IKE を構成します。IKE 構成ファイルの構文については、ike.config(4) のマニュアルページを参照してください。
- SA を手動で追加する場合は、538 ページの「IPsec セキュリティーアソシエーションを手動で作成する方法」を参照してください。

6 IPsec ポリシーを有効にします。

- Solaris 10 4/09 リリースより前のリリースを実行している場合は、システムをリブートします。

  ```
  # init 6
  ```

  その後、543 ページの「IPsec によってパケットが保護されていることを確認する方法」に進みます。

- Solaris 10 4/09 リリース以降では、IPsec サービスをリフレッシュし、鍵管理サービスを有効にします。

  手順 7 から手順 10 までの手順を完了します。

7 IPsec ポリシーファイルの構文を確認します。

```
# ipsecconf -c -f /etc/inet/ipsecinit.conf
```

エラーがあれば修正し、ファイルの構文を確認してから続行します。

8 IPsec ポリシーをリフレッシュします。

```
# svcadm refresh svc:/network/ipsec/policy:default
```

IPsec ポリシーはデフォルトで有効になっているので、「リフレッシュ」を行います。IPsec ポリシーを無効にしてある場合は有効にしてください。

```
# svcadm enable svc:/network/ipsec/policy:default
```

9 **IPsec 用の鍵を有効にします。**

- 手順 5 で **IKE** を構成した場合は、次のいずれかを実行します。
  - **ike** サービスが有効になっていない場合は有効にします。

    ```
    # svcadm enable svc:/network/ipsec/ike:default
    ```

  - **ike** サービスが有効になっている場合は再起動します。

    ```
    # svcadm restart svc:/network/ipsec/ike:default
    ```

- 手順 5 で鍵を手動で構成した場合は、次のいずれかを実行します。
  - **manual-key** サービスが有効になっていない場合は有効にします。

    ```
    # svcadm enable svc:/network/ipsec/manual-key:default
    ```

  - **manual-key** サービスが有効になっている場合はリフレッシュします。

    ```
    # svcadm refresh svc:/network/ipsec/manual-key:default
    ```

10 パケットが保護されていることを確認します。

手順については、543 ページの「IPsec によってパケットが保護されていることを確認する方法」を参照してください。

例 20–1 ssh 接続を使用している場合に IPsec ポリシーを追加する

この例では、管理者はスーパーユーザーとして 2 つのシステムの IPsec ポリシーと鍵を構成します。その際、ssh コマンドを使用して 2 番目のシステムにアクセスします。詳細は、ssh(1) のマニュアルページを参照してください。

- まず、前の手順の手順 2 から 手順 5 までを実行して、最初のシステムを構成します。
- 次に、別の端末ウィンドウで、ssh コマンドを使用して 2 番目のシステムにログインします。

  ```
  local-system # ssh other-system
  other-system #
  ```

- ssh セッションの端末ウィンドウで、手順 2 から 手順 6 までを実行して、2 番目のシステムの IPsec ポリシーと鍵を構成します。
- ここで ssh セッションを終了します。

  ```
  other-system # exit
  local-system #
  ```

- 最後に、手順 6 を実行して、最初のシステムの IPsec ポリシーを有効にします。

2 つのシステムが次に通信を行うとき、ssh 接続を使用した通信も含め、通信は IPsec で保護されます。

例 20–2 リブートを行わないで IPsec によるトラフィックの保護

次の例は、Solaris 10 4/09 リリースより前のリリースを実行している場合に役立ちます。つまり、そのようなリリースでは、IPsec はサービスとして管理されません。この例では、テスト環境での IPsec の実装方法を示します。実際の稼働環境では、`ipsecconf` コマンドを実行するよりもリブートする方が安全です。セキュリティー上の考慮事項については、この例の最後を参照してください。

手順 6 でリブートする代わりに、次のオプションのいずれかを選択します。

- IKE を使って鍵情報を作成した場合は、`in.iked` デーモンをいったん停止後、再起動します。

```
# pkill in.iked
# /usr/lib/inet/in.iked
```

- キーを手動で追加した場合は、`ipseckey` コマンドを使用して、データベースに SA を追加します。

```
# ipseckey -c -f /etc/inet/secret/ipseckeys
```

続いて、`ipsecconf` コマンドを使用して IPsec ポリシーを有効にします。

```
# ipsecconf -a /etc/inet/ipsecinit.conf
```

セキュリティーについて – `ipsecconf` コマンドを実行するときには、警告を読んでください。ソケットがすでにラッチされている (使用されている) 場合には、システムへ侵入される恐れがあります。詳細については、588 ページの「ipsecinit.conf と ipsecconf のセキュリティーについて」を参照してください。

▼ IPsec を使って Web 以外のトラフィックから Web サーバーを保護する方法

セキュアな Web サーバーでは、Web クライアントであれば Web サービスと通信できます。セキュアな Web サーバーでは、Web トラフィック以外のトラフィックは、セキュリティー検査を通る必要があります。次の手順には、Web トラフィックの検査省略手順が含まれています。さらに、この Web サーバーでは、セキュアでない DNS クライアント要求を出すことができます。その他のすべてのトラフィックでは、AES と SHA-1 アルゴリズムによる ESP が必要です。

始める前に IPsec ポリシーの構成は大域ゾーンで行う必要があります。排他的 IP ゾーンについては、非大域ゾーンで IPsec ポリシーを構成します。

529 ページの「IPsec で 2 つのシステム間のトラフィックを保護するには」を完了して、次の条件が成立しています。

- 2 つのシステム間の通信は IPsec で保護されています。
- 鍵情報は手動または IKE によって生成されます。

- パケットが保護されていることを確認してあります。

1 システムコンソール上で、**Primary Administrator** の役割を引き受けるか、スーパーユーザーになります。

Primary Administrator 役割には、Primary Administrator プロファイルが含まれます。役割を作成してユーザーに役割を割り当てるには、『Oracle Solaris の管理: 基本管理』の第 2 章「Solaris 管理コンソールの操作 (タスク)」を参照してください。

注 – リモートログインすると、セキュリティー上重要なトラフィックが盗聴される恐れがあります。何らかの方法でリモートログインを保護していても、システムのセキュリティーがリモートログインセッションレベルに低下します。セキュリティー保護されたリモートログインには、`ssh` コマンドを使用してください。

---

2 セキュリティーポリシー検査を省略するサービスを指定します。

Web サーバーの場合、TCP ポート 80 (HTTP) と 443 (保護 HTTP) が該当します。Web サーバーが DNS 名検査をするときは、TCP と UDP の両方にポート 53 も組み込む必要がある場合もあります。

3 **Web** サーバーの **IPsec** ポリシーを作成し、有効にします。

- **Solaris 10 4/09** リリース以降では、手順 **4** から手順 **7** までを実行します。
- **Solaris 10 4/09** リリースより前のリリースを実行している場合は、手順 **8** から手順 **11** までを実行します。

手順 12 はすべての Oracle Solaris リリースでオプションです。

4 **Web** サーバーのポリシーを **IPsec** ポリシーファイルに追加します。

`/etc/inet/ipsecinit.conf` ファイルに次の行を追加します。

```
# Web traffic that web server should bypass.
{lport  80 ulp tcp dir both} bypass {}
{lport 443 ulp tcp dir both} bypass {}

# Outbound DNS lookups should also be bypassed.
{rport 53 dir both} bypass {}

# Require all other traffic to use ESP with AES and SHA-1.
# Use a unique SA for outbound traffic from the port
{} ipsec {encr_algs aes encr_auth_algs sha1 sa shared}
```

これで、保護トラフィックだけがシステムへのアクセスを許可されます。ただし、手順 4 で説明した、検査を省略するトラフィックは例外です。

5 **IPsec** ポリシーファイルの構文を確認します。

```
# ipsecconf -c -f /etc/inet/ipsecinit.conf
```

6 **IPsec** ポリシーをリフレッシュします。

```
# svcadm refresh svc:/network/ipsec/policy:default
```

7 **IPsec** 用の鍵をリフレッシュします。

- 529 ページの「**IPsec** で 2 つのシステム間のトラフィックを保護するには」の手順 5 で **IKE** を構成した場合は、**ike** サービスを再起動します。

  ```
  # svcadm restart svc:/network/ipsec/ike
  ```

- 529 ページの「**IPsec** で 2 つのシステム間のトラフィックを保護するには」の手順 5 で鍵を手動で構成した場合は、**manual-key** サービスをリフレッシュします。

  ```
  # svcadm refresh svc:/network/ipsec/manual-key:default
  ```

これで設定が完了しました。必要に応じて、手順 12 を実行します。

8 **Web** サーバーポリシー用のファイルを **/etc/inet** ディレクトリに作成します。

注 – 次の手順は、Solaris 10 4/09 リリースより前のリリースを実行している Web サーバーを構成するためのものです。

このファイルにその目的を表す名前を与えます (たとえば、`IPsecWebInitFile`)。このファイルに次のように入力します。

```
# Web traffic that web server should bypass.
{lport  80 ulp tcp dir both} bypass {}
{lport 443 ulp tcp dir both} bypass {}

# Outbound DNS lookups should also be bypassed.
{rport 53 dir both} bypass {}

# Require all other traffic to use ESP with AES and SHA-1.
# Use a unique SA for outbound traffic from the port
{} ipsec {encr_algs aes encr_auth_algs sha1 sa shared}
```

これで、保護トラフィックだけがシステムへのアクセスを許可されます。ただし、手順 4 で説明した、検査を省略するトラフィックは例外です。

9 手順 8 で作成したファイルの内容を **/etc/inet/ipsecinit.conf** ファイルにコピーします。

10 **IPsecWebInitFile** ファイルを読み取り専用アクセス権で保護します。

```
# chmod 400 IPsecWebInitFile
```

11 リブートせずに **Web** サーバーをセキュリティー保護します。

次のオプションのいずれかを選択します。

- 鍵管理に IKE を使用する場合は、`in.iked` デーモンをいったん停止後、再起動します。

```
# pkill in.iked
# /usr/lib/inet/in.iked
```

- 手動でキーを管理する場合は、ipseckey および ipsecconf コマンドを実行します。

  ipsecconf コマンドの引数には、 IPsecWebInitFile を使用します。引数に ipsecinit.conf ファイルを使用すると、ipsecconf コマンドは、ファイル内のポリシーがすでにシステムに実装されている場合は、エラーを生成します。

```
# ipseckey -c -f /etc/inet/secret/ipseckeys
# ipsecconf -a /etc/inet/IPsecWebInitFile
```

注意 – ipsecconf コマンドの実行時には警告を読んでください。ソケットがすでにラッチされている (使用されている) 場合には、システムへ侵入される恐れがあります。詳細については、588 ページの「ipsecinit.conf と ipsecconf のセキュリティーについて」を参照してください。in.iked デーモンの再起動時にも、同じ警告が表示されます。

リブートすることもできます。システムをリブートすると、IPsec ポリシーがすべての TCP 接続に適用されます。リブート時に、IPsec ポリシーのファイルで指定したポリシーが TCP 接続で使用されます。

12 **(省略可能) Web 以外のトラフィックのために Web サーバーと通信する場合は、リモートシステムを有効にします。**

リモートシステムの ipsecinit.conf ファイルに次のポリシーを入力します。

```
# Communicate with web server about nonweb stuff
#
{laddr webserver} ipsec {encr_algs aes encr_auth_algs sha1 sa shared}
```

IPsec ポリシーが一致した場合にかぎり、リモートシステムは、非 Web トラフィックを持つ Web サーバーと安全に通信できます。

▼ IPsec ポリシーを表示するには

引数を指定しないで ipsecconf コマンドを実行すると、システムに構成されているポリシーを確認できます。

始める前に　ipsecconf コマンドは大域ゾーンで実行する必要があります。排他的 IP ゾーンについては、非大域ゾーンで ipsecconf コマンドを実行します。

1 **Network IPsec Management プロファイルを持つ役割またはスーパーユーザーになります。**

Solaris 10 4/09 リリースより前のリリースを実行している場合、Network IPsec Management プロファイルは使用できません。Network Security プロファイルを使用してください。

ネットワークセキュリティープロファイルを含む役割を作成し、その役割をユーザーに割り当てるには、545 ページの「ネットワークセキュリティーの役割を構成する方法」を参照してください。

2 **IPsec ポリシーの表示**

a. 追加された順序でグローバルな **IPsec** ポリシーエントリを表示します。

```
$ ipsecconf
```

各エントリが、「インデックス」とそのあとに番号が付いて表示されます。

b. 一致した順序で **IPsec** ポリシーエントリを表示します。

```
$ ipsecconf -l -n
```

c. トンネルごとのエントリも含め、**IPsec** ポリシーエントリを一致した順序で表示します。

```
$ ipsecconf -L -n
```

▼ Oracle Solaris System で乱数を生成する方法

キーを手動で指定する場合、鍵情報はランダムでなければなりません。IPsec 鍵用の鍵情報は 16 進形式です。ほかのオペレーティングシステムでは、ASCII 形式の鍵情報が必要になる場合があります。ASCII 形式を必要とするオペレーティングシステムと通信する Oracle Solaris システム用に鍵情報を生成する方法については、例 23–1 を参照してください。

乱数発生関数がすでにある場合は、それを使用してください。それ以外の場合、`/dev/random` デバイスを入力として `od` コマンドを使用できます。詳細は、`od(1)` のマニュアルページを参照してください。

Solaris 10 4/09 リリースでは、`pktool` コマンドも使用できます。このコマンドの構文は、`od` コマンドの構文より単純です。詳細は、『Solaris のシステム管理: セキュリティーサービス』の「pktool コマンドを使用して対称鍵を生成する方法」を参照してください。

1 **16 進数の乱数を生成します。**

```
% od -x|-X -A n file | head -n
```

`-x` 8 進数ダンプを 16 進数形式で表示します。16 進数形式は鍵情報を表すのに役立ちます。16 進数を 4 文字単位で表示します。

`-X` 8 進数ダンプを 16 進数形式で表示します。16 進数を 8 文字単位で表示します。

`-A n` 表示から入力オフセットベースを取り除きます。

file 乱数のソースとなります。

`head -n` 出力の冒頭の *n* 行に表示を限定します。

2 これらの出力を組み合わせて、適切な長さのキーを作成します。

同じ行にある乱数間のスペースを取り除き、32 文字キーを作成します。32 文字キーの長さは 128 ビットです。セキュリティーパラメータインデックス (SPI) の場合は、8 文字キー 1 個を使用します。そのキーは、`0x` 接頭辞を使用します。

例 20–3 IPsec のキー素材の生成

次の例は、8 つ 16 進数文字のグループそれぞれに 2 行のキーを表示します。

```
% od -X -A n /dev/random | head -2
        d54d1536 4a3e0352 0faf93bd 24fd6cad
        8ecc2670 f3447465 20db0b0c c83f5a4b
```

最初の行で 4 つの数字を組み合わせることで、32 文字のキーを作成できます。`0x` に続く 8 文字の数字は、`0xf3447465` などの適切な SPI 値を提供します。

次の例は、4 つ 16 進数文字のグループそれぞれに 2 行のキーを表示します。

```
% od -x -A n /dev/random | head -2
        34ce 56b2 8b1b 3677 9231 42e9 80b0 c673
        2f74 2817 8026 df68 12f4 905a db3d ef27
```

最初の行で 4 つの数字を組み合わせることで、32 文字のキーを作成できます。

▼ IPsec セキュリティーアソシエーションを手動で作成する方法

次の手順では、529 ページの「IPsec で 2 つのシステム間のトラフィックを保護するには」の手順で使用するキー作成素材を提供します。`partym` と `enigma` という 2 つのシステムの鍵を生成しようとしています。一方のシステムで鍵を生成してから、このシステムの鍵を両方のシステムで使用します。

始める前に 共有 IP ゾーンのキー情報の手動管理は、大域ゾーンで行う必要があります。

1 SA の鍵情報を生成します。

16 進のアウトバウンドトラフィックと、同じく 16 進のインバウンドトラフィックには、それぞれ 3 種類の乱数が必要です。

つまり、1 台のシステムで次の数値を生成する必要があります。

- spi キーワードの値として、2 つの 16 進数の乱数。1 つはアウトバウンドトラフィック用です。もう 1 つはインバウンドトラフィック用です。それぞれの乱数の最大桁数は 8 桁です。
- 認証の SHA1 アルゴリズム用として、2 つの 16 進数の乱数。160 ビットのキーの場合、各乱数は 40 桁でなければなりません。1 つは dst enigma 用です。もう 1 つは dst partym 用です。
- ESP 暗号化の AES アルゴリズム用として、2 つの 16 進数の乱数。256 ビットのキーの場合、各乱数は 64 桁でなければなりません。1 つは dst enigma 用です。もう 1 つは dst partym 用です。

乱数発生関数がすでにある場合は、それを使用してください。ない場合は、od コマンドを使用できます。手順については、537 ページの「Oracle Solaris System で乱数を生成する方法」を参照してください。

2 いずれかのシステムのシステムコンソールで、**Primary Administrator** の役割を引き受けるか、スーパーユーザーになります。

Primary Administrator 役割には、Primary Administrator プロファイルが含まれます。役割を作成してユーザーに役割を割り当てるには、『Oracle Solaris の管理: 基本管理』の第 2 章「Solaris 管理コンソールの操作 (タスク)」を参照してください。

注 – リモートログインすると、セキュリティー上重要なトラフィックが盗聴される恐れがあります。何らかの方法でリモートログインを保護していても、システムのセキュリティーがリモートログインセッションレベルに低下します。セキュリティー保護されたリモートログインには、ssh コマンドを使用してください。

---

3 **SA** を作成します。

- **Solaris 10 4/09** リリース以降では、手順 **8** から手順 **10** までを実行します。
- **Solaris 10 4/09** リリースより前のリリースを実行している場合は、手順 **4** から手順 **9** までを実行します。

4 **ipseckey** コマンドモードを有効にします。

```
# ipseckey

>
```

> プロンプトは、ipseckey コマンドモードであることを示します。

5 既存の **SA** を置き換える場合は、現在の **SA** を消去してください。

```
> flush
>
```

攻撃者に SA を破る時間を与えないように、キー作成素材は置き換えが必要です。

注 – 管理者は、通信システム上のキーの置き換えを調整する必要があります。あるシステムの SA を置き換える場合は、それと通信しているリモートシステムの SA も置き換える必要があります。

---

6 **SA** を作成するには、次のコマンドを実行します。

```
> add protocol spi random-hex-string \
src addr dst addr2 \
protocol-prefix_alg protocol-algorithm  \
protocol-prefixkey random-hex-string-of-algorithm-specified-length
```

次の構文で、フラッシュした SA を置き換えることもできます。

protocol
esp または ah を指定します。

random-hex-string
16 進数形式の最大 8 桁の乱数を指定します。0x が前置されます。セキュリティーパラメータインデックス (SPI) が受け取る以上の桁数を入力すると、超過部分は無視されます。SPI が受け取るより少ない桁数を入力すると、パディングが行われます。

addr
システムの IP アドレスを指定します。

addr2
addr のピアシステムの IP アドレスを指定します。

protocol-prefix
encr または auth を指定します。encr 接頭辞は esp プロトコルとともに使用されます。auth 接頭辞は ah プロトコルとともに、そして、esp プロトコルを認証する場合に使用されます。

protocol-algorithm
ESP または AH のアルゴリズムを指定します。それぞれのアルゴリズムには、特定の長さのキーが必要です。

認証アルゴリズムには MD5 と SHA1 があります。Solaris 10 4/09 リリース以降では、SHA256 と SHA512 がサポートされています。暗号化アルゴリズムには、DES、3DES、AES、および Blowfish があります。

random-hex-string-of-algorithm-specified-length
アルゴリズムによって必要とされる長さをもつ 16 進数の乱数を指定します。たとえば、MD5 アルゴリズムでは、128 ビットキーのため 32 桁の乱数が必要で

す。3DES アルゴリズムでは、192 ビットキーのため 48 桁の乱数が必要です。

a. たとえば、**enigma** システムで出力パケットを保護します。
手順 1 で生成した乱数を使用します。

Solaris 10 1/06 の場合:

```
> add esp spi 0x8bcd1407 \
src 192.168.116.16 dst 192.168.13.213 \
encr_alg aes \
auth_alg sha1 \
encrkey c0c65b888c2ee301c84245c3da63127e92b2676105d5330e85327c1442f37d49 \
authkey 6fab07fec4f2895445500ed992ab48835b9286ff
>
```

注 – ピアシステムでは、同じ鍵情報および同じ SPI を使用する必要があります。

b. **enigma** システムの **ipseckey** コマンドモードでは、入力パケットを保護します。
パケットを保護するには、次のコマンドを入力します。

```
> add esp spi 0x122a43e4 \
src 192.168.13.213 dst 192.168.116.16 \
encr_alg aes \
auth_alg sha1 \
encrkey a2ea934cd62ca7fa14907cb2ad189b68e4d18c976c14f22b30829e4b1ea4d2ae \
authkey c80984bc4733cc0b7c228b9b74b988d2b7467745
>
```

注 – これらのキーと SPI は、SA ごとに変更できます。SA ごとに異なるキーと異なる SPI を割り当てるべきです。

7 **ipseckey** コマンドモードを終了するには、**Control-D** キーを押すか、**quit** と入力します。

8 **/etc/inet/secret/ipseckeys** ファイルに鍵情報を追加します。
Solaris 10 4/09 リリースより前のリリースでは、この手順により、リブート時に IPsec で鍵情報を確実に使用できるようになります。

`/etc/inet/secret/ipseckeys` ファイルの行と `ipseckey` コマンド行の言語が同じになるようにします。

a. たとえば、**enigma** システム上の **/etc/inet/secret/ipseckeys** ファイルは次のようになります。

```
# ipseckeys - This file takes the file format documented in
#   ipseckey(1m).
#   Note that naming services might not be available when this file
#   loads, just like ipsecinit.conf.
#
# for outbound packets on enigma
```

```
   add esp spi 0x8bcd1407 \
      src 192.168.116.16 dst 192.168.13.213  \
      encr_alg aes \
      auth_alg sha1  \
      encrkey  c0c65b888c2ee301c84245c3da63127e92b2676105d5330e85327c1442f37d49 \
      authkey  6fab07fec4f2895445500ed992ab48835b9286ff
   #
   # for inbound packets
   add esp spi 0x122a43e4 \
      src 192.168.13.213 dst 192.168.116.16 \
      encr_alg aes \
      auth_alg sha1  \
      encrkey a2ea934cd62ca7fa14907cb2ad189b68e4d18c976c14f22b30829e4b1ea4d2ae \
      authkey c80984bc4733cc0b7c228b9b74b988d2b7467745
```

b. 読み取り専用ファイルを保護します。

```
# chmod 400 /etc/inet/secret/ipseckeys
```

9 **partym** システムでこの手順を繰り返します。

enigma システムの場合と同じ鍵情報を使用します。

両システムの鍵情報は同じでなければなりません。次の例のように、ipseckeys ファイル内のコメントだけが異なります。コメントが異なるのは、dst enigma が enigma システム上ではインバウンド、partym システム上ではアウトバウンドになるからです。

```
# partym ipseckeys file
#
# for inbound packets
add esp spi 0x8bcd1407 \
   src 192.168.116.16 dst 192.168.13.213  \
   encr_alg aes \
   auth_alg sha1  \
   encrkey  c0c65b888c2ee301c84245c3da63127e92b2676105d5330e85327c1442f37d49 \
   authkey  6fab07fec4f2895445500ed992ab48835b9286ff
#
# for outbound packets
add esp spi 0x122a43e4 \
   src 192.168.13.213 dst 192.168.116.16 \
   encr_alg aes \
   auth_alg sha1  \
   encrkey a2ea934cd62ca7fa14907cb2ad189b68e4d18c976c14f22b30829e4b1ea4d2ae \
   authkey c80984bc4733cc0b7c228b9b74b988d2b7467745
```

10 **manual-key** サービスを有効にします。

```
# svcadm enable svc:/network/ipsec/manual-key
```

現在のリリースで鍵を置き換える方法については、例 20–4 を参照してください。

例 20–4 IPsec SA を置き換える

この例では、管理者が現在の Oracle Solaris 10 リリースを実行しているシステムを構成しています。管理者は新しい鍵を生成し、ipseckeys ファイルの鍵情報を変更してから、サービスを再起動します。

- まず、管理者は 537 ページの「Oracle Solaris System で乱数を生成する方法」を実行して鍵を生成します。
- 次に、生成された鍵を /etc/inet/secret/ipseckeys ファイルで使用します。

  管理者は同じアルゴリズムを使用しました。そのため、SPI、encrkey、および authkey の値だけを変更します。

```
add esp spi 0x8xzy1492 \
   src 192.168.116.16 dst 192.168.13.213  \
   encr_alg aes \
   auth_alg sha1  \
   encrkey 0a1f3886b06ebd7d39f6f89e4c29c93f2741c6fa598a38af969907a29ab1b42a \
   authkey a7230aabf513f35785da73e33b064608be41f69a
#
# add esp spi 0x177xce34\
   src 192.168.13.213 dst 192.168.116.16 \
   encr_alg aes \
   auth_alg sha1  \
   encrkey 4ef5be40bf93498017b2151d788bb37e372f091add9b11149fba42435fefe328 \
   authkey 0e1875d9ff8e42ab652766a5cad49f38c9152821
```

- 最後に、管理者は manual-key サービスを再起動します。restart コマンドにより、新しい鍵を追加する前に古い鍵が消去されます

```
# svcadm restart manual-key
```

▼ IPsec によってパケットが保護されていることを確認する方法

パケットが保護されていることを確認するには、snoop コマンドで接続をテストします。snoop 出力に表示される接頭辞は、次のとおりです。

- AH: 接頭辞は、AH がヘッダーを保護していることを示します。AH: が表示されるのは、auth_alg を使ってトラフィックを保護している場合です。
- ESP: 接頭辞は、暗号化されたデータが送信されていることを示します。ESP: が表示されるのは、encr_auth_alg か encr_alg を使ってトラフィックを保護している場合です。

始める前に　snoop 出力を作成するためには、スーパーユーザーであるか、それと同等の役割でなければなりません。さらに、接続をテストするためには、両方のシステムにアクセスできなければなりません。

1　一方のシステム (たとえば、**partym**) でスーパーユーザー になります。

```
% su -
Password:       Type root password
#
```

2 **partym** システムから、リモートシステムからパケットをスヌープする準備をします。

partym の端末ウィンドウで、enigma システムからパケットをスヌープします。

```
# snoop -d hme0 -v enigma
Using device /dev/hme (promiscuous mode)
```

3 リモートシステムからパケットを送信します。

別の端末ウィンドウで、enigma システムにリモートからログインします。パスワードを入力します。次に、スーパーユーザーになり、enigma システムからのパケットを partym システムに送信します。パケットは、snoop -v enigma コマンドで取り込む必要があります。

```
% ssh enigma
Password:      Type your password
% su -
Password:      Type root password
# ping partym
```

4 **snoop** の出力を調べます。

partym システムで、冒頭の IP ヘッダー情報のあとに AH と ESP 情報が含まれている出力を確認します。次のような AH と ESP の情報は、パケットが保護されていることを示します。

```
IP:   Time to live = 64 seconds/hops
IP:   Protocol = 51 (AH)
IP:   Header checksum = 4e0e
IP:   Source address = 192.168.116.16, enigma
IP:   Destination address = 192.168.13.213, partym
IP:   No options
IP:
AH:  ----- Authentication Header -----
AH:
AH:  Next header = 50 (ESP)
AH:  AH length = 4 (24 bytes)
AH:  <Reserved field = 0x0>
AH:  SPI = 0xb3a8d714
AH:  Replay = 52
AH:  ICV = c653901433ef5a7d77c76eaa
AH:
ESP:  ----- Encapsulating Security Payload -----
ESP:
ESP:  SPI = 0xd4f40a61
ESP:  Replay = 52
ESP:     ....ENCRYPTED DATA....

ETHER:  ----- Ether Header -----
...
```

▼ ネットワークセキュリティーの役割を構成する方法

役割によるアクセス制御 (RBAC) でシステムを管理している場合は、ネットワーク管理またはネットワークセキュリティー上の役割を提供するためにこの手順を使用します。

1 ローカルの **prof_attr** データベースで **Network** 権利プロファイルを検索します。

現在のリリースでは、次のような出力が表示されます。

```
% cd /etc/security
% grep Network prof_attr
Network IPsec Management:::Manage IPsec and IKE...
Network Link Security:::Manage network link security...
Network Management:::Manage the host and network configuration...
Network Security:::Manage network and host security...
Network Wifi Management:::Manage wifi network configuration...
Network Wifi Security:::Manage wifi network security...
```

Solaris 10 4/09 リリースより前のリリースを実行している場合は、次のような出力が表示されます。

```
% cd /etc/security
% grep Network prof_attr
Network Management:::Manage the host and network configuration
Network Security:::Manage network and host security
System Administrator::: Network Management
```

Network Management プロファイルは、System Administrator プロファイルを補完するプロファイルです。System Administrator 権利プロファイルを役割に含めると、その役割は Network Management プロファイルでコマンドを実行できます。

2 **Network Management** 権利プロファイルに含まれているコマンドを調べます。

```
% grep "Network Management" /etc/security/exec_attr
Network Management:solaris:cmd:::/usr/sbin/ifconfig:privs=sys_net_config
...
Network Management:suser:cmd:::/usr/sbin/snoop:uid=0
```

`solaris` ポリシーコマンドは、特権 (`privs=sys_net_config`) で実行されます。`suser` ポリシーコマンドは、スーパーユーザー (`uid=0`) として実行されます。

3 サイトでのネットワークセキュリティーの役割の範囲を決定します。

決定には、手順 1 の権利プロファイルの定義を参考にしてください。

- すべてのネットワークセキュリティーを扱う役割を作成する場合は、**Network Security** 権利プロファイルを使用します。
- 現在のリリースで、**IPsec** と **IKE** だけを扱う役割を作成する場合は、**Network IPsec Management** 権利プロファイルを使用します。

4 **Network Management 権利プロファイルを含むネットワークセキュリティーの役割を作成します。**

Network Management プロファイルに加え、Network Security または Network IPsec Management 権利プロファイルを持つ役割は、`ifconfig`、`snoop`、`ipsecconf`、および `ipseckey` コマンドなどを適切な特権で実行できます。

役割の作成、役割のユーザーへの割り当て、ネームサービスによる変更点の登録については、『Solaris のシステム管理: セキュリティーサービス』の「RBAC の構成 (タスクマップ)」を参照してください。

例 20–5 ネットワークセキュリティーの責任を役割に振り分ける

この例では、管理者がネットワークセキュリティーの責任を 2 つの役割に振り分けます。一方の役割は Wifi とリンクのセキュリティーを管理し、もう一方の役割は IPsec と IKE を管理します。各役割には、シフトごとに 1 人、合計 3 人を割り当てます。

管理者によって次のように役割が作成されます。

- 最初の役割には LinkWifi という名前を付けます。
  - この役割には Network Wifi、Network Link Security、および Network Management 権利プロファイルを割り当てます。
  - その後、該当するユーザーにこの LinkWifi 役割を割り当てます。
- 2 番目の役割には IPsec Administrator という名前を付けます。
  - この役割には Network IPsec Management および Network Management 権利プロファイルを割り当てます。
  - その後、該当するユーザーにこの IPsec Administrator 役割を割り当てます。

▼ IKE および IPsec サービスを管理する方法

次の手順では、IPsec の管理、IKE の管理、および手動での鍵管理に SMF サービスを使用する代表的な方法について説明します。デフォルトでは、`policy` サービスと `ipsecalgs` サービスは有効になっています。また、デフォルトでは、`ike` サービスと `manual-key` サービスは無効になっています。

1 **IPsec ポリシーを管理するには、次のいずれかを実行します。**

- **`ipsecinit.conf` ファイルに新しいポリシーを追加したあと、`policy` サービスをリフレッシュします。**

```
# svcadm refresh svc:/network/ipsec/policy
```

- サービスのプロパティーの値を変更したあと、プロパティーの値を表示し、**policy** サービスをリフレッシュしてから再起動します。

```
# svccfg -s policy setprop config/config_file=/etc/inet/MyIpsecinit.conf
# svcprop -p config/config_file policy
/etc/inet/MyIpsecinit.conf
# svcadm refresh svc:/network/ipsec/policy
# svcadm restart svc:/network/ipsec/policy
```

2 鍵を自動的に管理するには、次のいずれかを実行します。

- **/etc/inet/ike/config** ファイルにエントリを追加したあと、**ike** サービスを有効にします。

```
# svcadm enable svc:/network/ipsec/ike
```

- **/etc/inet/ike/config** ファイルのエントリを変更したあと、**ike** サービスを再起動します。

```
# svcadm restart svc:/network/ipsec/ike
```

- サービスのプロパティーの値を変更したあと、プロパティーの値を表示し、サービスをリフレッシュしてから再起動します。

```
# svccfg -s ike setprop config/admin_privilege=modkeys
# svcprop -p config/admin_privilege ike
modkeys
# svcadm refresh svc:/network/ipsec/ike
# svcadm restart svc:/network/ipsec/ike
```

- **ike** サービスを停止するには、無効にします。

```
# svcadm disable svc:/network/ipsec/ike
```

3 鍵を手動で管理するには、次のいずれかを実行します。

- **/etc/inet/secret/ipseckeys** ファイルにエントリを追加したあと、**manual-key** サービスを有効にします。

```
# svcadm enable svc:/network/ipsec/manual-key
```

- **ipseckeys** ファイルを変更したあと、サービスをリフレッシュします。

```
# svcadm refresh manual-key
```

- サービスのプロパティーの値を変更したあと、プロパティーの値を表示し、サービスをリフレッシュしてから再起動します。

```
# svccfg -s manual-key setprop config/config_file=/etc/inet/secret/MyIpseckeyfile
# svcprop -p config/config_file manual-key
/etc/inet/secret/MyIpseckeyfile
# svcadm refresh svc:/network/ipsec/manual-key
# svcadm restart svc:/network/ipsec/manual-key
```

- 鍵を手動で管理できないようにするには、**manual-key** サービスを無効にします。

  ```
  # svcadm disable svc:/network/ipsec/manual-key
  ```

4. **IPsec** のプロトコルとアルゴリズムのテーブルを変更した場合は、**ipsecalgs** サービスをリフレッシュします。

   ```
   # svcadm refresh svc:/network/ipsec/ipsecalgs
   ```

注意事項 サービスのステータスを調べるには、`svcs` *service* コマンドを使用します。サービスが `maintenance` (保守) モードになっている場合は、`svcs -x` *service* コマンドの出力に表示されるデバッグのヒントに従ってください。

IPsec による VPN の保護

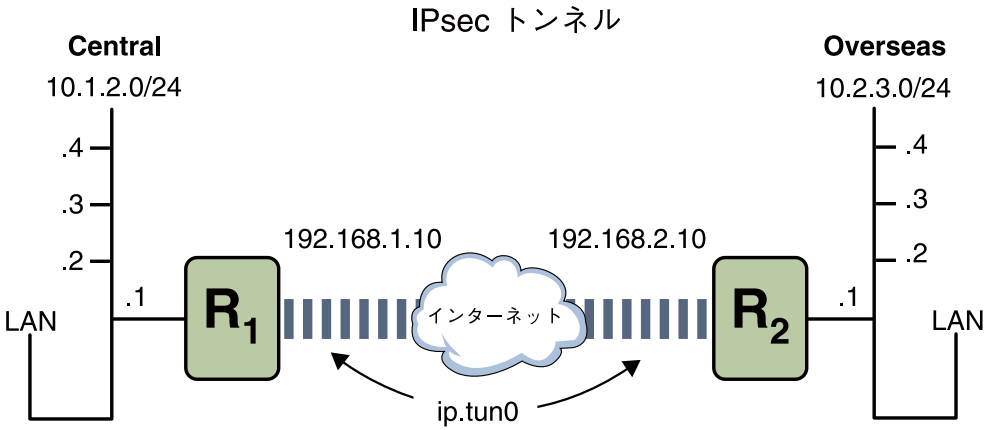
Oracle Solaris では、IPsec で保護された VPN を構成できます。トンネルは、トンネルモードまたはトランスポートモードで作成できます。「トンネルモード」は、ほかのベンダーによって実装された IPsec と相互運用できます。「トランスポートモード」は、以前のバージョンの Solaris OS と相互運用できます。トンネルのモードについては、518 ページの「IPsec のトランスポートモードとトンネルモード」を参照してください。

トンネルモードの IPsec を使用すると、トラフィックをよりきめ細かく制御できます。トンネルモードでは、内側の IP アドレスに対して、必要に応じて特定の保護をポート単位まで細かく指定できます。

- トンネルモードのトンネル用の IPsec ポリシーの例については、548 ページの「トンネルモードを使用して VPN を IPsec で保護する例」を参照してください。
- VPN を保護する手順については、550 ページの「IPsec による VPN の保護 (タスクマップ)」を参照してください。

トンネルモードを使用して VPN を IPsec で保護する例

図 20–1　IPsec トンネル図



次の例では、LAN のすべてのサブネットに対してトンネルを構成することを前提にしています。

```
## Tunnel configuration ##
# Tunnel name is ip.tun0
# Intranet point for the source is 10.1.2.1
# Intranet point for the destination is 10.2.3.1
# Tunnel source is 192.168.1.10
# Tunnel destination is 192.168.2.10
```

例 20–6　すべてのサブネットで使用できるトンネルの作成

この例では、図 20–1 の Central LAN のローカル LAN から送信されるすべてのトラフィックが、ルーター 1 からルーター 2 にトンネリングされ、Overseas LAN のすべてのローカル LAN に配信されます。トラフィックは AES で暗号化されます。

```
## IPsec policy ##
{tunnel ip.tun0 negotiate tunnel}
 ipsec {encr_algs aes encr_auth_algs sha1 sa shared}
```

例 20–7　2 つのサブネットだけを接続するトンネルの作成

この例では、Central LAN のサブネット `10.1.2.0/24` と Overseas LAN のサブネット `10.2.3.0/24` の間のトラフィックだけがトンネリングされ、暗号化されます。Central に対するほかの IPsec ポリシーがない場合、Central LAN がこのトンネル経由でほかの LAN にトラフィックを配信しようとすると、トラフィックはルーター 1 でドロップされます。

```
## IPsec policy ##
{tunnel ip.tun0 negotiate tunnel laddr 10.1.2.0/24 raddr 10.2.3.0/24}
 ipsec {encr_algs aes encr_auth_algs sha1 shared}
```

例 20–8　2 つのサブネット間だけの sendmail トラフィック用トンネルの作成

この例では、sendmail トラフィック専用のトンネルが作成されます。トラフィックは、Central LAN のサブネット 10.1.2.0/24 から、Overseas LAN の 10.2.3.0/24 サブネット上にある電子メールサーバーに配信されます。電子メールは Blowfish で暗号化されます。ポリシーは、リモートおよびローカルの電子メールポートに適用されます。rport ポリシーは、Central からリモートの Overseas の電子メールポートに送信される電子メールを保護します。lport ポリシーは、Central のローカルポート 25 で受信される Overseas からの電子メールを保護します。

```
## IPsec policy for email from Central to Overseas ##
{tunnel ip.tun0 negotiate tunnel ulp tcp rport 25
 laddr 10.1.2.0/24 raddr 10.2.3.0/24}
 ipsec {encr_algs blowfish encr_auth_algs sha1 sa shared}

## IPsec policy for email from Overseas to Central ##
{tunnel ip.tun0 negotiate tunnel ulp tcp lport 25
 laddr 10.1.2.0/24 raddr 10.2.3.0/24}
 ipsec {encr_algs blowfish encr_auth_algs sha1 sa shared}
```

例 20–9　すべてのサブネットに対する FTP トラフィック用トンネルの作成

この例の IPsec ポリシーは、図 20–1 の Central LAN のすべてのサブネットから Overseas LAN のすべてのサブネットまで、FTP ポートを AES で保護します。この構成は、FTP アクティブモードで機能します。

```
## IPsec policy for outbound FTP from Central to Overseas ##
{tunnel ip.tun0 negotiate tunnel ulp tcp rport 21}
  ipsec {encr_algs aes encr_auth_algs sha1 sa shared}
{tunnel ip.tun0 negotiate tunnel ulp tcp lport 20}
  ipsec {encr_algs aes encr_auth_algs sha1 sa shared}

## IPsec policy for inbound FTP from Central to Overseas ##
{tunnel ip.tun0 negotiate tunnel ulp tcp lport 21}
  ipsec {encr_algs aes encr_auth_algs sha1 sa shared}
{tunnel ip.tun0 negotiate tunnel ulp tcp rport 20}
  ipsec {encr_algs aes encr_auth_algs sha1 sa shared}
```

IPsec による VPN の保護 (タスクマップ)

次のタスクマップに、インターネット経由のトラフィックを保護するように IPsec を構成する手順を示します。これらの手順を使用すると、インターネット経由の 2 つのシステムの間に、セキュリティー保護された仮想プライベートネットワーク (VPN) を設定できます。このテクノロジの一般的な使い方の 1 つに、インターネットを経由してリモートオフィスを企業ネットワークにセキュアに接続することがあります。

タスク	説明	参照先
IPv4 トンネルモードでトンネルトラフィックを保護します。	2 つの Solaris 10 システム間、または Solaris 10 システムと Oracle Solaris 11 システムの間で、トンネルモードでトラフィックを保護します。Solaris 10 システムで Solaris 10 7/07 リリース以上が実行されている必要があります。 また、Solaris 10 システムまたは Oracle Solaris システムと、別のプラットフォームで稼働中のシステムの間でも、トンネルモードでトラフィックを保護します。Solaris 10 システムで Solaris 10 7/07 リリース以上が実行されている必要があります。	553 ページの「IPv4 を使用するトンネルモードの IPsec トンネルで VPN を保護する方法」
IPv6 トンネルモードでトンネルトラフィックを保護します。	IPv6 プロトコルを使用している 2 つの Oracle Solaris システム間で、トンネルモードでトラフィックを保護します。	564 ページの「IPv6 を使用するトンネルモードの IPsec トンネルで VPN を保護する方法」
IPv4 トランスポートモードでトンネルトラフィックを保護します。	2 つの Solaris 10 システム間、または Solaris 10 システムと Oracle Solaris システムの間で、トランスポートモードでトラフィックを保護します。Solaris 10 システムで Solaris 10 7/07 リリース以上が実行されている必要があります。 また、以前のバージョンの Solaris OS を実行しているシステムと、Solaris 10 または Oracle Solaris システムの間でも、トランスポートモードでトラフィックを保護します。Solaris 10 システムで Solaris 10 7/07 リリース以上が実行されている必要があります。	570 ページの「IPv4 を使用するトランスポートモードの IPsec トンネルで VPN を保護する方法」
	推奨されなくなった古い構文を使用してトラフィックを保護します。この方法は、以前のバージョンの Solaris OS を実行しているシステムと通信する場合に役立ちます。この方法を使用すると、2 つのシステム上にある構成ファイルの比較が簡単になります。	例 20–11 例 20–16
IPv6 トランスポートモードでトンネルトラフィックを保護します。	IPv6 プロトコルを使用している 2 つの Oracle Solaris システム間で、トランスポートモードでトラフィックを保護します。	576 ページの「IPv6 を使用するトランスポートモードの IPsec トンネルで VPN を保護する方法」
IP のスプーフィングを防止します。	SMF サービスを作成して、システムがパケットを復号化せずに VPN 上で転送することを防止します。	583 ページの「IP のスプーフィングを防止する方法」

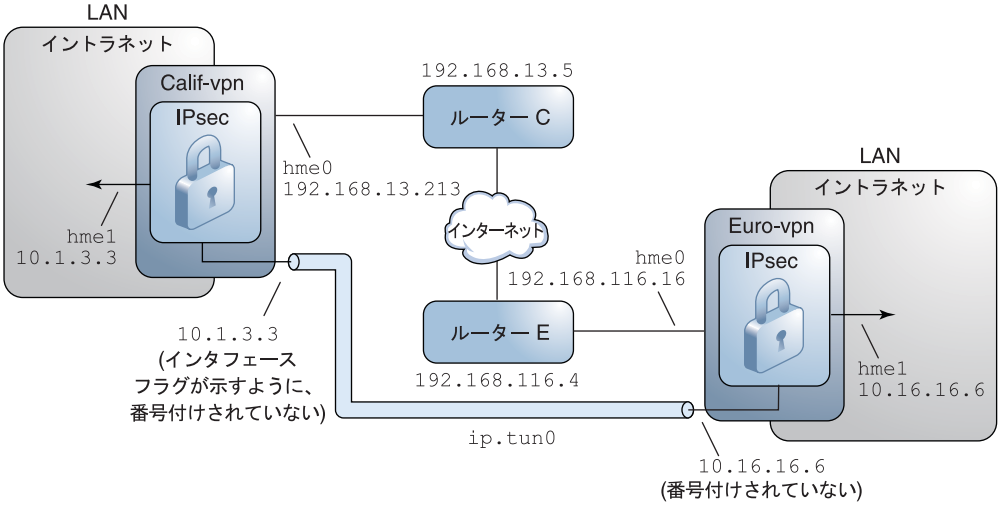
IPsec で VPN を保護するタスクのためのネットワークトポロジの説明

このセクション以降に説明する手順では、次の設定がすでになされているものとします。図 20–2 はこのネットワークを表しています。

- 各システムは IPv4 アドレス空間を使用します。

  IPv6 アドレスを使用する同様の例については、564 ページの「IPv6 を使用するトンネルモードの IPsec トンネルで VPN を保護する方法」を参照してください。
- 各システムには 2 つのインタフェースがあります。`hme0` インタフェースはインターネットに接続しています。この例では、インターネット IP アドレスは `192.168` で始まります。`hme1` インタフェースは社内の LAN、すなわちイントラネットに接続します。この例では、イントラネット IP アドレスは `10` で始まります。
- 各システムには、SHA–1 アルゴリズムを使用した ESP 認証が必要です。SHA–1 アルゴリズムには 160 ビットのキーが必要です。
- 各システムには、AES アルゴリズムを使用した ESP 暗号化が必要です。AES アルゴリズムは 128 ビットのキーまたは 256 ビットのキーを使用します。
- 各システムは、インターネットに直接アクセスするルーターに接続できます。
- 各システムは、共有セキュリティーアソシエーションを使用します。

図 20–2 インターネットで隔てられているオフィス間の VPN の例



前の図に示すように、IPv4 ネットワーク向けの手順では次の構成パラメータを使用します。

パラメータ	ヨーロッパ	カリフォルニア
システム名	enigma	partym
システムイントラネットインタフェース	hme1	hme1
システムイントラネットアドレス。手順 7 の *-point* アドレスでもある	10.16.16.6	10.1.3.3
システムインターネットインタフェース	hme0	hme0
システムイントラネットアドレス。手順 7 の *tsrc* アドレスでもある	192.168.116.16	192.168.13.213
インターネットルーターの名前	router-E	router-C
インターネットルーターのアドレス	192.168.116.4	192.168.13.5
トンネル名	ip.tun0	ip.tun0

手順では次の IPv6 アドレスを使用します。トンネル名は同じです。

パラメータ	ヨーロッパ	カリフォルニア
システムイントラネットアドレス	6000:6666::aaaa:1116	6000:3333::eeee:1113
システムインターネットアドレス	2001::aaaa:6666:6666	2001::eeee:3333:3333
インターネットルーターのアドレス	2001::aaaa:0:4	2001::eeee:0:1

▼ IPv4 を使用するトンネルモードの IPsec トンネルで VPN を保護する方法

トンネルモードでは、内側の IP パケットによって、その内容を保護する IPsec ポリシーが決まります。

この手順は、529 ページの「IPsec で 2 つのシステム間のトラフィックを保護するには」の手順の応用です。設定については、552 ページの「IPsec で VPN を保護するタスクのためのネットワークトポロジの説明」を参照してください。

注 – 両方のシステムでこの手順を実行してください。

この手順では、2 つのシステムを接続するだけでなく、これら 2 つのシステムに接続している 2 つのイントラネットを接続します。この手順における 2 つのシステムはゲートウェイとして機能します。

始める前に　システムまたは共有 IP ゾーンの IPsec ポリシーの構成は、大域ゾーンで行う必要があります。排他的 IP ゾーンについては、非大域ゾーンで IPsec ポリシーを構成します。

1 システムコンソール上で、**Primary Administrator** の役割を引き受けるか、スーパーユーザーになります。

Primary Administrator 役割には、Primary Administrator プロファイルが含まれます。役割を作成してユーザーに役割を割り当てるには、『Oracle Solaris の管理: 基本管理』の第 2 章「Solaris 管理コンソールの操作 (タスク)」を参照してください。

注 – リモートログインすると、セキュリティー上重要なトラフィックが盗聴される恐れがあります。何らかの方法でリモートログインを保護していても、システムのセキュリティーがリモートログインセッションレベルに低下します。セキュリティー保護されたリモートログインには、`ssh` コマンドを使用してください。

2 **IPsec** を構成する前に、パケットフローを制御します。

a. **IP** 転送と **IP** 動的経路制御が無効になっていることを確認します。

```
 # routeadm
Configuration   Current             Current
       Option   Configuration       System State
---------------------------------------------------------------
IPv4 routing      default (enabled)    enabled
IPv4 forwarding   disabled             disabled
...
```

IP 転送と IP 動的経路制御が有効の場合は、無効にします。

```
# routeadm -d ipv4-routing -d ipv4-forwarding
# routeadm -u
```

IP 転送をオフにすると、このシステムを介したあるネットワークから別のネットワークへのパケット転送ができなくなります。`routeadm` コマンドの説明については、`routeadm(1M)` のマニュアルページを参照してください。

b. 次のコマンドを入力して **IP** の厳密宛先マルチホームをオンに設定します。

```
# ndd -set /dev/ip ip_strict_dst_multihoming 1
```

IP 厳密マルチホームをオンに設定するには、システムの着信先アドレスのうちの 1 つに宛てたパケットが正しい着信先アドレスに到着する必要があります。

厳密宛先マルチホームが有効な状態では、特定のインタフェースに到着したパケットに、そのインタフェースのいずれかのローカル IP アドレスを指定する必要があります。その他のパケットは、システムのほかのローカルアドレスが指定されているものも含めてすべて捨てられます。

注意 – システムのブート時に、マルチホームの値はデフォルトに戻ります。変更した値を持続させる方法については、583 ページの「IP のスプーフィングを防止する方法」を参照してください。

c. 大部分のネットワークサービス、可能な場合はすべてのネットワークサービスを無効にします。

注 – 「制限付き」SMF プロファイルでシステムをインストールした場合、この手順は省略できます。Oracle Solaris の Secure Shell 機能以外のネットワークサービスは無効になります。

ネットワークサービスを無効にすると、IP パケットがシステムにダメージを与えるのを防止できます。たとえば、SNMP デーモン、`telnet` 接続、`rlogin` 接続などを最大限に活用できます。

次のオプションのいずれかを選択します。

- Solaris 10 11/06 リリースまたはそれ以降のリリースが稼働している場合は、「制限付き」SMF プロファイルを実行します。

  ```
  # netservices limited
  ```

- それ以外の場合は、ネットワークサービスを個別に無効にします。

  ```
  # svcadm disable network/ftp:default
  # svcadm disable network/finger:default
  # svcadm disable network/login:rlogin
  # svcadm disable network/nfs/server:default
  # svcadm disable network/rpc/rstat:default
  # svcadm disable network/smtp:sendmail
  # svcadm disable network/telnet:default
  ```

d. ほとんどのネットワークサービスが無効になっていることを確認します。

ループバックマウントと `ssh` サービスが稼働していることを確認します。

```
# svcs | grep network
online         Aug_02   svc:/network/loopback:default
...
online         Aug_09   svc:/network/ssh:default
```

3 2 つのシステム間に SA ペアを追加します。

次のオプションのいずれかを選択します。

- SA 用のキーを管理するように IKE を構成します。603 ページの「IKE の構成 (タスクマップ)」のいずれかの手順に従って、VPN 用の IKE を構成します。
- キーを手動で管理する決定的な理由がある場合は、538 ページの「IPsec セキュリティーアソシエーションを手動で作成する方法」を参照してください。

4 **IPsec ポリシーを追加します。**

`/etc/inet/ipsecinit.conf` ファイルを編集して、VPN 用の IPsec ポリシーを追加します。ポリシーを強化する方法については、例 20–12 を参照してください。その他の例については、548 ページの「トンネルモードを使用して VPN を IPsec で保護する例」を参照してください。

このポリシーでは、ローカル LAN 上のシステムとゲートウェイの内部 IP アドレスの間に IPsec 保護は必要でないため、`bypass` 文を追加します。

a. `enigma` システムで、`ipsecinit.conf` ファイルに次のエントリを入力します。

```
# LAN traffic to and from this host can bypass IPsec.
{laddr 10.16.16.6 dir both} bypass {}

# WAN traffic uses ESP with AES and SHA-1.
{tunnel ip.tun0 negotiate tunnel}
 ipsec {encr_algs aes encr_auth_algs sha1 sa shared}
```

b. `partym` システムで、`ipsecinit.conf` ファイルに次のエントリを追加します。

```
# LAN traffic to and from this host can bypass IPsec.
{laddr 10.1.3.3 dir both} bypass {}

# WAN traffic uses ESP with AES and SHA-1.
{tunnel ip.tun0 negotiate tunnel}
 ipsec {encr_algs aes encr_auth_algs sha1 sa shared}
```

5 **(省略可能) IPsec ポリシーファイルの構文を確認します。**

```
# ipsecconf -c -f /etc/inet/ipsecinit.conf
```

6 **トンネルを構成し、それを IPsec で保護する場合は、Oracle Solaris のリリースに応じて次の手順に従います。**

- **Solaris 10 4/09** リリース以降では、手順 7 から手順 13 までを実行したあと、手順 22 の経路制御プロトコルを実行します。
- **Solaris 10 4/09** リリースより前のリリースを実行している場合は、手順 14 から手順 22 までを実行します。

7 トンネル **ip.tun0** を **/etc/hostname.ip.tun0** ファイルで構成します。

ファイルの構文は次のとおりです。

system1-point system2-point `tsrc` *system1-taddr* `tdst` *system2-taddr* `router up`

a. **enigma** システムで、**hostname.ip.tun0** ファイルに次のエントリを追加します。

```
10.16.16.6 10.1.3.3 tsrc 192.168.116.16 tdst 192.168.13.213 router up
```

b. **partym** システムで、**hostname.ip.tun0** ファイルに次のように追加します。

```
10.1.3.3 10.16.16.6 tsrc 192.168.13.213 tdst 192.168.116.16 router up
```

8 作成した **IPsec** ポリシーでトンネルを保護します。

```
# svcadm refresh svc:/network/ipsec/policy:default
```

9 トンネル構成ファイルの内容をカーネルに読み込むには、ネットワークサービスを再起動します。

```
# svcadm restart svc:/network/initial:default
```

10 **hme1** インタフェースの **IP** 転送をオンに設定します。

a. **enigma** システムで、**/etc/hostname.hme1** ファイルにルーターエントリを追加します。

```
192.168.116.16 router
```

b. **partym** システムで、**/etc/hostname.hme1** ファイルにルーターエントリを追加します。

```
192.168.13.213 router
```

IP 転送とは、別のインタフェースから到着したパケットを転送できることを意味します。IP 転送はまた、送信するパケットがもともとは別のインタフェースから発信されたパケットである可能性も意味します。パケットを正しく転送するには、受信インタフェースと送信インタフェースの IP 転送をオンに設定しておきます。

`hme1` インタフェースはイントラネットの「内部」にあるため、`hme1` の IP 転送はオンに設定しておきます。さらに、`ip.tun0` はインターネットを通してこれら 2 つのシステムを接続するため、`ip.tun0` の IP 転送はオンに設定しておきます。

`hme0` インタフェースの IP 転送はオフです。そのため、「外部」からパケットが保護イントラネットに侵入するのを防ぐことができます。「外部」とはインターネットを意味します。

11 経路制御プロトコルによってイントラネット内のデフォルトのルートが通知されていないことを確認します。

a. **enigma** システムで、**private** フラグを **/etc/hostname.hme0** ファイルに追加します。

```
10.16.16.6 private
```

b. **partym** システムで、**private** フラグを **/etc/hostname.hme0** ファイルに追加します。

```
10.1.3.3 private
```

hme0 の IP 転送がオフになっていても、ルーティングプロトコルの実装によっては、このインタフェースを通知することがあります。たとえば、in.routed プロトコルは、イントラネット内のピアにパケットが転送される際に hme0 を有効なインタフェースとして通知する場合があります。インタフェースの「*private*」フラグを設定して、このような通知が行われないようにします。

12 **hme0** インタフェース経由のデフォルトルートを手動で追加します。

デフォルトルートは、インターネットに直接アクセスできるルーターでなければなりません。

a. **enigma** システムで、次のルートを追加します。

```
# route add default 192.168.116.4
```

b. **partym** システムで次のルートを追加します。

```
# route add default 192.168.13.5
```

hme0 インタフェースはイントラネットの一部ではありませんが、インターネットを介してそのピアシステムにアクセスする必要があります。hme0 は、自身のピアを見つけるために、インターネット経路制御情報を必要とします。インターネットの残りの要素にとって、VPN システムは、ルーターというよりもホストのような存在です。したがって、デフォルトルーターを使用するか、ルーター発見プロトコルを実行すれば、ピアシステムを見つけることができます。詳細については、route(1M) と in.routed(1M) のマニュアルページを参照してください。

13 手順を完了するために、手順 **22** に進んで経路制御プロトコルを実行します。

14 トンネル **ip.tun0** を構成します。

注 – 次の手順は、Solaris 10 4/09 リリースより前のリリースを実行しているシステムでトンネルを構成するためのものです。

---

ifconfig コマンドを使用してポイントツーポイントインタフェースを作成します。

```
# ifconfig ip.tun0 plumb

# ifconfig ip.tun0 system1-point system2-point \
tsrc system1-taddr tdst system2-taddr
```

a. **enigma** システムで、次のコマンドを入力します。

```
# ifconfig ip.tun0 plumb

# ifconfig ip.tun0 10.16.16.6 10.1.3.3 \
tsrc 192.168.116.16 tdst 192.168.13.213
```

b. **partym** システムで、次のコマンドを入力します。

```
# ifconfig ip.tun0 plumb

# ifconfig ip.tun0 10.1.3.3 10.16.16.6  \
tsrc 192.168.13.213 tdst 192.168.116.16
```

15 作成した **IPsec** ポリシーでトンネルを保護します。

```
# ipsecconf
```

16 トンネル用のルーターを起動します。

```
# ifconfig ip.tun0 router up
```

17 **hme1** インタフェースの **IP** 転送をオンに設定します。

```
# ifconfig hme1 router
```

IP 転送とは、別のインタフェースから到着したパケットを転送できることを意味します。IP 転送はまた、送信するパケットがもともとは別のインタフェースから発信されたパケットである可能性も意味します。パケットを正しく転送するには、受信インタフェースと送信インタフェースの IP 転送をオンに設定しておきます。

hme1 インタフェースはイントラネットの「内部」にあるため、hme1 の IP 転送はオンに設定しておきます。さらに、ip.tun0 はインターネットを通してこれら 2 つのシステムを接続するため、ip.tun0 の IP 転送はオンに設定しておきます。

hme0 インタフェースの IP 転送はオフです。そのため、「外部」からパケットが保護イントラネットに侵入するのを防ぐことができます。「外部」とはインターネットを意味します。

18 経路制御プロトコルによってイントラネット内のデフォルトのルートが通知されていないことを確認します。

```
# ifconfig hme0 private
```

hme0 の IP 転送がオフになっていても、経路制御プロトコルの実装によっては、このインタフェースを通知することがあります。たとえば、in.routed プロトコルは、イ

ントラネット内のピアにパケットが転送される際に hme0 を有効なインタフェースとして通知する場合があります。インタフェースの「*private*」フラグを設定して、このような通知が行われないようにします。

19 **hme0** 経由のデフォルトルートを手動で追加します。

デフォルトルートは、インターネットに直接アクセスできるルーターでなければなりません。

a. **enigma** システムで、次のルートを追加します。

```
# route add default 192.168.116.4
```

b. **partym** システムで次のルートを追加します。

```
# route add default 192.168.13.5
```

hme0 インタフェースはイントラネットの一部ではありませんが、インターネットを介してそのピアシステムにアクセスする必要があります。hme0 は、自身のピアを見つけるために、インターネット経路制御情報を必要とします。インターネットの残りの要素にとって、VPN システムは、ルーターというよりもホストのような存在です。したがって、デフォルトルーターを使用するか、ルーター発見プロトコルを実行すれば、ピアシステムを見つけることができます。詳細については、route(1M) と in.routed(1M) のマニュアルページを参照してください。

20 VPN がリブート後に開始するように、**/etc/hostname.ip.tun0** ファイルにエントリを追加します。

system1-point system2-point tsrc *system1-taddr* tdst *system2-taddr* router up

a. **enigma** システムで、**hostname.ip.tun0** ファイルに次のエントリを追加します。

```
10.16.16.6 10.1.3.3 tsrc 192.168.116.16 tdst 192.168.13.213 router up
```

b. **partym** システムで、**hostname.ip.tun0** ファイルに次のように追加します。

```
10.1.3.3 10.16.16.6 tsrc 192.168.13.213 tdst 192.168.116.16 router up
```

21 適切なパラメータが経路制御デーモンに渡されるようにインタフェースファイルを構成します。

a. **enigma** システムで、**/etc/hostname.***interface* ファイルを変更します。

```
# cat /etc/hostname.hme0
## enigma
10.16.16.6 private

# cat /etc/hostname.hme1
## enigma
192.168.116.16 router
```

b. **partym** システムで、**/etc/hostname.***interface* ファイルを変更します。

```
# cat /etc/hostname.hme0
## partym
10.1.3.3 private

# cat /etc/hostname.hme1
## partym
192.168.13.213 router
```

22 経路制御プロトコルを実行します。

```
# routeadm -e ipv4-routing
# routeadm -u
```

経路制御プロトコルを実行する前に経路制御プロトコルの構成が必要な場合があります。詳細は、261 ページの「Oracle Solaris のルーティングプロトコル」を参照してください。手順については、124 ページの「IPv4 ルーターの構成方法」を参照してください。

例 20–10 テスト時に一時的なトンネルを作成する

この例では、管理者が Solaris 10 4/09 システムでトンネルの作成をテストします。あとで 553 ページの「IPv4 を使用するトンネルモードの IPsec トンネルで VPN を保護する方法」の手順を使用して、これらを永続的なトンネルにします。テスト中に、管理者はシステム system1 および system2 で次の一連の手順を実行します。

- 両方のシステムで、管理者は 553 ページの「IPv4 を使用するトンネルモードの IPsec トンネルで VPN を保護する方法」の最初から 5 番目までの手順を完了します。
- 管理者は ifconfig コマンドを使用して、一時的なトンネルを plumb して構成します。

```
system1 # ifconfig ip.tun0 plumb
system1 # ifconfig ip.tun0 10.16.16.6 10.1.3.3 \
          tsrc 192.168.116.16 tdst 192.168.13.213

# ssh system2
Password:      admin-password-on-system2
system2 # ifconfig ip.tun0 plumb
system2 # ifconfig ip.tun0  10.1.3.3 10.16.16.6 \
          tsrc 192.168.13.213 tdst 192.168.116.16
```

- 管理者はトンネルに対して IPsec ポリシーを有効にします。このポリシーは、553 ページの「IPv4 を使用するトンネルモードの IPsec トンネルで VPN を保護する方法」の手順 4 で作成したものです。

```
system1 # svcadm refresh svc:/network/ipsec/policy:default
system2 # svcadm refresh svc:/network/ipsec/policy:default
```

- 管理者はインターネットインタフェースをルーターにし、経路制御プロトコルがイントラネットインタフェースを越えないようにします。

```
system1 # ifconfig hme1 router ; ifconfig hme0 private

system2 # ifconfig hme1 router ; ifconfig hme0 private
```

- 両方のシステムで 553 ページの「IPv4 を使用するトンネルモードの IPsec トンネルで VPN を保護する方法」の手順 12 と手順 22 を実行して、ルートを手動で追加し、ルーティングプロトコルを実行します。

例 20–11 コマンド行を使用して以前のバージョンの Solaris システムにトンネルを作成する

Solaris 10 7/07 リリースでは、`ifconfig` コマンドの構文が簡素化されました。この例では、Solaris 10 7/07 リリースより前のバージョンの Solaris を実行しているシステムに対するトンネルの作成を管理者がテストします。`ifconfig` コマンドの元の構文を使用すれば、通信する 2 つのシステムで同じコマンドを使用できます。あとで 553 ページの「IPv4 を使用するトンネルモードの IPsec トンネルで VPN を保護する方法」の手順を使用して、これらを永続的なトンネルにします。

テスト中に、管理者はシステム `system1` および `system2` で次の手順を実行します。

- 両方のシステムで、管理者は 553 ページの「IPv4 を使用するトンネルモードの IPsec トンネルで VPN を保護する方法」の最初から 5 番目までの手順を完了します。
- 管理者はトンネルを plumb して構成します。

```
system1 # ifconfig ip.tun0 plumb
system1 # ifconfig ip.tun0 10.16.16.6 10.1.3.3 \
          tsrc 192.168.116.16 tdst 192.168.13.213 \
          encr_algs aes encr_auth_algs sha1
system1 # ifconfig ip.tun0 router up

# ssh system2
Password:      admin-password-on-system2
system2 # ifconfig ip.tun0 plumb
system2 # ifconfig ip.tun0 10.1.3.3  10.16.16.6 \
          tsrc 192.168.13.213 tdst 192.168.116.16 \
          encr_algs aes encr_auth_algs sha1
system2 # ifconfig ip.tun0 router up
```

- 管理者はトンネルに対して IPsec ポリシーを有効にします。このポリシーは、553 ページの「IPv4 を使用するトンネルモードの IPsec トンネルで VPN を保護する方法」の手順 4 で作成したものです。

```
system1 # svcadm refresh svc:/network/ipsec/policy:default
system2 # svcadm refresh svc:/network/ipsec/policy:default
```

- 管理者はインターネットインタフェースをルーターにし、経路制御プロトコルがイントラネットインタフェースを越えないようにします。

```
system1 # ifconfig hme1 router ; ifconfig hme0 private
system2 # ifconfig hme1 router ; ifconfig hme0 private
```

- 両方のシステムで 553 ページの「IPv4 を使用するトンネルモードの IPsec トンネルで VPN を保護する方法」の手順 12 と手順 22 を実行して、ルーティングを追加します。

例 20–12 LAN 上のすべてのシステムでの IPsec ポリシーの要求

この例では、手順 4 で構成した bypass ポリシーをコメントにして、保護を強化します。このポリシーを構成した場合は、LAN 上のすべてのシステムが、ルーターと通信するために IPsec を有効にしなければなりません。

```
# LAN traffic must implement IPsec.
# {laddr 10.1.3.3 dir both} bypass {}

# WAN traffic uses ESP with AES and SHA-1.
{tunnel ip.tun0 negotiate tunnel} ipsec {encr_algs aes encr_auth_algs sha1}
```

例 20–13 IPsec を使用して Telnet トラフィックと SMTP トラフィックを異なる方法で保護する

この例の最初の規則は、ポート 23 の telnet トラフィックを Blowfish と SHA-1 で保護します。2 番目の規則はポート 25 の SMTP トラフィックを AES と MD5 で保護します。

```
{laddr 10.1.3.3 ulp tcp dport 23 dir both}
  ipsec {encr_algs blowfish encr_auth_algs sha1 sa unique}
{laddr 10.1.3.3 ulp tcp dport 25 dir both}
 ipsec {encr_algs aes encr_auth_algs md5 sa unique}
```

例 20–14 トンネルモードの IPsec トンネルを使用して、ほかのネットワークトラフィックとは異なる方法でサブネットを保護する

次のトンネル構成は、トンネルを経由するサブネット 10.1.3.0/24 からのすべてのトラフィックを保護します。

```
{tunnel ip.tun0 negotiate tunnel laddr 10.1.3.0/24}
  ipsec {encr_algs aes encr_auth_algs sha1 sa shared}
```

次のトンネル構成は、トンネルを経由するサブネット 10.1.3.0/24 からほかのサブネットへのトラフィックを保護します。10.2.*x.x* で始まるサブネットはトンネルを経由します。

```
{tunnel ip.tun0 negotiate tunnel laddr 10.1.3.0/24 raddr 10.2.1.0/24}
  ipsec {encr_algs blowfish encr_auth_algs sha1 sa shared}

{tunnel ip.tun0 negotiate tunnel laddr 10.1.3.0/24 raddr 10.2.2.0/24}
  ipsec {encr_algs blowfish encr_auth_algs sha1 sa shared}

{tunnel ip.tun0 negotiate tunnel laddr 10.1.3.0/24 raddr 10.2.3.0/24}
  ipsec {encr_algs aes encr_auth_algs sha1 sa shared}
```

▼ IPv6 を使用するトンネルモードの IPsec トンネルで VPN を保護する方法

IPv6 ネットワークで VPN を使用するには、IPv4 ネットワークの場合と同じ手順を実行します。ただし、コマンドの構文は少し違います。特定のコマンドを実行する理由についての詳細は、553 ページの「IPv4 を使用するトンネルモードの IPsec トンネルで VPN を保護する方法」の該当する手順を参照してください。

注 – 両方のシステムでこの手順を実行してください。

この手順では、次の構成パラメータを使用します。

パラメータ	ヨーロッパ	カリフォルニア
システム名	enigma	partym
システムイントラネットインタフェース	hme1	hme1
システムインターネットインタフェース	hme0	hme0
システムイントラネットアドレス	6000:6666::aaaa:1116	6000:3333::eeee:1113
システムインターネットアドレス	2001::aaaa:6666:6666	2001::eeee:3333:3333
インターネットルーターの名前	router-E	router-C
インターネットルーターのアドレス	2001::aaaa:0:4	2001::eeee:0:1
トンネル名	ip6.tun0	ip6.tun0

1 システムコンソール上で、**Primary Administrator** の役割を引き受けるか、スーパーユーザーになります。

Primary Administrator 役割には、Primary Administrator プロファイルが含まれます。役割を作成してユーザーに役割を割り当てるには、『Oracle Solaris の管理: 基本管理』の第 2 章「Solaris 管理コンソールの操作 (タスク)」を参照してください。

注 – リモートログインすると、セキュリティー上重要なトラフィックが盗聴される恐れがあります。何らかの方法でリモートログインを保護していても、システムのセキュリティーがリモートログインセッションレベルに低下します。セキュリティー保護されたリモートログインには、ssh コマンドを使用してください。

2 **IPsec を構成する前に、パケットフローを制御します。**

これらのコマンドの効果については、553 ページの「IPv4 を使用するトンネルモードの IPsec トンネルで VPN を保護する方法」の手順 2 を参照してください。

a. **IP 転送と IP 動的経路制御が無効になっていることを確認します。**

```
# routeadm
Configuration       Current         Current
       Option       Configuration  System State
---------------------------------------------------
...
IPv6 forwarding     disabled          disabled
   IPv6 routing     disabled          disabled
```

IP 転送や IP 動的経路制御が有効な場合は、次のように入力して無効にします。

```
# routeadm -d ipv6-forwarding -d ipv6-routing
# routeadm -u
```

b. **次のコマンドを入力して IP の厳密宛先マルチホームをオンに設定します。**

```
# ndd -set /dev/ip ip6_strict_dst_multihoming 1
```

注意 – システムのブート時に、`ip6_strict_dst_multihoming` の値はデフォルトに戻ります。変更した値を持続させる方法については、583 ページの「IP のスプーフィングを防止する方法」を参照してください。

c. **大部分のネットワークサービス、可能な場合はすべてのネットワークサービスを無効にします。**

注 – 「制限付き」SMF プロファイルでシステムをインストールした場合、この手順は省略できます。Secure Shell 以外のネットワークサービスは無効になります。

ネットワークサービスを無効にすると、IP パケットがシステムにダメージを与えるのを防止できます。たとえば、SNMP デーモン、`telnet` 接続、`rlogin` 接続などを最大限に活用できます。

次のオプションのいずれかを選択します。

- Solaris 10 11/06 リリースまたはそれ以降のリリースが稼働している場合は、「制限付き」SMF プロファイルを実行します。

```
# netservices limited
```

- それ以外の場合は、ネットワークサービスを個別に無効にします。

```
# svcadm disable network/ftp:default
# svcadm disable network/finger:default
# svcadm disable network/login:rlogin
# svcadm disable network/nfs/server:default
```

```
# svcadm disable network/rpc/rstat:default
# svcadm disable network/smtp:sendmail
# svcadm disable network/telnet:default
```

d. ほとんどのネットワークサービスが無効になっていることを確認します。

ループバックマウントと ssh サービスが稼働していることを確認します。

```
# svcs | grep network
online         Aug_02   svc:/network/loopback:default
...
online         Aug_09   svc:/network/ssh:default
```

3 **2 つのシステム間に SA ペアを追加します。**

次のオプションのいずれかを選択します。

- SA 用のキーを管理するように IKE を構成します。603 ページの「IKE の構成 (タスクマップ)」のいずれかの手順に従って、VPN 用の IKE を構成します。
- キーを手動で管理する決定的な理由がある場合は、538 ページの「IPsec セキュリティーアソシエーションを手動で作成する方法」を参照してください。

4 **VPN 用の IPsec ポリシーを追加します。**

/etc/inet/ipsecinit.conf ファイルを編集して、VPN 用の IPsec ポリシーを追加します。

a. **enigma** システムで、**ipsecinit.conf** ファイルに次のエントリを追加します。

```
# IPv6 Neighbor Discovery messages bypass IPsec.
{ulp ipv6-icmp type 133-137 dir both} pass {}

# LAN traffic to and from this host can bypass IPsec.
{laddr 6000:6666::aaaa:1116 dir both} bypass {}

# WAN traffic uses ESP with AES and SHA-1.
{tunnel ip6.tun0 negotiate tunnel}
  ipsec {encr_algs aes encr_auth_algs sha1 sa shared}
```

b. **partym** システムで、**ipsecinit.conf** ファイルに次のエントリを追加します。

```
# IPv6 Neighbor Discovery messages bypass IPsec.
{ulp ipv6-icmp type 133-137 dir both} pass {}

# LAN traffic to and from this host can bypass IPsec.
{laddr 6000:3333::eeee:1113 dir both} bypass {}

# WAN traffic uses ESP with AES and SHA-1.
{tunnel ip6.tun0 negotiate tunnel}
  ipsec {encr_algs aes encr_auth_algs sha1 sa shared}
```

5 **(省略可能) IPsec ポリシーファイルの構文を確認します。**

```
# ipsecconf -c -f /etc/inet/ipsecinit.conf
```

6 トンネルを構成し、それを **IPsec** で保護する場合は、**Oracle Solaris** のリリースに応じて次の手順に従います。

- **Solaris 10 4/09** リリース以降では、手順 **7** から手順 **13** までを実行したあと、手順 **22** の経路制御プロトコルを実行します。
- **Solaris 10 4/09** リリースより前のリリースを実行している場合は、手順 **14** から手順 **22** までを実行します。

7 **/etc/hostname.ip6.tun0** ファイルで、トンネル **ip6.tun0** を構成します。

a. **enigma** システムで、**hostname.ip6.tun0** ファイルに次のエントリを追加します。

```
6000:6666::aaaa:1116 6000:3333::eeee:1113 tsrc 2001::aaaa:6666:6666 tdst 2001::eeee:3333:3333 router up
```

b. **partym** システムで、**hostname.ip6.tun0** ファイルに次のエントリを追加します。

```
6000:3333::eeee:1113  6000:6666::aaaa:1116 tsrc 2001::eeee:3333:3333 tdst 2001::aaaa:6666:6666 router up
```

8 作成した **IPsec** ポリシーでトンネルを保護します。

```
# svcadm refresh svc:/network/ipsec/policy:default
```

9 トンネル構成ファイルの内容をカーネルに読み込むには、ネットワークサービスを再起動します。

```
# svcadm restart svc:/network/initial:default
```

10 **hme1** インタフェースの **IP** 転送をオンに設定します。

a. **enigma** システムで、**/etc/hostname6.hme1** ファイルにルーターエントリを追加します。

```
2001::aaaa:6666:6666 inet6 router
```

b. **partym** システムで、**/etc/hostname6.hme1** ファイルにルーターエントリを追加します。

```
2001::eeee:3333:3333 inet6 router
```

11 経路制御プロトコルによってイントラネット内のデフォルトのルートが通知されていないことを確認します。

a. **enigma** システムで、**private** フラグを **/etc/hostname6.hme0** ファイルに追加します。

```
6000:6666::aaaa:1116 inet6 private
```

b. **partym** システムで、**private** フラグを **/etc/hostname6.hme0** ファイルに追加します。

```
6000:3333::eeee:1113 inet6 private
```

12 **hme0** 経由のデフォルトルートを手動で追加します。

a. **enigma** システムで、次のルートを追加します。

```
# route add -inet6 default 2001::aaaa:0:4
```

b. **partym** システムで次のルートを追加します。

```
# route add -inet6 default 2001::eeee:0:1
```

13 手順を完了するために、手順 22 に進んで経路制御プロトコルを実行します。

14 セキュアトンネル **ip6.tun0** を構成します。

注 – 次の手順は、Solaris 10 4/09 リリースより前のリリースを実行しているシステムでトンネルを構成するためのものです。

---

a. **enigma** システムで、次のコマンドを入力します。

```
# ifconfig ip6.tun0 inet6 plumb

# ifconfig ip6.tun0 inet6 6000:6666::aaaa:1116 6000:3333::eeee:1113 \
tsrc 2001::aaaa:6666:6666   tdst 2001::eeee:3333:3333
```

b. **partym** システムで、次のコマンドを入力します。

```
# ifconfig ip6.tun0 inet6 plumb

# ifconfig ip6.tun0 inet6 6000:3333::eeee:1113 6000:6666::aaaa:1116 \
tsrc 2001::eeee:3333:3333   tdst 2001::aaaa:6666:6666
```

15 作成した **IPsec** ポリシーでトンネルを保護します。

```
# ipsecconf
```

16 トンネル用のルーターを起動します。

```
# ifconfig ip6.tun0 router up
```

17 各システムで、**hme1** インタフェースの **IP** 転送をオンに設定します。

```
# ifconfig hme1 router
```

18 経路制御プロトコルによってイントラネット内のデフォルトのルートが通知されていないことを確認します。

```
# ifconfig hme0 private
```

19 **hme0** 経由のデフォルトルートを手動で追加します。
デフォルトルートは、インターネットに直接アクセスできるルーターでなければなりません。

a. **enigma** システムで、次のルートを追加します。

```
# route add -inet6 default 2001::aaaa:0:4
```

b. **partym** システムで、次のルートを追加します。

```
# route add -inet6 default 2001::eeee:0:1
```

20 **VPN** がリブート後に開始するように、**/etc/hostname6.ip6.tun0** ファイルにエントリを追加します。
このエントリは、手順 14 で ifconfig コマンドに渡されたパラメータを複製します。

a. **enigma** システムで、**hostname6.ip6.tun0** ファイルに次のエントリを追加します。

```
6000:6666::aaaa:1116 6000:3333::eeee:1113 \
tsrc 2001::aaaa:6666:6666  tdst 2001::eeee:3333:3333 router up
```

b. **partym** システムでは、次のエントリを **hostname6.ip6.tun0** ファイルに追加します。

```
6000:3333::eeee:1113 6000:6666::aaaa:1116 \
tsrc 2001::eeee:3333:3333   tdst 2001::aaaa:6666:6666  router up
```

21 各システムで、適切なパラメータが経路制御デーモンに渡されるようにインタフェースファイルを構成します。

a. **enigma** システムで、**/etc/hostname6.***interface* ファイルを変更します。

```
# cat /etc/hostname6.hme0
## enigma
6000:6666::aaaa:1116 inet6 private

#  cat /etc/hostname6.hme1
## enigma
2001::aaaa:6666:6666 inet6 router
```

b. **partym** システムで、**/etc/hostname6.***interface* ファイルを変更します。

```
# cat /etc/hostname6.hme0
## partym
6000:3333::eeee:1113 inet6 private

# cat /etc/hostname6.hme1
## partym
2001::eeee:3333:3333 inet6 router
```

22 経路制御プロトコルを実行します。

```
# routeadm -e ipv6-routing
# routeadm -u
```

経路制御プロトコルを実行する前に経路制御プロトコルの構成が必要な場合があります。詳細は、261 ページの「Oracle Solaris のルーティングプロトコル」を参照してください。手順については、183 ページの「IPv6 ルーターの構成」を参照してください。

▼ IPv4 を使用するトランスポートモードの IPsec トンネルで VPN を保護する方法

トランスポートモードでは、外側のヘッダーによって、内側の IP パケットを保護する IPsec ポリシーが決まります。

この手順は、529 ページの「IPsec で 2 つのシステム間のトラフィックを保護するには」の手順の応用です。この手順では、2 つのシステムを接続するだけでなく、これら 2 つのシステムに接続している 2 つのイントラネットを接続します。この手順における 2 つのシステムはゲートウェイとして機能します。

この手順では、552 ページの「IPsec で VPN を保護するタスクのためのネットワークトポロジの説明」で説明されている設定を使用します。特定のコマンドを実行する理由についての詳細は、553 ページの「IPv4 を使用するトンネルモードの IPsec トンネルで VPN を保護する方法」の該当する手順を参照してください。

注 – 両方のシステムでこの手順を実行してください。

---

1 システムコンソール上で、**Primary Administrator** の役割を引き受けるか、スーパーユーザーになります。

Primary Administrator 役割には、Primary Administrator プロファイルが含まれます。役割を作成してユーザーに役割を割り当てるには、『Oracle Solaris の管理: 基本管理』の第 2 章「Solaris 管理コンソールの操作 (タスク)」を参照してください。

注 – リモートログインすると、セキュリティー上重要なトラフィックが盗聴される恐れがあります。何らかの方法でリモートログインを保護していても、システムのセキュリティーがリモートログインセッションレベルに低下します。セキュリティー保護されたリモートログインには、`ssh` コマンドを使用してください。

---

2 **IPsec** を構成する前に、パケットフローを制御します。

a. **IP** 転送と **IP** 動的経路制御が無効になっていることを確認します。

```
# routeadm
Configuration       Current         Current
       Option       Configuration  System State
---------------------------------------------------
IPv4 forwarding     disabled           disabled
   IPv4 routing     default (enabled)   enabled
...
```

IP 転送や IP 動的経路制御が有効な場合は、次のように入力して無効にします。

```
# routeadm -d ipv4-routing -d ipv4-forwarding
# routeadm -u
```

b. 次のコマンドを入力して **IP** の厳密宛先マルチホームをオンに設定します。

```
# ndd -set /dev/ip ip_strict_dst_multihoming 1
```

注意 – システムのブート時に、`ip_strict_dst_multihoming` の値はデフォルトに戻ります。変更した値を持続させる方法については、583 ページの「IP のスプーフィングを防止する方法」を参照してください。

c. 大部分のネットワークサービス、可能な場合はすべてのネットワークサービスを無効にします。

注 – 「制限付き」SMF プロファイルでシステムをインストールした場合、この手順は省略できます。Secure Shell 以外のネットワークサービスは無効になります。

ネットワークサービスを無効にすると、IP パケットがシステムにダメージを与えるのを防止できます。たとえば、SNMP デーモン、`telnet` 接続、`rlogin` 接続などを最大限に活用できます。

次のオプションのいずれかを選択します。

- Solaris 10 11/06 リリースまたはそれ以降のリリースが稼働している場合は、「制限付き」SMF プロファイルを実行します。

  ```
  # netservices limited
  ```

- それ以外の場合は、ネットワークサービスを個別に無効にします。

  ```
  # svcadm disable network/ftp:default
  # svcadm disable network/finger:default
  # svcadm disable network/login:rlogin
  # svcadm disable network/nfs/server:default
  # svcadm disable network/rpc/rstat:default
  # svcadm disable network/smtp:sendmail
  # svcadm disable network/telnet:default
  ```

d. ほとんどのネットワークサービスが無効になっていることを確認します。

ループバックマウントと `ssh` サービスが稼働していることを確認します。

```
# svcs | grep network
online         Aug_02   svc:/network/loopback:default
...
online         Aug_09   svc:/network/ssh:default
```

3 **2** つのシステム間に **SA** ペアを追加します。

次のオプションのいずれかを選択します。

- SA 用のキーを管理するように IKE を構成します。603 ページの「IKE の構成 (タスクマップ)」のいずれかの手順に従って、VPN 用の IKE を構成します。
- キーを手動で管理する決定的な理由がある場合は、538 ページの「IPsec セキュリティーアソシエーションを手動で作成する方法」を参照してください。

4 **IPsec** ポリシーを追加します。

`/etc/inet/ipsecinit.conf` ファイルを編集して、VPN 用の IPsec ポリシーを追加します。ポリシーを強化する方法については、例 20–15 を参照してください。

a. **enigma** システムで、**ipsecinit.conf** ファイルに次のエントリを入力します。

```
# LAN traffic to and from this host can bypass IPsec.
{laddr 10.16.16.6 dir both} bypass {}

# WAN traffic uses ESP with AES and SHA-1.
{tunnel ip.tun0 negotiate transport}
 ipsec {encr_algs aes encr_auth_algs sha1 sa shared}
```

b. **partym** システムで、**ipsecinit.conf** ファイルに次のエントリを追加します。

```
# LAN traffic to and from this host can bypass IPsec.
{laddr 10.1.3.3 dir both} bypass {}

# WAN traffic uses ESP with AES and SHA-1.
{tunnel ip.tun0 negotiate transport}
 ipsec {encr_algs aes encr_auth_algs sha1 sa shared}
```

5 (省略可能) **IPsec** ポリシーファイルの構文を確認します。

```
# ipsecconf -c -f /etc/inet/ipsecinit.conf
```

6 トンネルを構成し、それを **IPsec** で保護する場合は、**Oracle Solaris** のリリースに応じて次の手順に従います。

- **Solaris 10 4/09** リリース以降では、手順 **7** から手順 **13** までを実行したあと、手順 **22** の経路制御プロトコルを実行します。
- **Solaris 10 4/09** リリースより前のリリースを実行している場合は、手順 **14** から手順 **22** までを実行します。

7 **/etc/hostname.ip.tun0** ファイルで、トンネル **ip.tun0** を構成します。

a. **enigma** システムで、**hostname.ip.tun0** ファイルに次のエントリを追加します。

```
10.16.16.6 10.1.3.3 tsrc 192.168.116.16 tdst 192.168.13.213 router up
```

b. **partym** システムで、**hostname.ip.tun0** ファイルに次のように追加します。

```
10.1.3.3 10.16.16.6 tsrc 192.168.13.213 tdst 192.168.116.16 router up
```

8 作成した **IPsec** ポリシーでトンネルを保護します。

```
# svcadm refresh svc:/network/ipsec/policy:default
```

9 **hostname.ip.tun0** ファイルの内容をカーネルに読み込むには、ネットワークサービスを再起動します。

```
# svcadm restart svc:/network/initial:default
```

10 **hme1** インタフェースの **IP** 転送をオンに設定します。

a. **enigma** システムで、ルーターエントリを **/etc/hostname.hme1** ファイルに追加します。

```
192.168.116.16 router
```

b. **partym** システムで、ルーターエントリを **/etc/hostname.hme1** ファイルに追加します。

```
192.168.13.213 router
```

11 経路制御プロトコルによってイントラネット内のデフォルトのルートが通知されていないことを確認します。

a. **enigma** システムで、**private** フラグを **/etc/hostname.hme0** ファイルに追加します。

```
10.16.16.6 private
```

b. **partym** システムで、**private** フラグを **/etc/hostname.hme0** ファイルに追加します。

```
10.1.3.3 private
```

12 **hme0** 経由のデフォルトルートを手動で追加します。

a. **enigma** システムで、次のルートを追加します。

```
# route add default 192.168.116.4
```

b. **partym** システムで次のルートを追加します。

```
# route add default 192.168.13.5
```

13 手順を完了するために、手順 **22** に進んで経路制御プロトコルを実行します。

14 トンネル **ip.tun0** を構成します。

注 – 次の手順は、Solaris 10 4/09 リリースより前のリリースを実行しているシステムでトンネルを構成するためのものです。

---

ifconfig コマンドを使用してポイントツーポイントインタフェースを作成します。

```
# ifconfig ip.tun0 plumb

# ifconfig ip.tun0 system1-point system2-point \
tsrc system1-taddr tdst system2-taddr
```

a. **enigma** システムで、次のコマンドを入力します。

```
# ifconfig ip.tun0 plumb

# ifconfig ip.tun0 10.16.16.6 10.1.3.3 \
tsrc 192.168.116.16 tdst 192.168.13.213
```

b. **partym** システムで、次のコマンドを入力します。

```
# ifconfig ip.tun0 plumb

# ifconfig ip.tun0 10.1.3.3 10.16.16.6  \
tsrc 192.168.13.213 tdst 192.168.116.16
```

15 作成した **IPsec** ポリシーでトンネルを保護します。

```
# ipsecconf
```

16 トンネル用のルーターを起動します。

```
# ifconfig ip.tun0 router up
```

17 **hme1** インタフェースの **IP** 転送をオンに設定します。

```
# ifconfig hme1 router
```

18 経路制御プロトコルによってイントラネット内のデフォルトのルートが通知されていないことを確認します。

```
# ifconfig hme0 private
```

19 **hme0** 経由のデフォルトルートを手動で追加します。

デフォルトルートは、インターネットに直接アクセスできるルーターでなければなりません。

```
# route add default router-on-hme0-subnet
```

a. **enigma** システムで、次のルートを追加します。

```
# route add default 192.168.116.4
```

b. **partym** システムで、次のルートを追加します。

```
# route add default 192.168.13.5
```

20 **VPN** がリブート後に開始するように、**/etc/hostname.ip.tun0** ファイルにエントリを追加します。

system1-point system2-point tsrc *system1-taddr* \
tdst *system2-taddr* encr_algs aes encr_auth_algs sha1 router up

a. **enigma** システムで、**hostname.ip.tun0** ファイルに次のエントリを追加します。

```
10.16.16.6 10.1.3.3 tsrc 192.168.116.16 \
tdst 192.168.13.213 router up
```

b. **partym** システムで、**hostname.ip.tun0** ファイルに次のように追加します。

```
10.1.3.3 10.16.16.6 tsrc 192.168.13.213 \
tdst 192.168.116.16 router up
```

21 適切なパラメータが経路制御デーモンに渡されるようにインタフェースファイルを構成します。

a. **enigma** システムで、**/etc/hostname.***interface* ファイルを変更します。

```
# cat /etc/hostname.hme0
## enigma
10.16.16.6 private

# cat /etc/hostname.hme1
## enigma
192.168.116.16 router
```

b. **partym** システムで、**/etc/hostname.***interface* ファイルを変更します。

```
# cat /etc/hostname.hme0
## partym
10.1.3.3 private

# cat /etc/hostname.hme1
## partym
192.168.13.213 router
```

22 経路制御プロトコルを実行します。

```
# routeadm -e ipv4-routing
# routeadm -u
```

例 20–15 すべてのシステムにトランスポートモードで IPsec ポリシーを要求する

この例では、手順 4 で構成した bypass ポリシーをコメントにして、保護を強化します。このポリシーを構成した場合は、LAN 上のすべてのシステムが、ルーターと通信するために IPsec を有効にしなければなりません。

```
# LAN traffic must implement IPsec.
# {laddr 10.1.3.3 dir both} bypass {}

# WAN traffic uses ESP with AES and SHA-1.
{tunnel ip.tun0 negotiate transport} ipsec {encr_algs aes encr_auth_algs sha1}
```

例 20–16 推奨されなくなった構文を使用してトランスポートモードの IPsec トンネルを構成する

この例では、Solaris 10 7/07 システムを、Oracle Solaris 10 リリースを実行しているシステムに接続します。したがって、管理者は構成ファイルで Solaris 10 の構文を使用し、`ifconfig` コマンドに IPsec アルゴリズムを含めます。

管理者は、570 ページの「IPv4 を使用するトランスポートモードの IPsec トンネルで VPN を保護する方法」の手順に従いますが、構文を次のように変更します。

- 手順 4 で、`ipsecinit.conf` ファイルの構文は次のとおりです。

```
# LAN traffic to and from this address can bypass IPsec.
{laddr 10.1.3.3 dir both} bypass {}

# WAN traffic uses ESP with AES and SHA-1.
{} ipsec {encr_algs aes encr_auth_algs sha1}
```

- 手順 14 から手順 16 で、セキュアトンネルを構成するための構文は次のとおりです。

```
# ifconfig ip.tun0 plumb

# ifconfig ip.tun0 10.16.16.6 10.1.3.3 \
tsrc 192.168.116.16 tdst 192.168.13.213 \
encr_algs aes encr_auth_algs sha1

# ifconfig ip.tun0 router up

# ifconfig ip.tun0 plumb

# ifconfig ip.tun0 10.16.16.6 10.1.3.3 \
tsrc 192.168.116.16 tdst 192.168.13.213 \
encr_algs aes encr_auth_algs sha1
```

`ifconfig` コマンドに渡す IPsec ポリシーは、`ipsecinit.conf` ファイルに指定されている IPsec ポリシーと同じでなければなりません。各システムは、リブート時にそのポリシーを含む `ipsecinit.conf` ファイルを読み込みます。

- 手順 20 で、`hostname.ip.tun0` ファイルの構文は次のとおりです。

```
10.16.16.6 10.1.3.3 tsrc 192.168.116.16 \
tdst 192.168.13.213 encr_algs aes encr_auth_algs sha1 router up
```

▼ IPv6 を使用するトランスポートモードの IPsec トンネルで VPN を保護する方法

IPv6 ネットワークで VPN を使用するには、IPv4 ネットワークの場合と同じ手順を実行します。ただし、コマンドの構文は少し違います。特定のコマンドを実行する理由についての詳細は、553 ページの「IPv4 を使用するトンネルモードの IPsec トンネルで VPN を保護する方法」の該当する手順を参照してください。

注 – 両方のシステムでこの手順を実行してください。

この手順では、次の構成パラメータを使用します。

パラメータ	ヨーロッパ	カリフォルニア
システム名	enigma	partym
システムイントラネットインタフェース	hme1	hme1
システムインターネットインタフェース	hme0	hme0
システムイントラネットアドレス	6000:6666::aaaa:1116	6000:3333::eeee:1113
システムインターネットアドレス	2001::aaaa:6666:6666	2001::eeee:3333:3333
インターネットルーターの名前	router-E	router-C
インターネットルーターのアドレス	2001::aaaa:0:4	2001::eeee:0:1
トンネル名	ip6.tun0	ip6.tun0

1 システムコンソール上で、**Primary Administrator** の役割を引き受けるか、スーパーユーザーになります。

Primary Administrator 役割には、Primary Administrator プロファイルが含まれます。役割を作成してユーザーに役割を割り当てるには、『Oracle Solaris の管理: 基本管理』の第 2 章「Solaris 管理コンソールの操作 (タスク)」を参照してください。

注 – リモートログインすると、セキュリティー上重要なトラフィックが盗聴される恐れがあります。何らかの方法でリモートログインを保護していても、システムのセキュリティーがリモートログインセッションレベルに低下します。セキュリティー保護されたリモートログインには、ssh コマンドを使用してください。

2 **IPsec** を構成する前に、パケットフローを制御します。

a. **IP** 転送と **IP** 動的経路制御が無効になっていることを確認します。

```
# routeadm
Configuration       Current         Current
       Option       Configuration  System State
---------------------------------------------------
...
IPv6 forwarding     disabled          disabled
   IPv6 routing     disabled          disabled
```

IP 転送や IP 動的経路制御が有効な場合は、次のように入力して無効にします。

```
# routeadm -d ipv6-forwarding -d ipv6-routing
# routeadm -u
```

b. 次のコマンドを入力して **IP** の厳密宛先マルチホームをオンに設定します。

```
# ndd -set /dev/ip ip6_strict_dst_multihoming 1
```

注意 – システムのブート時に、`ip6_strict_dst_multihoming` の値はデフォルトに戻ります。変更した値を持続させる方法については、583 ページの「IP のスプーフィングを防止する方法」を参照してください。

c. ほとんどのネットワークサービスが無効になっていることを確認します。

ループバックマウントと `ssh` サービスが稼働していることを確認します。

```
# svcs | grep network
online         Aug_02   svc:/network/loopback:default
...
online         Aug_09   svc:/network/ssh:default
```

3 **2** つのシステム間に **SA** ペアを追加します。

次のオプションのいずれかを選択します。

- SA 用のキーを管理するように IKE を構成します。603 ページの「IKE の構成 (タスクマップ)」のいずれかの手順に従って、VPN 用の IKE を構成します。
- キーを手動で管理する決定的な理由がある場合は、538 ページの「IPsec セキュリティーアソシエーションを手動で作成する方法」を参照してください。

4 **IPsec** ポリシーを追加します。

`/etc/inet/ipsecinit.conf` ファイルを編集して、VPN 用の IPsec ポリシーを追加します。

a. **enigma** システムで、**ipsecinit.conf** ファイルに次のエントリを追加します。

```
# IPv6 Neighbor Discovery messages bypass IPsec.
{ulp ipv6-icmp type 133-137 dir both} pass {}

# LAN traffic can bypass IPsec.
{laddr 6000:6666::aaaa:1116 dir both} bypass {}

# WAN traffic uses ESP with AES and SHA-1.
{tunnel ip6.tun0 negotiate transport}
 ipsec {encr_algs aes encr_auth_algs sha1}
```

b. **partym** システムで、**ipsecinit.conf** ファイルに次のエントリを追加します。

```
# IPv6 Neighbor Discovery messages bypass IPsec.
{ulp ipv6-icmp type 133-137 dir both} pass {}

# LAN traffic can bypass IPsec.
```

```
{laddr 6000:3333::eeee:1113 dir both} bypass {}

# WAN traffic uses ESP with AES and SHA-1.
{tunnel ip6.tun0 negotiate transport}
 ipsec {encr_algs aes encr_auth_algs sha1}
```

5 (省略可能) **IPsec** ポリシーファイルの構文を確認します。

```
# ipsecconf -c -f /etc/inet/ipsecinit.conf
```

6 トンネルを構成し、それを **IPsec** で保護する場合は、**Oracle Solaris** のリリースに応じて次の手順に従います。

- **Solaris 10 4/09** リリース以降では、手順 **7** から手順 **13** までを実行したあと、手順 **22** の経路制御プロトコルを実行します。
- **Solaris 10 4/09** リリースより前のリリースを実行している場合は、手順 **14** から手順 **22** までを実行します。

7 **/etc/hostname.ip6.tun0** ファイルで、トンネル **ip6.tun0** を構成します。

a. **enigma** システムで、**hostname.ip6.tun0** ファイルに次のエントリを追加します。

```
6000:6666::aaaa:1116 6000:3333::eeee:1113 tsrc 2001::aaaa:6666:6666 tdst 2001::eeee:3333:3333 router up
```

b. **partym** システムで、**hostname.ip6.tun0** ファイルに次のエントリを追加します。

```
6000:3333::eeee:1113  6000:6666::aaaa:1116 tsrc 2001::eeee:3333:3333 tdst 2001::aaaa:6666:6666 router up
```

8 作成した **IPsec** ポリシーでトンネルを保護します。

```
# svcadm refresh svc:/network/ipsec/policy:default
```

9 **hostname.ip6.tun0** ファイルの内容をカーネルに読み込むには、ネットワークサービスを再起動します。

```
# svcadm restart svc:/network/initial:default
```

10 **hme1** インタフェースの **IP** 転送をオンに設定します。

a. **enigma** システムで、**/etc/hostname6.hme1** ファイルにルーターエントリを追加します。

```
2001::aaaa:6666:6666 inet6 router
```

b. **partym** システムで、**/etc/hostname6.hme1** ファイルにルーターエントリを追加します。

```
2001::eeee:3333:3333 inet6 router
```

11 経路制御プロトコルによってイントラネット内のデフォルトのルートが通知されていないことを確認します。

a. **enigma** システムで、**private** フラグを **/etc/hostname6.hme0** ファイルに追加します。

```
6000:6666::aaaa:1116 inet6 private
```

b. **partym** システムで、**private** フラグを **/etc/hostname6.hme0** ファイルに追加します。

```
6000:3333::eeee:1113 inet6 private
```

12 **hme0** 経由のデフォルトルートを手動で追加します。

a. **enigma** システムで、次のルートを追加します。

```
# route add -inet6 default 2001::aaaa:0:4
```

b. **partym** システムで次のルートを追加します。

```
# route add -inet6 default 2001::eeee:0:1
```

13 手順を完了するために、手順 **22** に進んで経路制御プロトコルを実行します。

14 セキュアトンネル **ip6.tun0** を構成します。

注 – 次の手順は、Solaris 10 4/09 リリースより前のリリースを実行しているシステムでトンネルを構成するためのものです。

---

a. **enigma** システムで、次のコマンドを入力します。

```
# ifconfig ip6.tun0 inet6 plumb

# ifconfig ip6.tun0 inet6 6000:6666::aaaa:1116 6000:3333::eeee:1113 \
tsrc 2001::aaaa:6666:6666   tdst 2001::eeee:3333:3333
```

b. **partym** システムで、次のコマンドを入力します。

```
# ifconfig ip6.tun0 inet6 plumb

# ifconfig ip6.tun0 inet6  6000:3333::eeee:1113  6000:6666::aaaa:1116 \
tsrc 2001::eeee:3333:3333   tdst 2001::aaaa:6666:6666
```

15 作成した **IPsec** ポリシーでトンネルを保護します。

```
# ipsecconf
```

16 トンネル用のルーターを起動します。

```
# ifconfig ip6.tun0 router up
```

17 **hme1** インタフェースの **IP** 転送をオンに設定します。

```
# ifconfig hme1 router
```

18 経路制御プロトコルによってイントラネット内のデフォルトのルートが通知されていないことを確認します。

```
# ifconfig hme0 private
```

19 各システムで、**hme0** 経由のデフォルトルートを手動で追加します。

デフォルトルートは、インターネットに直接アクセスできるルーターでなければなりません。

a. **enigma** システムで、次のルートを追加します。

```
# route add -inet6 default 2001::aaaa:0:4
```

b. **partym** システムで、次のルートを追加します。

```
# route add -inet6 default 2001::eeee:0:1
```

20 各システムで、**VPN** がリブート後に開始するように、**/etc/hostname6.ip6.tun0** ファイルにエントリを追加します。

このエントリは、手順 14 で `ifconfig` コマンドに渡されたパラメータを複製します。

a. **enigma** システムで、**hostname6.ip6.tun0** ファイルに次のエントリを追加します。

```
6000:6666::aaaa:1116  6000:3333::eeee:1113 \
tsrc 2001::aaaa:6666:6666   tdst 2001::eeee:3333:3333  router up
```

b. **partym** システムでは、次のエントリを **hostname6.ip6.tun0** ファイルに追加します。

```
6000:3333::eeee:1113  6000:6666::aaaa:1116 \
tsrc 2001::eeee:3333:3333   tdst 2001::aaaa:6666:6666  router up
```

21 適切なパラメータが経路制御デーモンに渡されるようにインタフェースファイルを構成します。

a. **enigma** システムで、**/etc/hostname6.***interface* ファイルを変更します。

```
# cat /etc/hostname6.hme0
## enigma
6000:6666::aaaa:1116 inet6 private

#  cat /etc/hostname6.hme1
## enigma
2001::aaaa:6666:6666 inet6 router
```

b. **partym** システムで、**/etc/hostname6.***interface* ファイルを変更します。

```
# cat /etc/hostname6.hme0
## partym
6000:3333::eeee:1113 inet6 private

# cat /etc/hostname6.hme1
##
partym2001::eeee:3333:3333 inet6 router
```

22 経路制御プロトコルを実行します。

```
# routeadm -e ipv6-routing
# routeadm -u
```

例 20–17 非推奨の構文を使用して IPv6 を使用するトランスポートモードの IPsec を構成する

この例では、Solaris 10 7/07 システムを、Oracle Solaris 10 リリースを実行しているシステムに接続します。したがって、管理者は構成ファイルで Solaris 10 の構文を使用し、`ifconfig` コマンドに IPsec アルゴリズムを含めます。

管理者は、576 ページの「IPv6 を使用するトランスポートモードの IPsec トンネルで VPN を保護する方法」の手順に従いますが、構文を次のように変更します。

- 手順 4 で、`ipsecinit.conf` ファイルの構文は次のとおりです。

  ```
  # IPv6 Neighbor Discovery messages bypass IPsec.
  {ulp ipv6-icmp type 133-137 dir both} pass {}

  # LAN traffic can bypass IPsec.
  {laddr 6000:3333::eeee:1113 dir both} bypass {}

  # WAN traffic uses ESP with AES and SHA-1.
  {} ipsec {encr_algs aes encr_auth_algs sha1}
  ```

- 手順 14 から手順 17 で、セキュアトンネルを構成するための構文は次のとおりです。

  ```
  # ifconfig ip6.tun0 inet6 plumb

  # ifconfig ip6.tun0 inet6 6000:6666::aaaa:1116 6000:3333::eeee:1113 \
  tsrc 2001::aaaa:6666:6666 tdst 2001::eeee:3333:3333 \
  encr_algs aes encr_auth_algs sha1

  # ifconfig ip6.tun0 inet6 router up
  ```

  `ifconfig` コマンドに渡す IPsec ポリシーは、`ipsecinit.conf` ファイルに指定されている IPsec ポリシーと同じでなければなりません。各システムは、リブート時にそのポリシーを含む `ipsecinit.conf` ファイルを読み込みます。

- 手順 20 で、`hostname6.ip6.tun0` ファイルの構文は次のとおりです。

  ```
  6000:6666::aaaa:1116  6000:3333::eeee:1113 \
  tsrc 2001::aaaa:6666:6666   tdst 2001::eeee:3333:3333 \
  encr_algs aes encr_auth_algs sha1 router up
  ```

▼ IP のスプーフィングを防止する方法

システムがパケットの復号化を試みずに別のインタフェースに転送することを防止するには、IP のスプーフィングをチェックする必要があります。その方法の 1 つは、ndd コマンドを使用して IP 厳密宛先マルチホームのパラメータを設定することです。SMF マニフェストでこのパラメータが設定されている場合、システムのリブート時にこのパラメータが設定されます。

注 – 両方のシステムでこの手順を実行してください。

---

1 システムコンソール上で、**Primary Administrator** の役割を引き受けるか、スーパーユーザーになります。

Primary Administrator 役割には、Primary Administrator プロファイルが含まれます。役割を作成してユーザーに役割を割り当てるには、『Oracle Solaris の管理: 基本管理』の第 2 章「Solaris 管理コンソールの操作 (タスク)」を参照してください。

2 **IP** のスプーフィングをチェックするように、サイト固有の **SMF** マニフェストを作成します。

次のサンプルスクリプト `/var/svc/manifest/site/spoof_check.xml` を使用します。

```
<?xml version="1.0"?>
<!DOCTYPE service_bundle SYSTEM "/usr/share/lib/xml/dtd/service_bundle.dtd.1">

<service_bundle type='manifest' name='Custom:ip_spoof_checking'>

<!--    This is a custom smf(5) manifest for this system. Place this
        file in /var/svc/manifest/site, the directory for local
        system customizations. The exec method uses an unstable
        interface to provide a degree of protection against IP
        spoofing attacks when this system is acting as a router.

        IP spoof protection can also be achieved by using ipfilter(5).
        If ipfilter is configured, this service can be disabled.

        Note: Unstable interfaces might be removed in later
        releases.  See attributes(5).
-->

<service
        name='site/ip_spoofcheck'
        type='service'
        version='1'>

        <create_default_instance enabled='false' />
        <single_instance />

        <!--    Don't enable spoof protection until the
                network is up.
        -->
        <dependency
                name='basic_network'
```

```
                grouping='require_all'
                restart_on='none'
                type='service'>
        <service_fmri value='svc:/milestone/network' />
        </dependency>

        <exec_method
                type='method'
                name='start'
                exec='/usr/sbin/ndd -set /dev/ip ip_strict_dst_multihoming 1'
<!--
     For an IPv6 network, use the IPv6 version of this command, as in:
                exec='/usr/sbin/ndd -set /dev/ip ip6_strict_dst_multihoming 1
-->
                timeout_seconds='60'
        />

        <exec_method
                type='method'
                name='stop'
                exec=':true'
                timeout_seconds='3'
        />

        <property_group name='startd' type='framework'>
                <propval
                        name='duration'
                        type='astring'
                        value='transient'
                />
        </property_group>

        <stability value='Unstable' />

</service>
</service_bundle>
```

3 このマニフェストを **SMF** リポジトリにインポートします。

```
# svccfg import /var/svc/manifest/site/spoof_check.xml
```

4 **ip_spoofcheck** サービスを有効にします。

マニフェストで定義されている名前 /site/ip_spoofcheck を使用します。

```
# svcadm enable /site/ip_spoofcheck
```

5 **ip_spoofcheck** サービスがオンラインになっていることを確認します。

```
# svcs /site/ip_spoofcheck
```

◆ ◆ ◆ 第 21 章

IP セキュリティーアーキテクチャー (リファレンス)

この章では、次の内容について説明します。

使用しているネットワークに IPsec を実装する方法については、第 20 章「IPsec の構成 (タスク)」を参照してください。IPsec の概要については、第 19 章「IP セキュリティーアーキテクチャー (概要)」を参照してください。

IPsec サービス

サービス管理機能 (SMF) は、次の IPsec サービスを提供します。

- svc:/network/ipsec/policy サービス – IPsec ポリシーを管理します。デフォルトでは、このサービスは有効になっています。config_file プロパティーの値によって ipsecinit.conf ファイルの場所が決まります。初期値は /etc/inet/ipsecinit.conf です。
- svc:/network/ipsec/ipsecalgs サービス – IPsec で使用できるアルゴリズムを管理します。デフォルトでは、このサービスは有効になっています。
- svc:/network/ipsec/manual-key サービス – 手動での鍵管理を有効にします。デフォルトでは、このサービスは無効になっています。config_file プロパティーの値によって ipseckeys 構成ファイルの場所が決まります。初期値は /etc/inet/secret/ipseckeys です。

- svc:/network/ipsec/ike サービス – IKE を管理します。デフォルトでは、このサービスは無効になっています。構成可能なプロパティーについては、649 ページの「IKE サービス」を参照してください。

SMF の詳細については、『Oracle Solaris の管理: 基本管理』の第 18 章「サービスの管理 (概要)」を参照してください。smf(5)、svcadm(1M)、および svccfg(1M) のマニュアルページも参照してください。

ipsecconf コマンド

ホストの IPsec ポリシーを構成するには、ipsecconf コマンドを使用します。このコマンドを実行してポリシーを構成すると、IPsec ポリシーのエントリがカーネル内に作成されます。システムは、これらのエントリを使用して、インバウンドおよびアウトバウンドの IP データグラムすべてがポリシーに沿っているかどうかを検査します。転送されたデータグラムは、このコマンドで追加されたポリシー検査の対象外になります。また、ipsecconf コマンドはセキュリティーポリシーデータベース (SPD) を構成します。

- 転送されたパケットを保護する方法については、ifconfig(1M) と tun(7M) のマニュアルページを参照してください。
- IPsec ポリシーオプションについては、ipsecconf(1M) のマニュアルページを参照してください。

ipsecconf コマンドを呼び出すには、スーパーユーザーになるか、同等の役割を引き受ける必要があります。このコマンドは、両方向のトラフィックを保護するエントリを受け入れます。このコマンドは、片方向だけのトラフィックを保護するエントリも受け入れます。

ローカルアドレスとリモートアドレスというパターンのポリシーエントリは、1 つのポリシーエントリで両方向のトラフィックを保護します。たとえば、指定されたホストに対して方向が指定されていない場合、laddr host1 と raddr host2 というパターンを含むエントリは、両方向のトラフィックを保護します。そのため、各ホストにポリシーエントリを 1 つだけ設定すれば済みます。

ソースアドレスから宛先アドレスへというパターンのポリシーエントリは、1 方向のみのトラフィックを保護します。たとえば、saddr host1 daddr host2 というパターンのポリシーエントリは、インバウンドかアウトバウンドのどちらかのトラフィックのみを保護します。両方向ともは保護しません。したがって、両方向のトラフィックを保護するには、saddr host2 daddr host1 という先ほどとは逆方向のエントリを ipsecconf コマンドに渡す必要があります。

マシンがブートするときに IPsec ポリシーが確実にアクティブになるようにするには、IPsec ポリシーファイル /etc/inet/ipsecinit.conf を作成します。このファイルは、ネットワークサービスが起動するときに読み取られます。IPsec ポリシーファイルを作成する方法については、527 ページの「IPsec によるトラフィックの保護 (タスクマップ)」を参照してください。

Solaris 10 4/09 リリース以降では、-c オプションを指定して ipsecconf コマンドを実行すると、引数として指定した IPsec ポリシーファイルの構文がチェックされます。

ipsecconf コマンドで追加されたポリシーエントリには持続性がなく、システムのリブート時に失われます。システムのブート時に IPsec ポリシーが確実にアクティブになるようにするには、/etc/inet/ipsecinit.conf ファイルにポリシーエントリを追加します。現在のリリースでは、policy サービスをリフレッシュするか有効にします。Solaris 10 4/09 リリースより前のリリースでは、リブートするか ipsecconf コマンドを使用します。例については、527 ページの「IPsec によるトラフィックの保護 (タスクマップ)」を参照してください。

ipsecinit.conf ファイル

Oracle Solaris を起動したときに IPsec セキュリティーポリシーを有効化するには、特定の IPsec ポリシーエントリを使用して構成ファイルを作成し IPsec を初期化します。このファイルのデフォルトの名前は /etc/inet/ipsecinit.conf です。ポリシーエントリとその形式の詳細については、ipsecconf(1M) のマニュアルページを参照してください。ポリシーの構成後、ipsecconf コマンドを使用すると、既存の構成を表示または変更できます。Solaris 10 4/09 リリース以降では、policy サービスをリフレッシュすることによって、既存の構成を変更します。

サンプルの ipsecinit.conf ファイル

Oracle Solaris ソフトウェアには、サンプルの IPsec ポリシーファイル ipsecinit.sample が含まれます。このファイルをテンプレートとして独自の ipsecinit.conf ファイルを作成できます。ipsecinit.sample ファイルには、次のエントリが含まれています。

```
#
# For example,
#
#     {rport 23} ipsec {encr_algs des encr_auth_algs md5}
#
# will protect the telnet traffic originating from the host with ESP using
# DES and MD5. Also:
#
#     {raddr 10.5.5.0/24} ipsec {auth_algs any}
#
# will protect traffic to or from the 10.5.5.0 subnet with AH
# using any available algorithm.
#
#
# To do basic filtering, a drop rule may be used. For example:
#
#     {lport 23 dir in} drop {}
#     {lport 23 dir out} drop {}
# will disallow any remote system from telnetting in.
```

```
#
# If you are using IPv6, it may be useful to bypass neighbor discovery
# to allow in.iked to work properly with on-link neighbors. To do that,
# add the following lines:
#
#       {ulp ipv6-icmp type 133-137 dir both } pass { }
#
# This will allow neighbor discovery to work normally.
```

ipsecinit.conf と ipsecconf のセキュリティーについて

ipsecinit.conf ファイルをネットワーク経由で転送するときには、特に注意する必要があります。ネットワークマウントファイルが読み取られている場合、不正に読み取られる可能性があります。たとえば、NFS マウントされたファイルシステムから /etc/inet/ipsecinit.conf ファイルをアクセスまたはコピーしようとした場合、このファイルに含まれるポリシーは不正に変更される可能性があります。

確立された接続の IPsec ポリシーを変更することはできません。ポリシーの変更ができないソケットを、「ラッチされたソケット」と呼びます。新しいポリシーエントリは、すでにラッチされたソケットを保護しません。詳細については、connect(3SOCKET) と accept(3SOCKET) のマニュアルページを参照してください。自信がない場合は、接続を再起動してください。

ネーミングシステムを保護してください。次の 2 つの条件に該当する場合、そのホスト名は信頼できません。

- ソースアドレスが、ネットワークを介して参照できるホストである。
- ネーミングシステムの信頼性に問題がある。

セキュリティーの弱点の多くは、実際のツールではなく、ツールの使用方法にあります。ipsecconf コマンドを使用するときは注意が必要です。各操作の最も安全なモードでコンソールを使用するか、ハード接続の TTY を使用してください。

ipsecalgs コマンド

暗号化フレームワークは、認証と暗号化のアルゴリズムを IPsec に提供します。ipsecalgs コマンドを使用すると、各 IPsec プロトコルでサポートされているアルゴリズムを一覧表示できます。ipsecalgs の構成は /etc/inet/ipsecalgs ファイルに保存されます。通常、このファイルを変更する必要はありません。ただし、このファイルを変更する必要がある場合は、ipsecalgs コマンドを使用します。決して直接には編集しないでください。現在のリリースでは、サポートされるアルゴリズムは、システムのブート時に svc:/network/ipsec/ipsecalgs:default サービスによってカーネルと同期されます。

有効な IPsec プロトコルおよびアルゴリズムは、RFC 2407 に記載されている ISAKMP 解釈ドメイン (DOI) によって記述されます。一般的な意味では、DOI は、データ形式、ネットワークトラフィック交換タイプ、およびセキュリティー関連情報の命名規約を定義します。セキュリティー関連情報の例としては、セキュリティーポリシーや、暗号化アルゴリズム、暗号化モードなどがあります。

具体的には、ISAKMP DOI は、有効な IPsec アルゴリズムとそのプロトコル (`PROTO_IPSEC_AH` と `PROTO_IPSEC_ESP`) の命名規則と番号付け規則を定義します。1 つのアルゴリズムは 1 つのプロトコルだけに関連します。このような ISAKMP DOI 定義は、`/etc/inet/ipsecalgs` ファイルにあります。アルゴリズム番号とプロトコル番号は、Internet Assigned Numbers Authority (IANA) によって定義されます。`ipsecalgs` コマンドは、IPsec アルゴリズムのリストを拡張します。

アルゴリズムの詳細については、`ipsecalgs(1M)` のマニュアルページを参照してください。暗号化フレームワークの詳細については、『Solaris のシステム管理: セキュリティーサービス』の第 13 章「Oracle Solaris の暗号化フレームワーク (概要)」を参照してください。

IPsec のセキュリティーアソシエーションデータベース

IPsec セキュリティーサービスの鍵情報は、セキュリティーアソシエーションデータベース (SADB) に保存されます。セキュリティーアソシエーション (SA) は、インバウンドパケットとアウトバウンドパケットを保護します。SADB の保守は、1 つまたは複数の (そしておそらくは協力する) ユーザープロセスがメッセージを特殊なソケット経由で送信することによって行われます。SADB を保守するこの方法は、`route(7P)` のマニュアルページで説明している方法に類似しています。SADB にアクセスできるのは、スーパーユーザーか、同等の役割を引き受けたユーザーだけです。

`in.iked` デーモンと `ipseckey` コマンドは `PF_KEY` ソケットインタフェースを使用して SADB を保守します。SADB が要求やメッセージを処理する方法の詳細については、`pf_key(7P)` のマニュアルページを参照してください。

IPsec の SA を生成するためのユーティリティー

IKE プロトコルは、IPv4 アドレスおよび IPv6 アドレスの鍵を自動的に管理します。IKEを設定する方法については、第 23 章「IKE の構成 (タスク)」を参照してください。手動キーイングユーティリティーは `ipseckey` コマンドです (`ipseckey(1M)` のマニュアルページを参照)。

セキュリティーアソシエーションデータベース (SADB) を手動で生成するには、`ipseckey` コマンドを使用します。通常、手動での SA 生成は、何らかの理由で IKE を使用できない場合に使用します。ただし、SPI の値が一意であれば、手動での SA 生成と IKE を同時に使用することができます。

ipseckey コマンドを使用すると、鍵が手動で追加された場合でも、IKE によって追加された場合でも、システムで認識されているすべての SA を表示できます。Solaris 10 4/09 リリース以降では、-c オプションを指定して ipseckey コマンドを実行すると、引数として指定した鍵ファイルの構文がチェックされます。

ipseckey コマンドで追加された IPsec SA には持続性がなく、システムのリブート時に失われます。現在のリリースでは、手動で追加した SA をシステムのブート時に有効にするには、/etc/inet/secret/ipseckeys ファイルにエントリを追加してから、svc:/network/ipsec/manual-key:default サービスを有効にします。手順については、538 ページの「IPsec セキュリティーアソシエーションを手動で作成する方法」を参照してください。

ipseckey コマンドには少数の一般オプションしかありませんが、多くのコマンド言語をサポートしています。マニュアルキー操作に固有のプログラムインタフェースで要求を配信するように指定することもできます。詳細については、pf_key(7P) のマニュアルページを参照してください。

ipseckey におけるセキュリティーについて

ipseckey コマンドを使用すると、スーパーユーザーまたは Network Security または Network IPsec Management 権利プロファイルの役割を引き受けたユーザーは、暗号鍵に関する機密情報を入力できます。場合によっては、不正にこの情報にアクセスして IPsec トラフィックのセキュリティーを損なうことも可能です。

注 – 可能であれば、ipseckey による手動のキーイングではなく、IKE を使用してください。

鍵情報を扱う場合および ipseckey コマンドを使用する場合には、次のことに注意してください。

- 鍵情報をリフレッシュしているかどうか。定期的に鍵をリフレッシュすることが、セキュリティーの基本作業となります。鍵をリフレッシュすることで、アルゴリズムと鍵の脆弱性が暴かれないように保護し、公開された鍵の侵害を制限します。
- TTY がネットワークに接続されているか。ipseckey コマンドは対話モードで実行されているか。
  - 対話モードの場合、鍵情報のセキュリティーは、TTY のトラフィックに対応するネットワークパスのセキュリティーになります。平文の telnet や rlogin セッションでは、ipseckey コマンドを使用しないでください。
  - ローカルウィンドウでも、ウィンドウを読み取ることのできる隠密プログラムからの攻撃には無防備です。
- -f オプションを使用しているか。ファイルはネットワークを介してアクセスされているか。ファイルは外部から読み取り可能か。

- ネットワークマウントファイルが読み取られている場合、不正に読み取られる可能性があります。外部から読み取れるファイルに鍵情報を保存して使用しないでください。
- ネーミングシステムを保護してください。次の 2 つの条件に該当する場合、そのホスト名は信頼できません。
  - ソースアドレスが、ネットワークを介して参照できるホストである。
  - ネーミングシステムの信頼性に問題がある。

セキュリティーの弱点の多くは、実際のツールではなく、ツールの使用方法にあります。`ipseckey` コマンドを使用するときには注意が必要です。各操作の最も安全なモードでコンソールを使用するか、ハード接続の TTY を使用してください。

その他のユーティリティーに対する IPsec 拡張機能

`ifconfig` コマンドには、トンネルインタフェースで IPsec ポリシーを管理するオプションがあります。`snoop` コマンドは、AH ヘッダーと ESP ヘッダーを構文解析できます。

`ifconfig` コマンドと IPsec

Solaris 10、**Solaris 10 7/05**、**Solaris 10 1/06**、および **Solaris 10 11/06** リリースの場合: IPsec をサポートするため、`ifconfig` では次のオプションを利用できます。Solaris 10 7/07 リリースでは、これらのセキュリティーオプションは `ipsecconf` コマンドで処理されます。

- `auth_algs`
- `encr_auth_algs`
- `encr_algs`

すべての IPsec セキュリティーオプションは 1 度の呼び出しでトンネルに指定する必要があります。たとえば、ESP だけを使ってトラフィックを保護する場合、次のように両方のセキュリティーオプションを指定して、トンネル `ip.tun0` を構成します。

```
# ifconfig ip.tun0 encr_algs aes encr_auth_algs md5
```

同様に、`ipsecinit.conf` エントリは、次のように 1 度に両方のセキュリティーオプションを指定して、トンネルを構成します。

```
# WAN traffic uses ESP with AES and MD5.
   {} ipsec {encr_algs aes encr_auth_algs md5}
```

auth_algsセキュリティーオプション

このオプションは、指定した認証アルゴリズムで、トンネルの IPsec AH を有効にできます。auth_algs オプションの書式は次のとおりです。

auth_algs *authentication-algorithm*

アルゴリズムには、番号またはアルゴリズム名を指定できます。特定のアルゴリズムが指定されないようにするパラメータ *any* も使用できます。トンネルセキュリティーを無効にするには、次のオプションを指定します。

auth_algs none

利用できる認証アルゴリズムのリストについては、ipsecalgs コマンドを実行してください。

注 – auth_algs オプションは NAT-Traversal と一緒には機能しません。詳細は、521 ページの「IPsec と NAT 越え」を参照してください。

---

encr_auth_algs セキュリティーオプション

このオプションは、指定した認証アルゴリズムで、トンネルの IPsec ESP を有効にできます。encr_auth_algs オプションの書式は次のとおりです。

encr_auth_algs *authentication-algorithm*

アルゴリズムには、番号またはアルゴリズム名を指定できます。特定のアルゴリズムが指定されないようにするパラメータ *any* も使用できます。ESP 暗号化アルゴリズムを指定し、認証アルゴリズムを指定しない場合、ESP 認証アルゴリズム値はデフォルトのパラメータ *any* になります。

利用できる認証アルゴリズムのリストについては、ipsecalgs コマンドを実行してください。

encr_algs セキュリティーオプション

このオプションは、指定した暗号化アルゴリズムで、トンネルの IPsec ESP を有効にできます。encr_algs オプションの書式は次のとおりです。

encr_algs *encryption-algorithm*

このアルゴリズムの場合、番号またはアルゴリズム名を指定できます。トンネルセキュリティーを無効にするには、次のオプションを指定します。

encr_algs none

ESP 認証アルゴリズムを指定し、暗号化アルゴリズムを指定しない場合、ESP 暗号化アルゴリズム値はデフォルトのパラメータ *null* になります。

利用できる暗号化アルゴリズムのリストについては、ipsecalgs コマンドを実行してください。

snoop コマンドと IPsec

snoop コマンドは、AH ヘッダーと ESP ヘッダーを構文解析できます。ESP はそのデータを暗号化するため、ESP で暗号化および保護されたヘッダーは snoop コマンドでは読み取ることができません。しかし、AH はデータを暗号化しません。したがって、AH で保護されたトラフィックは snoop コマンドで読み取ることができます。このコマンドに -V オプションを指定すると、いつ AH がパケットに使用されているかを表示できます。詳細は、snoop(1M) のマニュアルページを参照してください。

保護されたパケットに snoop コマンドを実行した場合の詳細な出力については、543 ページの「IPsec によってパケットが保護されていることを確認する方法」を参照してください。

◆ ◆ ◆ 第 22 章

22 インターネット鍵交換 (概要)

インターネット鍵交換 (IKE) は、IPsec の鍵管理を自動化します。Oracle Solaris は IKEv1 を実装します。IKE について説明するこの章の内容は次のとおりです。

IKE を実装する手順については、第 23 章「IKE の構成 (タスク)」を参照してください。参照情報については、第 24 章「インターネット鍵交換 (リファレンス)」を参照してください。IPsec については、第 19 章「IP セキュリティーアーキテクチャー (概要)」を参照してください。

IKE の新機能

Solaris 10 4/09: このリリース以降、サービス管理機能 (SMF) は IKE をサービスとして管理します。デフォルトでは、`svc:/network/ipsec/ike:default` サービスは無効になっています。また、このリリースでは、IPsec と IKE を管理するための Network IPsec Management 権利プロファイルが用意されています。

Solaris 10 7/07: このリリース以降では、IKE で AES アルゴリズムを使用でき、IKE を大域ゾーンで構成して非大域ゾーンで使用できます。

- `SO_ALLZONES` ソケットオプションを指定すると、IKE で非大域ゾーンのトラフィックを処理できるようになります。
- Oracle Solaris の新機能の完全な一覧や各 Solaris リリースの説明については、『Oracle Solaris 10 1/13 の新機能』を参照してください。

IKE による鍵管理

IPsec セキュリティーアソシエーション (SA) の鍵情報を管理することを「鍵管理」といいます。自動鍵管理では、鍵の作成、認証、および交換にセキュアな通信チャネルを要求します。Oracle Solaris ではインターネット鍵交換バージョン 1 (IKE) を使用して鍵管理を自動化します。IKE を使用すれば、セキュアなチャネルを大量のトラフィックに割り当てるために容易にスケーリングできます。IPv4 および IPv6 パケットの IPsec SA では、IKE の利点を生かすことができます。

IKE では、ハードウェアアクセラレーションとハードウェアストレージを利用できます。ハードウェアアクセラレータによって、負荷のかかる鍵操作をシステム外で処理できます。ハードウェア上での鍵の格納によって、保護機能が強化されます。

IKE の鍵ネゴシエーション

IKE デーモン `in.iked` は、安全な方法で IPsec SA のキーイング素材をネゴシエートし、認証します。デーモンは OS によって提供される内部機能から鍵用のランダムシードを使用します。IKE は、PFS (Perfect Forward Secrecy) をサポートしています。PFS では、データ伝送を保護する鍵を使用しないで追加鍵を取得します。また、データ伝送の鍵の作成に使用するシードを再利用しません。`in.iked(1M)` のマニュアルページを参照してください。

IKE の鍵用語について

次の表は、鍵ネゴシエーションで使用される用語と、一般的に使われるその略語、各用語の定義と使用についてまとめたものです。

表 22–1 鍵ネゴシエーションの用語、略語、使用

鍵ネゴシエーションの用語	略語	定義と使用
鍵交換		非対称暗号化アルゴリズムのキーを生成する処理。主な 2 つの方法は、RSA と Diffie-Hellman プロトコルです。
Diffie-Hellman アルゴリズム	DH	鍵生成と鍵認証を提供する鍵交換アルゴリズム。しばしば「認証された鍵交換」と呼ばれます。
RSA アルゴリズム	RSA	鍵生成と鍵転送を提供する鍵交換アルゴリズム。このプロトコル名は、作成者の Rivest、Shamir、Adleman の三氏に因んでいます。

表 22–1 鍵ネゴシエーションの用語、略語、使用 (続き)

鍵ネゴシエーションの用語	略語	定義と使用
Perfect Forward Secrecy	PFS	認証された鍵交換だけに適用されます。PFS では、データ伝送を保護するために使用される鍵が、追加の鍵を導き出すために使用されることはありません。さらに、データ伝送を保護するために使用される鍵のソースが、追加の鍵を導き出すために使用されることはありません。
Oakley グループ		フェーズ 2 の鍵を安全な方法で確立する 1 つの手法。Oakley グループは PFS のネゴシエーションに使用されます。The Internet Key Exchange (IKE) (`http://www.faqs.org/rfcs/rfc2409.html`) のセクション 6 を参照してください。

IKE フェーズ 1 交換

フェーズ 1 交換は、「メインモード」といわれているものです。フェーズ 1 交換では、IKE は公開鍵暗号方式を使用して、ピア IKE エンティティーと IKE 自体を認証します。その結果がインターネットセキュリティーアソシエーションと鍵管理プロトコル (ISAKMP) セキュリティーアソシエーション (SA) で、IKE で IP データグラムの鍵情報のネゴシエーションを行うためのセキュアなチャネルとなります。IPsec SA とは異なり、ISAKMP SA は双方向であるため、1 つの SA だけ必要です。

IKE がフェーズ 1 交換で鍵情報をネゴシエートする方法は構成可能です。IKE では、`/etc/inet/ike/config` ファイルから構成情報を読み取ります。次の構成情報があります。

- グローバルパラメータ (公開鍵証明書の名前など)
- PFS (Perfect Forward Secrecy) を使用する場合
- 影響を受けるインタフェース
- セキュリティープロトコルとそのアルゴリズム
- 認証方式

認証方式には、事前共有鍵と公開鍵証明書の 2 つがあります。公開鍵証明書は自己署名付きであっても、公開鍵インフラ (PKI) 組織の 認証局 (CA) が発行したものであってもかまいません。

IKE フェーズ 2 交換

フェーズ 2 交換は「クイックモード」といいます。フェーズ 2 交換では、IKE は IKE デーモンを実行するシステム間の IPsec SA を作成および管理します。また、フェーズ 1 交換で作成したセキュリティー保護されたチャネルを使用して、鍵情報の伝送を保護します。IKE デーモンは、`/dev/random` デバイスを使用して乱数発生関数からキーを作成します。また、IKE デーモンは、キーを一定の割合 (構成可能) でリフ

レッシュします。この鍵情報は、IPsec ポリシーの構成ファイル ipsecinit.conf に指定されているアルゴリズムによって使用されます。

IKE 構成の選択

/etc/inet/ike/config 構成ファイルには、IKE ポリシーのエントリが含まれています。2 つの IKE デーモンを相互に認証するためには、これらのエントリが有効でなければなりません。さらに、鍵情報も必要です。構成ファイルのエントリは、フェーズ 1 交換を認証するための鍵情報の使用方法を決定します。選択肢は、事前共有鍵か公開鍵証明書のどちらかです。

エントリ auth_method preshared は、事前共有鍵が使用されることを示します。auth_method の値が preshared 以外の場合には、公開鍵証明書が使用されることを示します。公開鍵証明書は自己署名付きにするか、PKI 組織から発行できます。詳細は、ike.config(4) のマニュアルページを参照してください。

IKE と事前共有鍵認証

事前共有鍵は、複数のピアシステムを認証するために使用されます。事前共有鍵は、1 つのシステム上の管理者によって作成される 16 進数または ASCII 文字列です。この鍵はその後、ピアシステムの管理者によって帯域外で共有されます。事前共有鍵が傍受者によって傍受されると、その傍受者はピアシステムの 1 つを偽装できる可能性があります。

この認証方法を使用するピア上の事前共有鍵は、同一である必要があります。これらの鍵は、特定の IP アドレスに関連付けられています。鍵は、各システムの /etc/inet/secret/ike.preshared ファイルに保存されます。IPsec の場合は ipseckeys ファイルですが、IKE の場合は ike.preshared ファイルとなります。ike.preshared ファイルにある鍵に何らかの問題があると、すべての伝送に問題が発生します。あるシステムの管理者が通信先のシステムを制御する場合、これらの鍵は最も安全です。詳細は、ike.preshared(4) のマニュアルページを参照してください。

IKE と公開鍵証明書

公開鍵証明書を使用すると、通信するシステムが秘密鍵情報を帯域外で共有する必要がなくなります。公開鍵では、鍵の認証とネゴシエーションに Diffie-Hellman アルゴリズム (DH) を使用します。公開鍵証明書には、2 つの方法があります。公開鍵証明書は、自己署名付きにすることも、認証局 (CA) が認証することもできます。

自己署名付き公開鍵証明書は、自ら (管理者) が作成します。ikecert certlocal -ks コマンドを実行して、システムの公開鍵と非公開鍵のペアの非公開部分を作成します。そのあと、管理者は、リモートシステムから X.509 形式で自己署名付き証明書の

出力を取得します。リモートシステムの証明書は、鍵のペアの公開部分の ikecert certdb コマンドに入力されます。自己署名付き証明書は、通信するシステムの /etc/inet/ike/publickeys ディレクトリに保存されます。証明書をシステムに接続されているハードウェアに保存したい場合は、-T オプションを指定します。

自己署名付き証明書は、事前共有鍵と CA 間の中間ポイントになります。事前共有鍵とは異なり、自己署名付き証明書は移動体システムまたは再番号付け可能なシステムで使用できます。固定番号を使用しないで、システムの証明書に自己署名するには、DNS (www.example.org) または email (root@domain.org) の代替名を使用します。

公開鍵は、PKI または CA 組織で配信できます。公開鍵とそれに関連する CA は、/etc/inet/ike/publickeys ディレクトリに格納されます。証明書をシステムに接続されているハードウェアに保存したい場合は、-T オプションを指定します。また、ベンダーは証明書失効リスト (CRL) も発行します。管理者は鍵と CA を格納するだけでなく、CRL を /etc/inet/ike/crls ディレクトリに格納する責任があります。

CA には、サイトの管理者ではなく、外部の機関によって認証されるといった特長があります。その点では、CA は公証された証明書となります。自己署名付き証明書と同様に、CA は移動体システムまたは再番号付け可能なシステムで使用できます。その一方、自己署名付き証明書とは異なり、CA は通信する多くのシステムを保護するために容易にスケーリングできます。

IKE とアクセラレータハードウェア

IKE アルゴリズムは、特にフェーズ 1 交換において大量の処理を必要とします。大量の交換処理を行うシステムでは、Sun Crypto Accelerator 1000 または Sun Crypto Accelerator 6000 ボードを使って公開鍵の操作を処理できます。フェーズ 1 の大量の処理には、Sun Crypto Accelerator 6000 ボードおよび Sun Crypto Accelerator 4000 ボードを使用することもできます。

IKE の負荷を高速化ボードに移すために IKE をどのように構成するかについては、642 ページの「Sun Crypto Accelerator 4000 ボードを検出するように IKE を構成する方法」を参照してください。キーを格納する方法については、642 ページの「Sun Crypto Accelerator 4000 ボードを検出するように IKE を構成する方法」と cryptoadm(1M) のマニュアルページを参照してください。

IKE とハードウェアストレージ

<公開鍵証明書、非公開鍵、公開鍵は、Sun Crypto Accelerator 6000 または Sun Crypto Accelerator 4000 ボードに格納できます。RSA 暗号化の場合、これらのボードは最大 2048 ビットの鍵をサポートします。DSA 暗号化の場合、ボードは最大で 1024 ビットをサポートします。Sun Crypto Accelerator 6000 ボードは、SHA-512 および ECC アルゴリズムをサポートします。

ボードにアクセスするように IKE を構成する方法については、642 ページの「Sun Crypto Accelerator 4000 ボードを検出するように IKE を構成する方法」を参照してください。ボードに証明書や公開鍵を追加する方法については、627 ページの「ハードウェアで公開鍵証明書を生成および格納する方法」を参照してください。

IKE ユーティリティーおよび IKE ファイル

次の表は、IKE ポリシーの構成ファイルや、IKE キーの格納場所、IKE を実装する各種のコマンドとサービスについてまとめたものです。サービスについては、『Oracle Solaris の管理: 基本管理』の第 18 章「サービスの管理 (概要)」を参照してください。

表 22–2 IKE 構成ファイル、鍵の格納場所、コマンド、サービス

ファイル、場所、コマンド、またはサービス	説明	マニュアルページ
svc:/network/ipsec/ike	現在のリリースでは、IKE を管理する SMF サービス。	smf(5)
/usr/lib/inet/in.iked	インターネット鍵交換 (IKE) デーモン。自動鍵管理を有効にします。現在のリリースでは、ike サービスがこのデーモンを有効にします。これより前のリリースでは、in.iked コマンドを使用します。	in.iked(1M)
/usr/sbin/ikeadm	IKE ポリシーの表示および変更用 IKE 管理コマンド。	ikeadm(1M)
/usr/sbin/ikecert	公開鍵証明書が格納されているローカルデータベースを操作する証明書データベース管理コマンド。データベースは、接続されたハードウェアにも格納できます。	ikecert(1M)
/etc/inet/ike/config	デフォルトの IKE ポリシー構成ファイル。インバウンド IKE 要求のマッチングとアウトバウンド IKE 要求の準備に関するサイトの規則が含まれています。 現在のリリースでは、このファイルが存在する場合、ike サービスが有効になると in.iked デーモンが起動します。このファイルの場所は svccfg コマンドで変更することができます。	ike.config(4)
ike.preshared	/etc/inet/secret ディレクトリにある事前共有鍵ファイル。フェーズ 1 交換での認証の秘密鍵情報が含まれます。事前共有鍵を使って IKE を構成するときに使用されます。	ike.preshared(4)
ike.privatekeys	/etc/inet/secret ディレクトリにある非公開鍵ディレクトリ。公開鍵と非公開鍵のペアの非公開部分が含まれています。	ikecert(1M)
publickeys ディレクトリ	/etc/inet/ike ディレクトリ内のディレクトリ。公開鍵と証明書ファイルが格納されています。公開鍵と非公開鍵のペアの公開部分が含まれています。	ikecert(1M)

表 22–2 IKE 構成ファイル、鍵の格納場所、コマンド、サービス (続き)

ファイル、場所、コマンド、またはサービス	説明	マニュアルページ
crls ディレクトリ	/etc/inet/ike ディレクトリ内のディレクトリ。公開鍵や証明書ファイルの失効リストが格納されています。	ikecert(1M)
Sun Crypto Accelerator 1000 ボード	オペレーティングシステムの処理を少なくすることで公開鍵操作を高速化するハードウェア。	ikecert(1M)
Sun Crypto Accelerator 4000 ボード	オペレーティングシステムの処理を少なくすることで公開鍵操作を高速化するハードウェア。公開鍵、非公開鍵、および公開鍵証明書も格納します。Sun Crypto Accelerator 6000 ボードはレベル 3 の FIPS 140-2 認定デバイスです。	ikecert(1M)

Oracle Solaris 10 リリースにおける IKE の変更

Solaris 9 リリース以降、IKE に次の機能が追加されています。

- IKE を使用すれば、IPv6 ネットワークでの IPsec のキー交換を自動化できます。詳細は、596 ページの「IKE による鍵管理」を参照してください。

  注 – IKE を使って、非大域ゾーンでの IPsec のキーを管理することはできません。

- IKE での公開鍵操作を Sun Crypto Accelerator 1000 ボードや Sun Crypto Accelerator 4000 ボードで高速化できます。処理がボードで実行されます。このため、暗号化処理が高速化され、オペレーティング環境の消費リソースも少なくて済みます。詳細は、599 ページの「IKE とアクセラレータハードウェア」を参照してください。手順については、641 ページの「接続したハードウェアを検出するように IKE を構成する」を参照してください。
- 公開鍵証明書や、非公開鍵、公開鍵を Sun Crypto Accelerator 4000 ボードに格納できます。キーのストレージについては、599 ページの「IKE とハードウェアストレージ」を参照してください。
- IKE を使用すれば、NAT ボックスの後ろから IPsec のキー交換を自動化できます。ただし、NAT を横断する IPsec ESP キーをハードウェアで高速化することはできません。詳細は、521 ページの「IPsec と NAT 越え」を参照してください。手順については、633 ページの「移動体システム用の IKE の構成 (タスクマップ)」を参照してください。
- 再伝送パラメータとパケットタイムアウトパラメータが /etc/inet/ike/config ファイルに追加されています。これらのパラメータは、IKE フェーズ 1 (メインモード) ネゴシエーションを調整することによって、ネットワークの干渉や、重いネットワークトラフィック、さらに、IKE プロトコルの実装方法が異なるプラットフォームとの相互操作などを処理するためのものです。パラメータの詳細

は、ike.config(4) のマニュアルページを参照してください。手順については、645 ページの「IKE 転送パラメータの変更 (タスクマップ)」を参照してください。

◆ ◆ ◆ 第 23 章

IKE の構成 (タスク)

この章では、使用するシステムにあわせて Internet Key Exchange (IKE) を構成する方法について説明します。IKE の構成が完了すると、そのネットワークにおける IPsec の鍵情報が自動的に生成されます。

この章では、次の内容について説明します。

IKE の概要については、第 22 章「インターネット鍵交換 (概要)」を参照してください。IKE の参照情報については、第 24 章「インターネット鍵交換 (リファレンス)」を参照してください。詳細な手順については、`ikeadm`(1M)、`ikecert`(1M)、および `ike.config`(4) のマニュアルページで使用例のセクションを参照してください。

IKE の構成 (タスクマップ)

IKE を認証するには、事前共有鍵、自己署名付き証明書、および認証局 (CA) の証明書を使用できます。規則として、保護しようとしているエンドポイントには、特定の IKE 認証方法を関連付けます。したがって、1 つのシステムに 1 つまたはすべての IKE 認証方法を使用できます。PKCS #11 ライブラリへのポインタによって、証明書は、接続されたハードウェアアクセラレータを使用できます。

IKE を構成したあと、IKE 構成を使用する IPsec タスクを実行します。次の表に、特定の IKE 構成に注目したタスクマップを示します。

タスク	説明	参照先
事前共有鍵で IKE を構成します	秘密鍵を共有するシステム間での通信を保護します。	604 ページの「事前共有鍵による IKE の構成 (タスクマップ)」
公開鍵証明書で IKE の構成します	公開鍵証明書を使って通信を保護します。証明書は、自己署名付き、または PKI 機関の保証付きです。	615 ページの「公開鍵証明書による IKE の構成 (タスクマップ)」
NAT 境界を越えます	IPsec と IKE を構成して、移動体システムと通信します。	633 ページの「移動体システム用の IKE の構成 (タスクマップ)」
公開鍵証明書を生成し、接続されたハードウェアに格納するように構成します	Sun Crypto Accelerator 1000 ボードまたは Sun Crypto Accelerator 4000 ボードを使って IKE 操作を高速化します。Sun Crypto Accelerator 4000 ボードには、公開鍵証明書も格納できます。	641 ページの「接続したハードウェアを検出するように IKE を構成する」
フェーズ 1 鍵ネゴシエーションパラメータを調整します	IKE 鍵ネゴシエーションのタイミングを変更します。	645 ページの「IKE 転送パラメータの変更 (タスクマップ)」

事前共有鍵による IKE の構成 (タスクマップ)

次の表に、事前共有鍵で IKE を構成および保守する手順を示します。

タスク	説明	参照先
事前共有鍵で IKE を構成します	共有する IKE ポリシーファイルと 1 つの鍵を作成します。	605 ページの「事前共有鍵により IKE を構成する方法」
実行中の IKE システムで事前共有鍵をリフレッシュします	通信するシステムに最新の鍵情報を追加します。	608 ページの「IKE の事前共有鍵をリフレッシュする方法」
実行中の IKE システムへ事前共有鍵を追加します	現在 IKE ポリシーを実施しているシステムに、新しい IKE ポリシーエントリと新しい鍵情報を追加します。	611 ページの「`ipsecinit.conf` の新しいポリシーエントリ用に IKE 事前共有鍵を追加する方法」
事前共有鍵が同一であることをチェックします	両方のシステムの事前共有鍵を表示して、その鍵が同一であることを調べます。	613 ページの「事前共有鍵が同一であることを確認する方法」

事前共有鍵による IKE の構成

事前共有鍵は、IKE 用のもっとも簡単な認証方法です。2 つのシステムが IKE を使用するように構成している場合、さらに、両方のシステムの管理者である場合、事前共有鍵を使用することはよい選択です。しかし、公開鍵認証とは異なり、事前共有鍵は特定の IP アドレスに縛られます。事前共有鍵は、移動体システムなど、番号が変更される可能性があるシステムでは使用できません。

▼ 事前共有鍵により IKE を構成する方法

IKE 実装では、鍵の長さが異なるさまざまなアルゴリズムが提供されます。鍵の長さは、サイトのセキュリティーに応じて選択します。一般的に、鍵の長さが長いほど、セキュリティーが高くなります。

これらの手順には、システム名 `enigma` および `partym` を使用します。`enigma` と `partym` を各自使用しているシステムの名前に置き換えてください。

1. システムコンソール上で、**Primary Administrator** の役割を引き受けるか、スーパーユーザーになります。

   Primary Administrator 役割には、Primary Administrator プロファイルが含まれます。役割を作成してユーザーに役割を割り当てるには、『Oracle Solaris の管理: 基本管理』の第 2 章「Solaris 管理コンソールの操作 (タスク)」を参照してください。

   注 – リモートログインすると、セキュリティー上重要なトラフィックが盗聴される恐れがあります。何らかの方法でリモートログインを保護していても、システムのセキュリティーがリモートログインセッションレベルに低下します。セキュリティー保護されたリモートログインには、`ssh` コマンドを使用してください。

   ---

2. システムごとに、**/etc/inet/ike/config.sample** ファイルを **/etc/inet/ike/config** にコピーします。

3. システムごとに、規則とグローバルパラメータを **ike/config** ファイルに入力します。

   これらの規則やグローバルパラメータは、システムの `ipsecinit.conf` ファイルに設定されている IPsec ポリシーが正しく動作するものでなければなりません。次の `ike/config` の例は、529 ページの「IPsec で 2 つのシステム間のトラフィックを保護するには」の `ipsecinit.conf` の例で機能します。

   a. たとえば、**enigma** システムの **/etc/inet/ike/config** ファイルを次のように変更します。

   ```
   ### ike/config file on enigma, 192.168.116.16

   ## Global parameters
   ```

```
#
## Phase 1 transform defaults
p1_lifetime_secs 14400
p1_nonce_len 40
#
## Defaults that individual rules can override.
p1_xform
  { auth_method preshared oakley_group 5 auth_alg sha encr_alg 3des }
p2_pfs 2
#
## The rule to communicate with partym
#  Label must be unique
{ label "enigma-partym"
  local_addr 192.168.116.16
  remote_addr 192.168.13.213
  p1_xform
    { auth_method preshared oakley_group 5 auth_alg sha1 encr_alg aes }
  p2_pfs 5
}
```

b. **partym** システムの **/etc/inet/ike/config** ファイルを次のように変更します。

```
### ike/config file on partym, 192.168.13.213
## Global Parameters
#
p1_lifetime_secs 14400
p1_nonce_len 40
#
p1_xform
  { auth_method preshared oakley_group 5 auth_alg sha encr_alg 3des }
p2_pfs 2

## The rule to communicate with enigma
#  Label must be unique
{ label "partym-enigma"
  local_addr 192.168.13.213
  remote_addr 192.168.116.16
p1_xform
  { auth_method preshared oakley_group 5 auth_alg sha1 encr_alg aes }
p2_pfs 5
}
```

4 システムごとに、ファイルの構文を確認します。

```
# /usr/lib/inet/in.iked -c -f /etc/inet/ike/config
```

5 鍵情報として使用する乱数を生成します。

乱数発生関数がすでにある場合は、それを使用してください。Oracle Solaris 10 システムでは、od コマンドを使用できます。たとえば、次のコマンドを入力すると、16 進数の数値が 2 行に渡って表示されます。

```
% od -X -A n /dev/random | head -2
         f47cb0f4 32e14480 951095f8 2b735ba8
         0a9467d0 8f92c880 68b6a40e 0efe067d
```

od コマンドについては、537 ページの「Oracle Solaris System で乱数を生成する方法」と od(1) のマニュアルページを参照してください。

注 – ほかのオペレーティングシステムでは、ASCII 形式の鍵情報が必要になる場合があります。同じ鍵を 16 進形式と ASCII 形式で生成する方法については、例 23–1 を参照してください。

6 手順 **5** の出力から、**1** つの鍵を作成します。

```
f47cb0f432e14480951095f82b735ba80a9467d08f92c88068b6a40e
```

この手順における認証アルゴリズムは SHA–1 です (手順 3 を参照)。事前共有鍵として推奨する最小のサイズは、ハッシュのサイズ (つまり、認証アルゴリズムの出力のサイズ) で決まります。SHA–1 アルゴリズムの出力は 160 ビット、すなわち 40 文字です。例の鍵の長さは 56 文字であり、IKE が使用する鍵情報が追加されています。

7 システムごとに **/etc/inet/secret/ike.preshared** ファイルを作成します。

各ファイルに事前共有鍵を書き込みます。

a. たとえば、**enigma** システムの **ike.preshared** ファイルは次のようになります。

```
# ike.preshared on enigma, 192.168.116.16
#...
{ localidtype IP
    localid 192.168.116.16
    remoteidtype IP
    remoteid 192.168.13.213
    # enigma and partym's shared key in hex (192 bits)
    key f47cb0f432e14480951095f82b735ba80a9467d08f92c88068b6a40e
    }
```

b. **partym** システムの **ike.preshared** ファイルは次のようになります。

```
# ike.preshared on partym, 192.168.13.213
#...
{ localidtype IP
    localid 192.168.13.213
    remoteidtype IP
    remoteid 192.168.116.16
    # partym and enigma's shared key in hex (192 bits)
    key f47cb0f432e14480951095f82b735ba80a9467d08f92c88068b6a40e
    }
```

注 – 両システムの事前共有鍵は同一にする必要があります。

例 23–1 オペレーティングシステムの異なる 2 つのシステムに対して同じ鍵情報を生成する

Oracle Solaris の IPsec 機能は、ほかのオペレーティングシステムと相互運用できます。ASCII 形式の事前共有鍵を必要とするシステムと通信する場合は、1 つの鍵を 16 進形式と ASCII 形式の 2 つの形式で生成する必要があります。

この例では、Oracle Solaris システムの管理者が 56 文字の鍵情報を使用します。管理者は、次のコマンドを使用して、ASCII パスフレーズから 16 進形式の鍵を生成します。オプション -tx1 を指定すると、一度に 1 バイトずつ、すべての Oracle Solaris システムに出力されます。

```
# /bin/echo "papiermache with cashews and\c" | od -tx1 | cut -c 8-55 | \
tr -d '\n' | tr -d ' ' | awk '{print}'
7061706965726d6163686520776974682063617368657777320616e64
```

オフセットを削除して 16 進出力を連結すると、Oracle Solaris システム用の 16 進形式の鍵は 7061706965726d6163686520776974682063617368657777320616e64 になります。管理者は、この値を Oracle Solaris システムの ike.preshared ファイルに格納します。

```
# Shared key in hex (192 bits)
key 7061706965726d6163686520776974682063617368657777320616e64
```

ASCII 形式の事前共有鍵を必要とするシステムでは、パスフレーズが事前共有鍵になります。Oracle Solaris システムの管理者は、相手の管理者に電話し、パスフレーズ papiermache with cashews and を伝えます。

▼ IKE の事前共有鍵をリフレッシュする方法

この手順では、一定の間隔で既存の事前共有鍵を置き換えたい場合を想定しています。

1 システムコンソール上で、**Primary Administrator** の役割を引き受けるか、スーパーユーザーになります。

Primary Administrator 役割には、Primary Administrator プロファイルが含まれます。役割を作成してユーザーに役割を割り当てるには、『Oracle Solaris の管理: 基本管理』の第 2 章「Solaris 管理コンソールの操作 (タスク)」を参照してください。

注 – リモートログインすると、セキュリティー上重要なトラフィックが盗聴される恐れがあります。何らかの方法でリモートログインを保護していても、システムのセキュリティーがリモートログインセッションレベルに低下します。セキュリティー保護されたリモートログインには、ssh コマンドを使用してください。

---

2 乱数を生成し、適切な長さのキーを作成します。

詳細については、537 ページの「Oracle Solaris System で乱数を生成する方法」を参照してください。Oracle Solaris システムが ASCII 形式を必要とするオペレーティングシステムと通信する場合: 事前共有鍵を生成する方法については、例 23–1 を参照してください。

3. 現在の鍵を新しい鍵で置き換えます。
   たとえば、ホスト enigma と partym において、 /etc/inet/secret/ike.preshared ファイルの key の値を、同じ長さの新しい番号で置き換えます。

4. 新しい鍵をカーネルに読み込みます。

   - **Solaris 10 4/09** リリース以降では、**ike** サービスを再起動します。

     ```
     # svcadm enable ike
     ```

   - **Solaris 10 4/09** リリースより前のリリースを実行している場合は、**in.iked** デーモンを強制終了および再起動します。

     a. **in.iked** デーモンの特権レベルをチェックします。

        ```
        # /usr/sbin/ikeadm get priv
        Current privilege level is 0x0, base privileges enabled
        ```

        コマンドから 0x1 または 0x2 の特権レベルが戻された場合には、鍵情報を変更できます。0x0 の特権レベルでは、鍵情報を変更または表示する操作を行うことはできません。デフォルトでは、in.iked デーモンは 0x0 の特権レベルで実行されます。

     b. 特権レベルが **0x0** の場合、デーモンを強制終了および再起動します。
        デーモンを再起動すると、ike.preshared ファイルの新しいバージョンを読み取ります。

        ```
        # pkill in.iked
        # /usr/lib/inet/in.iked
        ```

     c. 特権レベルが **0x1** または **0x2** である場合、**ike.preshared** ファイルの新しいバージョンを読み取ります。

        ```
        # ikeadm read preshared
        ```

▼ IKE の事前共有鍵を表示する方法

デフォルトでは、ikeadm コマンドではフェーズ 1 SA のダンプに実際の鍵を表示できないようになっています。鍵を表示するとデバッグに役立ちます。

実際の鍵を表示するには、デーモンの特権レベルを高くする必要があります。特権レベルについては、651 ページの「ikeadm コマンド」を参照してください。

注 – Solaris 10 4/09 リリースより前のリリースでこの手順を実行するには、例 23–2 を参照してください。

始める前に　IKE は構成済みで、ike サービスは実行中です。

1 **IKE** の事前共有鍵を表示します。

```
# ikeadm
ikeadm> dump preshared
```

2 エラーが発生する場合は、**in.iked** デーモンの特権レベルを高くします。

a. **SMF** リポジトリの **in.iked** デーモンの特権レベルを高くします。

```
# svcprop -p config/admin_privilege ike
base
# svccfg -s ike setprop config/admin_privilege=keymat
```

b. 実行中の **in.iked** デーモンの特権レベルを高くします。

```
# svcadm refresh ike ; svcadm restart ike
```

c. (省略可能) 特権レベルが **keymat** であることを確認します。

```
# svcprop -p config/admin_privilege ike
keymat
```

d. 手順 **1** をもう一度実行して鍵を表示します。

3 **IKE** デーモンを基本の特権レベルに戻します。

a. 鍵を表示したあと、特権レベルをデフォルトに戻します。

```
# svccfg -s ike setprop config/admin_privilege=base
```

b. **IKE** をリフレッシュしてから再起動します。

```
# svcadm refresh ike ; svcadm restart ike
```

例 23–2 Solaris 10 4/09 リリースより前のリリースで IKE の事前共有鍵を確認する

次の例では、現在の Oracle Solaris 10 リリースが稼働していない Solaris システムで管理者が鍵を表示しています。管理者は、このシステムの鍵が通信先のシステムの鍵と同じであることを確認する必要があります。2 つのシステムの鍵が同じであることを確認したあと、管理者は特権レベルを 0 に戻します。

- まず、管理者は `in.iked` デーモンの特権レベルを調べます。

```
adm1 # /usr/sbin/ikeadm get priv
Current privilege level is 0x0, base privileges enabled
```

- 特権レベルが 0x1 または 0x2 になっていないため、管理者は `in.iked` デーモンを停止してから特権レベルを 2 に上げます。

```
adm1 # pkill in.iked
adm1 # /usr/lib/inet/in.iked -p 2
Setting privilege level to 2
```

- 管理者は鍵を表示します。

```
adm1 # ikeadm dump preshared
PSKEY: Preshared key (24 bytes): f47cb.../192
LOCIP: AF_INET: port 0, 192.168.116.16 (adm1).
REMIP: AF_INET: port 0, 192.168.13.213 (com1).
```

- 管理者は通信先のシステムにリモートでログインし、鍵が同じかどうかを調べます。
- その後、基本の特権レベルに戻します。

```
# ikeadm set priv base
```

▼ ipsecinit.conf の新しいポリシーエントリ用に IKE 事前共有鍵を追加する方法

同じピア間で動作中の構成に対して IPsec ポリシーエントリを追加した場合、IPsec ポリシーサービスをリフレッシュする必要があります。IKE の再構成または再起動は不要です。

IPsec ポリシーに新しいピアを追加した場合、IPsec の変更に加えて IKE 構成を変更する必要があります。

注 – Solaris 10 4/09 リリースより前のリリースでこの手順を実行するには、例 23–3 を参照してください。

始める前に　`ipsecinit.conf` ファイルをリフレッシュし、ピアシステムの IPsec ポリシーをリフレッシュしました。

1. システムコンソール上で、**Primary Administrator** の役割を引き受けるか、スーパーユーザーになります。

   Primary Administrator 役割には、Primary Administrator プロファイルが含まれます。役割を作成してユーザーに役割を割り当てるには、『Oracle Solaris の管理: 基本管理』の第 2 章「Solaris 管理コンソールの操作 (タスク)」を参照してください。

   注 – リモートログインすると、セキュリティー上重要なトラフィックが盗聴される恐れがあります。何らかの方法でリモートログインを保護していても、システムのセキュリティーがリモートログインセッションレベルに低下します。セキュリティー保護されたリモートログインには、`ssh` コマンドを使用してください。

   ---

2. このシステムで乱数を生成し、**64** から **448** ビットの鍵を作成します。

   詳細については、537 ページの「Oracle Solaris System で乱数を生成する方法」を参照してください。Oracle Solaris システムが ASCII 形式を必要とするオペレーティングシステムと通信する場合: 事前共有鍵を生成する方法については、例 23–1 を参照してください。

3 このキーを何らかの方法でリモートシステムの管理者に送信します。

両者は、同じ事前共有鍵を同時に追加する必要があります。この鍵の安全性は転送メカニズムの安全性と同じです。登録済みメールや保護済み FAX マシンなど、帯域外メカニズムを使用することが最良です。ssh セッションを使用して両方のシステムを管理することもできます。

4 **enigma** と新しいピア **ada** の鍵を管理するための **IKE** の規則を作成します。

a. **enigma** システムで、次の規則を **/etc/inet/ike/config** ファイルに追加します。

```
### ike/config file on enigma, 192.168.116.16

## The rule to communicate with ada

{label "enigma-to-ada"
 local_addr 192.168.116.16
 remote_addr 192.168.15.7
 p1_xform
 {auth_method preshared oakley_group 5 auth_alg sha1 encr_alg blowfish}
 p2_pfs 5
    }
```

b. **ada** システムで、次の規則を追加します。

```
### ike/config file on ada, 192.168.15.7

## The rule to communicate with enigma

{label "ada-to-enigma"
 local_addr 192.168.15.7
 remote_addr 192.168.116.16
 p1_xform
 {auth_method preshared oakley_group 5 auth_alg sha1 encr_alg blowfish}
 p2_pfs 5
}
```

5 リブート時に **IKE** 事前共有鍵が利用できることを確認します。

a. **enigma** システムで、次の情報を **/etc/inet/secret/ike.preshared** ファイルに追加します。

```
# ike.preshared on enigma for the ada interface
#
{ localidtype IP
  localid 192.168.116.16
  remoteidtype IP
  remoteid 192.168.15.7
  # enigma and ada's shared key in hex (32 - 448 bits required)
  key 8d1fb4ee500e2bea071deb2e781cb48374411af5a9671714672bb1749ad9364d
}
```

b. **ada** システムで、次の情報を **ike.preshared** ファイルに追加します。

```
# ike.preshared on ada for the enigma interface
#
{ localidtype IP
```

```
    localid 192.168.15.7
    remoteidtype IP
    remoteid 192.168.116.16
    # ada and enigma's shared key in hex (32 - 448 bits required)
    key 8d1fb4ee500e2bea071deb2e781cb48374411af5a9671714672bb1749ad9364d
  }
```

6 各システムで、**ike** サービスをリフレッシュします。

```
# svcadm refresh ike
```

7 両システムが通信できることを確認します。

詳細は、613 ページの「事前共有鍵が同一であることを確認する方法」を参照してください。

例 23–3 新しい IPsec ポリシーエントリに IKE 事前共有鍵を追加する

次の例では、現在の Oracle Solaris 10 リリースが稼働していない Solaris システムに管理者が事前共有鍵を追加しています。管理者は前の手順に従って `ike/config` ファイルと `ike.preshared` ファイルを変更し、鍵を生成し、リモートシステムに接続します。

- 新しい鍵を生成する前に、管理者は `in.iked` デーモンの特権レベルを 2 に設定します。

```
# pkill in.iked
# /usr/lib/inet/in.iked -p 2
Setting privilege level to 2
```

- 通信先のシステムに鍵を送信し、新しい鍵をシステムに追加したあと、管理者は特権レベルを低くします。

```
# ikeadm set priv base
```

- 最後に、新しい IKE ルールをカーネルに読み込みます。

```
# ikeadm read rules
```

次の手順 IPsec ポリシーの設定がまだ完了していない場合、IPsec の手順に戻って IPsec ポリシーを有効にするかリフレッシュしてください。

▼ 事前共有鍵が同一であることを確認する方法

通信するシステム上の事前共有鍵が同一でない場合、それらのシステムは認証できません。

始める前に テストしている 2 つのシステム間では IPsec が構成されており、有効になっています。現在の Oracle Solaris 10 リリースが稼働しています。

注 – Solaris 10 4/09 リリースより前のリリースでこの手順を実行するには、例 23–2 を参照してください。

1 システムコンソール上で、**Primary Administrator** の役割を引き受けるか、スーパーユーザーになります。

Primary Administrator 役割には、Primary Administrator プロファイルが含まれます。役割を作成してユーザーに役割を割り当てるには、『Oracle Solaris の管理: 基本管理』の第 2 章「Solaris 管理コンソールの操作 (タスク)」を参照してください。

注 – リモートログインすると、セキュリティー上重要なトラフィックが盗聴される恐れがあります。何らかの方法でリモートログインを保護していても、システムのセキュリティーがリモートログインセッションレベルに低下します。セキュリティー保護されたリモートログインには、`ssh` コマンドを使用してください。

2 各システムで、**in.iked** デーモンの特権レベルをチェックします。

```
# svcprop -p config/admin_privilege ike
base
```

- 特権レベルが **keymat** であれば、手順 **3** で続けます。
- 特権レベルが **base** または **modkeys** の場合は、特権レベルを高くします。

  その後、`ike` サービスをリフレッシュしてから再起動します。

```
# svccfg -s ike setprop config/admin_privilege=keymat
# svcadm refresh ike ; svcadm restart ike
# svcprop -p config/admin_privilege ike
keymat
```

3 システムごとに、事前共有鍵情報を表示します。

```
# ikeadm dump preshared
PSKEY: Preshared key (24 bytes): f47cb.../192
LOCIP: AF_INET: port 0, 192.168.116.16 (enigma).
REMIP: AF_INET: port 0, 192.168.13.213 (partym).
```

4 両方のダンプを比較します。

事前共有鍵が同一でない場合は、`/etc/inet/secret/ike.preshared` ファイルで、一方のキーを他方のキーで置き換えます。

5 確認が終わったら、各システム上で特権レベルをデフォルトに戻します。

```
# svccfg -s ike setprop config/admin_privilege=base
# svcadm restart ike
```

公開鍵証明書による IKE の構成 (タスクマップ)

次の表に、IKE の公開鍵証明書を作成する手順を示します。これらの手順には、接続されたハードウェア上で証明書を高速化および格納する方法が含まれます。

公開証明書は一意である必要があるため、公開鍵証明書の作成者は証明書について一意となる任意の名前を生成します。通常は、X.509 識別名が使用されます。識別用の代替名も使用できます。これらの名前の形式は、*tag=value* です。値は任意ですが、値の形式は、そのタグの種類に対応する必要があります。たとえば、`email` タグの形式は *name@domain.suffix* です。

タスク	説明	参照先
自己署名付き公開鍵証明書で IKE を構成します	システムごとに 2 つの証明書を作成および格納します。 ■ 自己署名付き証明書 ■ リモートシステムからの公開鍵証明書	616 ページの「自己署名付き公開鍵証明書により IKE を構成する方法」
PKI 認証局で IKE を構成します	1 つの証明書要求を作成して、そのあと、システムごとに次の 3 つの証明書を格納します。 ■ 証明書要求に応じて認証局 (CA) が作成した証明書 ■ CA からの公開鍵証明書 ■ CA からの CRL	621 ページの「CA からの署名付き証明書により IKE を構成する方法」
ローカルハードウェア上で公開鍵証明書を構成します	次のいずれかが含まれます。 ■ ローカルハードウェア上で自己署名付き証明書を生成し、リモートシステムの公開鍵をハードウェアに追加する ■ ローカルハードウェア上で証明書リクエストを生成し、CA から取得した公開鍵証明書をハードウェアに追加する	627 ページの「ハードウェアで公開鍵証明書を生成および格納する方法」
PKI からの証明書失効リスト CRL を更新します	中央の配布ポイントから CRL にアクセスします。	631 ページの「証明書失効リストを処理する方法」

公開鍵証明書による IKE の構成

公開鍵証明書を使用すると、通信するシステムは秘密鍵情報を帯域外で共有する必要がなくなります。事前共有鍵とは異なり、公開鍵証明書は、移動体システムなど、番号が変更される可能性があるシステムでも使用できます。

公開鍵証明書はまた、接続されたハードウェアに格納できます。手順については、641 ページの「接続したハードウェアを検出するように IKE を構成する」を参照してください。

▼ 自己署名付き公開鍵証明書により IKE を構成する方法

この手順では、証明書ペアを作成します。非公開鍵はディスク上のローカル証明書データベースに格納され、`certlocal` サブコマンドを使用して参照できます。証明書ペアの公開部分は、公開証明書データベースに格納されます。これは `certdb` サブコマンドを使用して参照できます。公開部分をピアシステムと交換します。2 つの証明書を組み合わせたものが、IKE 転送を認証するために使用されます。

自己署名付き証明書は、CA からの公開鍵証明書よりもオーバーヘッドが少ないのですが、あまり簡単には拡大できません。CA によって発行される証明書とは異なり、自己署名付き証明書は帯域外で検証する必要があります。

1 システムコンソール上で、**Primary Administrator** の役割を引き受けるか、スーパーユーザーになります。

Primary Administrator 役割には、Primary Administrator プロファイルが含まれます。役割を作成してユーザーに役割を割り当てるには、『Oracle Solaris の管理: 基本管理』の第 2 章「Solaris 管理コンソールの操作 (タスク)」を参照してください。

注 – リモートログインすると、セキュリティー上重要なトラフィックが盗聴される恐れがあります。何らかの方法でリモートログインを保護していても、システムのセキュリティーがリモートログインセッションレベルに低下します。セキュリティー保護されたリモートログインには、`ssh` コマンドを使用してください。

---

2 自己署名付き証明書を **`ike.privatekeys`** データベース内に作成します。

```
# ikecert certlocal -ks|-kc -m keysize -t keytype \
-D dname -A altname \
[-S validity-start-time] [-F validity-end-time] [-T token-ID]
```

`-ks`	自己署名付き証明書を作成します。
`-kc`	証明書要求を作成します。手順については、621 ページの「CA からの署名付き証明書により IKE を構成する方法」を参照してください。
`-m` *keysize*	鍵のサイズです。*keysize* は、512、1024、2048、3072、4096 のいずれかです。
`-t` *keytype*	使用するアルゴリズムのタイプを指定します。*keytype* は `rsa-sha1`、`rsa-md5`、`dsa-sha1` のいずれかです。
`-D` *dname*	証明書主体の X.509 識別名です。*dname* の一般的な形式は次のとおりです。C = country (国)、O = organization (組織)、OU = organizational unit (組織単位)、CN = common name (共通名)。有効なタグは、C、O、OU、CN です。

-A *altname*	証明書の代替名です。*altname* の形式は tag=value です。有効なタグは IP、DNS、email、および DN です。
-S *validity-start-time*	証明書の有効期間の開始時間を絶対値または相対値で指定します。
-F *validity-end-time*	証明書の有効期間の終了時間を絶対値または相対値で指定します。
-T *token-ID*	PKCS #11 ハードウェアトークンで鍵を生成できるようにします。その後、証明書はハードウェアに格納されます。

a. たとえば、**partym** システムでは、コマンドは次のようになります。

```
# ikecert certlocal -ks -m 1024 -t rsa-sha1 \
-D "C=US, O=PartyCo, OU=US-Partym, CN=Partym" \
-A IP=192.168.13.213
Creating software private keys.
  Writing private key to file /etc/inet/secret/ike.privatekeys/0.
Enabling external key providers - done.
Acquiring private keys for signing - done.
Certificate:
 Proceeding with the signing operation.
 Certificate generated successfully (.../publickeys/0)
Finished successfully.
Certificate added to database.
-----BEGIN X509 CERTIFICATE-----
MIICLTCCAZagAwIBAgIBATANBgkqhkiG9w0BAQQFADBNMQswCQYDVQQGEwJVUzEX
...
6sKTxpg4GP3GkQGcd0r1rhW/3yaWBkDwOdFCqEUyffzU
-----END X509 CERTIFICATE-----
```

注 – -D および -A オプションの値は任意です。値は証明書を識別するためだけに使用されます。これらは 192.168.13.213 などのシステムを識別するためには使用されません。実際、これらは固有の値であるため、正しい証明書がピアシステムにインストールされていることを帯域外で検証する必要があります。

---

b. **enigma** システムでは、コマンドは次のようになります。

```
# ikecert certlocal -ks -m 1024 -t rsa-sha1 \
-D "C=JA, O=EnigmaCo, OU=JA-Enigmax, CN=Enigmax" \
-A IP=192.168.116.16
Creating software private keys.
  ...
Certificate added to database.
-----BEGIN X509 CERTIFICATE-----
MIICKDCCAZGgAwIBAgIBATANBgkqhkiG9w0BAQQFADBJMQswCQYDVQQGEwJVUzEV
...
jpxfLM98xyFVyLCbkr3dZ3Tvxvi732BXePKF2A==
-----END X509 CERTIFICATE-----
```

3 証明書を保存し、リモートシステムに送信します。

証明書は、電子メールに貼り付けることもできます。

出力は、証明書の公開部分のエンコード済みバージョンです。この証明書を電子メールにペーストしても安全です。手順 5 で示すように、受け取り側は正しい証明書をインストールしていることを帯域外で検証する必要があります。

a. たとえば、次の **partym** 証明書の公開部分を **enigma** 管理者に送信します。

```
To: admin@ja.enigmaexample.com
From: admin@us.partyexample.com
Message: -----BEGIN X509 CERTIFICATE-----
MIICLTCCAZagAwIBAgIBATANBgkqhkiG9w0BAQQFADBNMQswCQYDVQQGEwJVUzEX
...
6sKTxpg4GP3GkQGcd0r1rhW/3yaWBkDwOdFCqEUyffzU
-----END X509 CERTIFICATE-----
```

b. **enigma** 管理者から、次の **enigma** 証明書の公開部分が送信されます。

```
To: admin@us.partyexample.com
From: admin@ja.enigmaexample.com
Message: -----BEGIN X509 CERTIFICATE-----
MIICKDCCAZGgAwIBAgIBATANBgkqhkiG9w0BAQQFADBJMQswCQYDVQQGEwJVUzEV
...
jpxfLM98xyFVyLCbkr3dZ3Tvxvi732BXePKF2A==
-----END X509 CERTIFICATE-----
```

4 システムごとに、受信した証明書を追加します。

a. 管理者の電子メールから公開鍵をコピーします。

b. **ikecert certdb -a** コマンドを入力して、**Return** キーを押します。

Return キーを押してもプロンプトは表示されません。

```
# ikecert certdb -a       Press the Return key
```

c. 公開鍵を貼り付けます。続いて **Return** キーを押します。**Control-D** キーを押して入力を終了します。

```
-----BEGIN X509 CERTIFICATE-----
MIIC...
...
----END X509 CERTIFICATE-----       Press the Return key
<Control>-D
```

5 通信するシステムの管理者と一緒に、証明書がその管理者のものであることを確認します。

たとえば、ほかの管理者に電話して、自分が持つ公開証明書のハッシュが、ほかの管理者のみが持つ非公開証明書のハッシュに一致することを検証できます。

a. **partym** に格納されている証明書を一覧表示します。

次の例で、Note 1 はスロット 0 の証明書の識別名 (DN) を示します。スロット 0 の非公開証明書は同じハッシュを持つ (注 3 を参照) ため、これらの証明書は同一の証明書ペアです。公開証明書が機能するには、一致するペアを持つ必要があります。`certdb` サブコマンドは公開部分を示し、`certlocal` サブコマンドは非公開部分を示します。

```
partym # ikecert certdb -l
Certificate Slot Name: 0   Type: rsa-sha1
    Subject Name: <C=US, O=PartyCo, OU=US-Partym, CN=Partym>     Note 1
    Key Size: 1024
    Public key hash: 2239A6A127F88EE0CB40F7C24A65B818

Certificate Slot Name: 1   Type: rsa-sha1
    (Private key in certlocal slot 0)
    Subject Name: <C=JA, O=EnigmaCo, OU=JA-Enigmax, CN=Enigmax>
    Key Size: 1024
    Public key hash: B2BD13FCE95FD27ECE6D2DCD0DE760E2
partym # ikecert certlocal -l
Local ID Slot Name: 0   Key Type: rsa-sha1
    Key Size: 1024
    Public key hash: 2239A6A127F88EE0CB40F7C24A65B818     Note 3

Local ID Slot Name: 1   Key Type: rsa-sha1
        Key Size: 1024
        Public key hash: FEA65C5387BBF3B2C8F16C019FEBC388
...
```

このチェックでは、`partym` システムが有効な証明書ペアを持つことが検証されました。

b. **enigma** システムが **partym** の公開証明書を持つことを確認します。

公開鍵ハッシュは電話で伝えることができます。

前の手順の `partym` における Note 3 のハッシュと、`enigma` における Note 4 を比較します。

```
enigma # ikecert certdb -l
Certificate Slot Name: 4   Type: rsa-sha1
    Subject Name: <C=JA, O=EnigmaCo, OU=JA-Enigmax, CN=Enigmax>
    Key Size: 1024
    Public key hash: DF3F108F6AC669C88C6BD026B0FCE3A0

Certificate Slot Name: 5   Type: rsa-sha1
```

```
    Subject Name: <C=US, O=PartyCo, OU=US-Partym, CN=Partym>
    Key Size: 1024
    Public key hash: 2239A6A127F88EE0CB40F7C24A65B818     Note 4
```

enigma の公開証明書データベースに格納されている最後の証明書の公開鍵ハッシュとサブジェクト名が、前の手順の partym の非公開証明書と一致します。

6 システムごとに、両方の証明書を信頼します。

/etc/inet/ike/config ファイルを編集して、証明書を認識します。

パラメータ cert_trust、remote_addr、および remote_id の値は、リモートシステムの管理者が提供します。

a. たとえば、**partym** システム上の **ike/config** ファイルは次のようになります。

```
# Explicitly trust the self-signed certs
# that we verified out of band. The local certificate
# is implicitly trusted because we have access to the private key.
cert_trust "192.168.116.16"      Remote system's certificate Subject Alt Name

## Parameters that may also show up in rules.

p1_xform
  { auth_method preshared oakley_group 5 auth_alg sha encr_alg 3des }
p2_pfs 5

{
 label "US-partym to JA-enigmax"
 local_id_type dn
 local_id "C=US, O=PartyCompany, OU=US-Partym, CN=Partym"
 remote_id "C=JA, O=EnigmaCo, OU=JA-Enigmax, CN=Enigmax"

 local_addr  192.168.13.213

# We could explicitly enter the peerâs IP address here, but we donât need
# to do this with certificates, so use a wildcard address. The wildcard
# allows the remote device to be mobile or behind a NAT box.
# remote_addr 192.168.116.16
 remote_addr 0.0.0.0/0

 p1_xform
   { auth_method rsa_sig oakley_group 2 auth_alg sha1 encr_alg aes }
}
```

b. **enigma** システムで、**ike/config** ファイルにローカルパラメータの **enigma** 値を追加します。

リモートパラメータには、partym 値を使用します。label キーワードの値がローカルシステム上で一意であることを確認します。

```
...
{
 label "JA-enigmax to US-partym"
 local_id_type dn
 local_id "C=JA, O=EnigmaCo, OU=JA-Enigmax, CN=Enigmax"
 remote_id "C=US, O=PartyCompany, OU=US-Partym, CN=Partym"

 local_addr  192.168.116.16
 remote_addr 0.0.0.0/0
...
```

例 **23–4** 証明書の開始時間と終了時間を指定する

この例では、partym システムの管理者が、証明書の有効期間の日付を設定します。証明書は、2.5 日前の発効とし、作成日から 4 年 6 か月間有効とします。

```
# ikecert certlocal -ks -m 1024 -t rsa-sha1 \
-D "C=US, O=PartyCompany, OU=US-Partym, CN=Partym" \
-A IP=192.168.13.213 \
-S -2d12h -F +4y6m
```

enigma システムの管理者が、証明書の有効期間の日付を設定します。証明書は、2 日前の発効とし、2010 年 12 月 31 日の午前 0 時まで有効とします。

```
# ikecert certlocal -ks -m 1024 -t rsa-sha1 \
-D "C=JA, O=EnigmaCo, OU=JA-Enigmax, CN=Enigmax" \
-A IP=192.168.116.16 \
-S -2d -F "12/31/2010 12:00 AM"
```

▼ CA からの署名付き証明書により IKE を構成する方法

認証局 (CA) からの公開鍵証明書では、外部機関とのネゴシエーションが必要となります。この証明書は非常に簡単に拡大できるため、通信するシステムを数多く保護できます。

1 システムコンソール上で、**Primary Administrator** の役割を引き受けるか、スーパーユーザーになります。

Primary Administrator 役割には、Primary Administrator プロファイルが含まれます。役割を作成してユーザーに役割を割り当てるには、『Oracle Solaris の管理: 基本管理』の第 2 章「Solaris 管理コンソールの操作 (タスク)」を参照してください。

注 – リモートログインすると、セキュリティー上重要なトラフィックが盗聴される恐れがあります。何らかの方法でリモートログインを保護していても、システムのセキュリティーがリモートログインセッションレベルに低下します。セキュリティー保護されたリモートログインには、ssh コマンドを使用してください。

2 **ikecert certlocal -kc** コマンドを使用して、証明書要求を作成します。

コマンドの引数については、616 ページの「自己署名付き公開鍵証明書により IKE を構成する方法」の手順 2 を参照してください。

```
# ikecert certlocal -kc -m keysize -t keytype \
-D dname -A altname
```

a. たとえば、次のコマンドでは、**partym** システム上に証明書要求が作成されます。

```
# ikecert certlocal -kc -m 1024 -t rsa-sha1 \
> -D "C=US, O=PartyCompany\, Inc., OU=US-Partym, CN=Partym" \
> -A "DN=C=US, O=PartyCompany\, Inc., OU=US-Partym"
Creating software private keys.
  Writing private key to file /etc/inet/secret/ike.privatekeys/2.
Enabling external key providers - done.
Certificate Request:
  Proceeding with the signing operation.
  Certificate request generated successfully (.../publickeys/0)
Finished successfully.
-----BEGIN CERTIFICATE REQUEST-----
MIIByjCCATMCAQAwUzELMAkGA1UEBhMCVVMxHTAbBgNVBAoTFEV4YW1wbGVDb21w
...
lcM+tw0ThRrfuJX9t/Qa1R/KxRlMA3zckO80mO9X
-----END CERTIFICATE REQUEST-----
```

b. 次のコマンドでは、**enigma** システム上に証明書要求が作成されます。

```
# ikecert certlocal -kc -m 1024 -t rsa-sha1 \
> -D "C=JA, O=EnigmaCo\, Inc., OU=JA-Enigmax, CN=Enigmax" \
> -A "DN=C=JA, O=EnigmaCo\, Inc., OU=JA-Enigmax"
Creating software private keys.
...
Finished successfully.
-----BEGIN CERTIFICATE REQUEST-----
MIIBuDCCASECAQAwSTELMAkGA1UEBhMCVVMxFTATBgNVBAoTDFBhcnR5Q29tcGFu
...
8qlqdjaStLGfhDOO
-----END CERTIFICATE REQUEST-----
```

3 この証明書要求を **PKI** 機関に送信します。

証明書要求の送信方法については PKI に問い合わせてください。ほとんどの機関は、Web サイトに送信フォームを掲載しています。フォームの記入に当たっては、その送信が正当なものであることを証明する必要があります。通常は、証明書要求をフォームに貼り付けます。要求を受け取った機関は、それをチェックしてから、次の 2 つの証明書オブジェクトと、証明書失効リストを発行します。

- 公開鍵証明書 – この証明書は機関に送信した要求に基づいて作成されます。送信した証明書要求も、公開鍵証明書の一部として含まれます。この証明書によって一意に識別されます。
- 認証局 – 機関の署名。CA によって公開鍵証明書が正規のものであることが確認されます。
- 証明書失効リスト (CRL) – 機関が無効にした証明書の最新リストです。CRL へのアクセスが公開鍵証明書に組み込まれている場合には、CRL が別個の証明書オブジェクトとして送信されることはありません。

  CRL の URI が公開鍵証明書に組み込まれている場合には、IKE は CRL を自動的に取り出すことができます。同様に、DN (LDAP サーバー上のディレクトリ名) エントリが公開鍵証明書に組み込まれている場合には、IKE は、指定された LDAP サーバーから CRL を取得し、キャッシュできます。

  公開鍵証明書に組み込まれている URI と DN エントリの例については、631 ページの「証明書失効リストを処理する方法」を参照してください。

4 各証明書をシステムに追加します。

`ikecert certdb -a` コマンドの `-a` オプションは、張り付けられたオブジェクトをシステムの適切な証明書データベースに追加します。 詳細については、598 ページの「IKE と公開鍵証明書」を参照してください。

a. システムコンソール上で、**Primary Administrator** の役割を引き受けるか、スーパーユーザーになります。

b. **PKI** 機関から受け取った公開鍵証明書を追加します。

```
# ikecert certdb -a
      Press the Return key
      Paste the certificate:
-----BEGIN X509 CERTIFICATE-----
...
-----END X509 CERTIFICATE----
      Press the Return key
<Control>-D
```

c. **PKI** 機関の **CA** を追加します。

```
# ikecert certdb -a
      Press the Return key
      Paste the CA:
```

```
-----BEGIN X509 CERTIFICATE-----
...
-----END X509 CERTIFICATE----
     Press the Return key
<Control>-D
```

d. **PKI** 機関が証明書失効リスト **(CRL)** を送信してきている場合は、これを **certrldb** データベースに追加します。

```
# ikecert certrldb -a
     Press the Return key
     Paste the CRL:
-----BEGIN CRL-----
...
-----END CRL----
     Press the Return key
<Control>-D
```

5 **cert_root** キーワードを使用して、**/etc/inet/ike/config** ファイルの **PKI** 機関を識別します。

PKI 機関が提供する名前を使用します。

a. たとえば、**partym** システムの **ike/config** ファイルは次のようになります。

```
# Trusted root cert
# This certificate is from Example PKI
# This is the X.509 distinguished name for the CA that it issues.

cert_root "C=US, O=ExamplePKI\, Inc., OU=PKI-Example, CN=Example PKI"

## Parameters that may also show up in rules.

p1_xform
  { auth_method rsa_sig oakley_group 1 auth_alg sha1 encr_alg 3des }
p2_pfs 2

{
 label "US-partym to JA-enigmax - Example PKI"
 local_id_type dn
 local_id  "C=US, O=PartyCompany, OU=US-Partym, CN=Partym"
 remote_id "C=JA, O=EnigmaCo, OU=JA-Enigmax, CN=Enigmax"

 local_addr  192.168.13.213
 remote_addr 192.168.116.16

 p1_xform
   { auth_method rsa_sig oakley_group 2 auth_alg sha1 encr_alg aes }
}
```

注 – auth_method パラメータのすべての引数は同じ行になければなりません。

---

b. enigma システム上で、同様なファイルを作成します。

特に、enigma ike/config ファイルは、次の条件を満たしている必要があります。

- cert_root には同じ値を使用する。
- ローカルパラメータには enigma 値を使用する。
- リモートパラメータには partym 値を使用する。
- label キーワードには一意の値を作成する。この値は、リモートシステムの label 値とは異なる値でなくてはなりません。

```
...
cert_root "C=US, O=ExamplePKI\, Inc., OU=PKI-Example, CN=Example PKI"
...
{
 label "JA-enigmax to US-partym - Example PKI"
 local_id_type dn
 local_id   "C=JA, O=EnigmaCo, OU=JA-Enigmax, CN=Enigmax"
 remote_id  "C=US, O=PartyCompany, OU=US-Partym, CN=Partym"

 local_addr  192.168.116.16
 remote_addr 192.168.13.213
...
```

6 CRL を処理する方法を IKE に伝えます。

適切なオプションを選択します。

- **CRL 利用不可**

  PKI 機関が CRL を提供しない場合、キーワード ignore_crls を ike/config ファイルに追加します。

  ```
  # Trusted root cert
  ...
  cert_root "C=US, O=ExamplePKI\, Inc., OU=PKI-Example,...
  ignore_crls
  ...
  ```

  ignore_crls キーワードにより、IKE は CRL を検索しなくなります。

- **CRL 利用可能**

  PKI 機関から CRL の一元的な配布ポイントを知らされている場合は、ike/config ファイルを変更してこの場所を指定できます。

  例については、631 ページの「証明書失効リストを処理する方法」を参照してください。

例 23–5 IKE の構成時における rsa_encrypt の使用

ike/config ファイルで auth_method rsa_encrypt を使用する場合には、ピアの証明書を publickeys データベースに追加する必要があります。

1. その証明書をリモートシステムの管理者に送信します。

   証明書は、電子メールに貼り付けることもできます。

   たとえば、partym の管理者は次のような電子メールを送信します。

   ```
   To: admin@ja.enigmaexample.com
   From: admin@us.partyexample.com
   Message: -----BEGIN X509 CERTIFICATE-----
   MII...
   ----END X509 CERTIFICATE-----
   ```

   enigma の管理者は次のような電子メールを送信します。

   ```
   To: admin@us.partyexample.com
   From: admin@ja.enigmaexample.com
   Message: -----BEGIN X509 CERTIFICATE-----
   MII
   ...
   -----END X509 CERTIFICATE-----
   ```

2. システムごとに、電子メールで送信された証明書をローカルの publickeys データベースに追加します。

   ```
   # ikecert certdb -a
       Press the Return key
   -----BEGIN X509 CERTIFICATE-----
   MII...
   -----END X509 CERTIFICATE-----
       Press the Return key
   <Control>-D
   ```

RSA 暗号化の認証方法は、IKE 内の識別子を盗聴者から隠します。rsa_encrypt メソッドはピアの識別子を隠すため、IKE はピアの証明書を取得できません。結果として、rsa_encrypt メソッドでは、IKE ピアが互いの公開鍵を知っておく必要があります。

よって、/etc/inet/ike/config ファイルの auth_method に rsa_encrypt を指定する場合には、ピアの証明書を publickeys データベースに追加する必要があります。この結果、publickeys データベースには、通信するシステムペアごとに 3 つの証明書が存在することになります。

- ユーザーの公開鍵証明書
- CA 証明書
- ピアの公開鍵証明書

トラブルシューティング – IKE ペイロードは 3 つの証明書を持っており、大きくなりすぎて、rsa_encrypt が暗号化できないことがあります。「authorization failed (承認

に失敗しました)」や「malformed payload (ペイロードが不正です)」などのエラーは、rsa_encrypt メソッドがペイロード全体を暗号化できないことを示します。証明書を 2 つしか必要としない rsa_sig などのメソッドを使用して、ペイロードのサイズを減らします。

▼ ハードウェアで公開鍵証明書を生成および格納する方法

ハードウェア上で公開鍵証明書を生成および格納することは、システム上で公開鍵証明書を生成および格納することと似ています。ハードウェア上では、ikecert certlocal および ikecert certdb コマンドがハードウェアを識別しなければなりません。トークン ID に -T オプションを指定すると、コマンドがハードウェアを識別するようになります。

始める前に
- ハードウェアの構成が完了していること。
- /etc/inet/ike/config ファイルの pkcs11_path キーワードが別のライブラリを指している場合を除き、ハードウェアは /usr/lib/libpkcs11.so ライブラリを使用します。ライブラリが、 RSA Security Inc. PKCS #11 Cryptographic Token Interface (Cryptoki) に準拠して実装されているライブラリ、すなわち PKCS #11 ライブラリであること。

  設定の手順については、642 ページの「Sun Crypto Accelerator 4000 ボードを検出するように IKE を構成する方法」を参照してください。

1. システムコンソール上で、**Primary Administrator** の役割を引き受けるか、スーパーユーザーになります。

   Primary Administrator 役割には、Primary Administrator プロファイルが含まれます。役割を作成してユーザーに役割を割り当てるには、『Oracle Solaris の管理: 基本管理』の第 2 章「Solaris 管理コンソールの操作 (タスク)」を参照してください。

   注 – リモートログインすると、セキュリティー上重要なトラフィックが盗聴される恐れがあります。何らかの方法でリモートログインを保護していても、システムのセキュリティーがリモートログインセッションレベルに低下します。セキュリティー保護されたリモートログインには、ssh コマンドを使用してください。

2. 自己署名付き証明書または証明書要求を作成して、トークン **ID** を指定します。

   次のオプションのいずれかを選択します。

   注 – Sun Crypto Accelerator 4000 および Sun Crypto Accelerator 6000 ボードは、RSA で最大 2048 ビットの鍵をサポートします。DSA の場合、最大 1024 ビットの鍵をサポートします。

- 自己署名付き証明書の場合、次の構文を使用する

```
# ikecert certlocal -ks -m 1024 -t rsa-sha1 \
> -D "C=US, O=PartyCompany, OU=US-Partym, CN=Partym" \
> -a -T dca0-accel-stor IP=192.168.116.16
Creating hardware private keys.
Enter PIN for PKCS#11 token:      Type user:password
```

  -T オプションの引数は、接続されたボードのトークン ID です。

- 証明書要求の場合、次の構文を使用します。

```
# ikecert certlocal -kc -m 1024 -t rsa-sha1 \
> -D "C=US, O=PartyCompany, OU=US-Partym, CN=Partym" \
> -a -T dca0-accel-stor IP=192.168.116.16
Creating hardware private keys.
Enter PIN for PKCS#11 token:      Type user:password
```

ikecert コマンドの引数の詳細については、ikecert(1M) のマニュアルページを参照してください。

3 **PIN** のプロンプトに、ボードのユーザー、コロン、ユーザーのパスワードを入力します。

ボードのユーザー ikemgr のパスワードが rgm4tigt の場合、次のように入力します。

```
Enter PIN for PKCS#11 token: ikemgr:rgm4tigt
```

注 – PIN の応答は、ディスク上に「クリアテキストとして」格納されます。

パスワードの入力後、証明書が印刷されます。

```
Enter PIN for PKCS#11 token: ikemgr:rgm4tigt
-----BEGIN X509 CERTIFICATE-----
MIIBuDCCASECAQAwSTELMAkGA1UEBhMCVVMxFTATBgNVBAoTDFBhcnR5Q29tcGFu
...
oKUDBbZ9O/pLWYGr
-----END X509 CERTIFICATE-----
```

4 通信先に証明書を送信します。

次のオプションのいずれかを選択します。

- リモートシステムに自己署名付き証明書を送信します。

  証明書は、電子メールに貼り付けることもできます。

- **PKI** を処理する機関に証明書要求を送信します。

  証明書要求は、PKI 機関の指示に従って送信します。詳細については、手順 3 of 621 ページの「CA からの署名付き証明書により IKE を構成する方法」を参照してください。

5 システム上で、**/etc/inet/ike/config** ファイルを編集して、証明書が認識されるようにします。

次のオプションのどちらか 1 つを選択します。

- 自己署名付き証明書

  リモートシステムの管理者がパラメータ cert_trust、remote_id、および remote_addr 用に提供する値を使用します。たとえば、enigma システムの ike/config ファイルは次のようになります。

```
# Explicitly trust the following self-signed certs
# Use the Subject Alternate Name to identify the cert

cert_trust "192.168.116.16"       Local system's certificate Subject Alt Name
cert_trust "192.168.13.213"       Remote system's certificate Subject Alt name


# Solaris 10 1/06 release: default path does not have to be typed in #pkcs11_path
"/usr/lib/libpkcs11.so"      Hardware connection

# Solaris 10 release: use this path
#pkcs11_path "/opt/SUNWconn/cryptov2/lib/libvpkcs11.so"
...
{
 label "JA-enigmax to US-partym"
 local_id_type dn
 local_id "C=JA, O=EnigmaCo, OU=JA-Enigmax, CN=Enigmax"
 remote_id "C=US, O=PartyCompany, OU=US-Partym, CN=Partym"

 local_addr  192.168.116.16
 remote_addr 192.168.13.213

 p1_xform
  {auth_method rsa_sig oakley_group 2 auth_alg sha1 encr_alg aes}
}
```

- 証明書リクエスト

  PKI 機関が cert_root キーワードの値として提供する名前を入力します。たとえば、enigma システムの ike/config ファイルは次のようになります。

```
# Trusted root cert
# This certificate is from Example PKI
# This is the X.509 distinguished name for the CA that it issues.

cert_root "C=US, O=ExamplePKI\, Inc., OU=PKI-Example, CN=Example PKI"

# Solaris 10 1/06 release: default path does not have to be typed in #pkcs11_path
"/usr/lib/libpkcs11.so"      Hardware connection


# Solaris 10 release: use this path
```

```
#pkcs11_path "/opt/SUNWconn/cryptov2/lib/libvpkcs11.so"
...
{
 label "JA-enigmax to US-partym - Example PKI"
 local_id_type dn
 local_id "C=JA, O=EnigmaCo, OU=JA-Enigmax, CN=Enigmax"
 remote_id  "C=US, O=PartyCompany, OU=US-Partym, CN=Partym"

 local_addr  192.168.116.16
 remote_addr 192.168.13.213

 p1_xform
  {auth_method rsa_sig oakley_group 2 auth_alg sha1 encr_alg aes}
}
```

6 通信先から受け取った証明書をハードウェアに格納します。
手順 3 で応答したように、PIN 要求に応答します。

注 – 公開鍵証明書は、公開鍵を生成したハードウェアに追加する必要があります。

- 自己署名付き証明書。
  リモートシステムの自己署名付き証明書を追加します。この例では、証明書は DCA.ACCEL.STOR.CERT ファイルに格納されています。

  ```
  # ikecert certdb -a -T dca0-accel-stor < DCA.ACCEL.STOR.CERT
  Enter PIN for PKCS#11 token:      Type user:password
  ```

  自己署名付き証明書が rsa_encrypt を auth_method パラメータの値として使用していた場合、ピアの証明書をハードウェア格納場所に追加します。

- **PKI 機関からの証明書**
  機関が証明書要求から生成した証明書を追加して、認証局 (CA) を追加します。

  ```
  # ikecert certdb -a -T dca0-accel-stor < DCA.ACCEL.STOR.CERT
  Enter PIN for PKCS#11 token:      Type user:password

  # ikecert certdb -a -T dca0-accel-stor < DCA.ACCEL.STOR.CA.CERT
  Enter PIN for PKCS#11 token:      Type user:password
  ```

  PKI 機関からの証明書失効リスト (CRL) を追加する方法については、631 ページの「証明書失効リストを処理する方法」を参照してください。

▼ 証明書失効リストを処理する方法

証明書失効リスト (CRL) には、認証局が発行した証明書のうち、期限切れになったりセキュリティーが低下したりした証明書が含まれます。CRL を処理する方法には、次の 4 つがあります。

- CA 機関が CRL を発行しない場合、CRL を無視するように IKE に指示する必要があります。このオプションは、手順 6 in 621 ページの「CA からの署名付き証明書により IKE を構成する方法」に示されています。
- CA から受け取った公開鍵証明書に URI (Uniform Resource Indicator) のアドレスが組み込まれている場合は、URI から CRL にアクセスするように IKE に指示することができます。
- CA から受け取った公開鍵証明書に LDAP サーバーの DN (ディレクトリ名) エントリが組み込まれている場合は、LDAP サーバーから CRL にアクセスするように IKE に指示することができます。
- CRL は `ikecert certrldb` コマンドへの引数として指定できます。例については、例 23–6 を参照してください。

次の手順に、中央の配布ポイントから CRL を使用するように IKE に指示する手順を示します。

1 CA から受信した証明書を表示する

```
# ikecert certdb -lv certspec
```

`-l` IKE 証明書データベースにある証明書を一覧表示します。

`-v` 証明書を冗長モードで一覧表示します。このオプションは慎重に使用してください。

certspec IKE 証明書データベース内の証明書と一致するパターンです。

たとえば、次の証明書は Oracle が発行しました。詳細は変更されています。

```
# ikecert certdb -lv example-protect.oracle.com
Certificate Slot Name: 0   Type: dsa-sha1
   (Private key in certlocal slot 0)
 Subject Name: <O=Oracle, CN=example-protect.oracle.com>
 Issuer Name: <CN=Oracle CA (Cl B), O=Oracle>
 SerialNumber: 14000D93
   Validity:
      Not Valid Before: 2002 Jul 19th, 21:11:11 GMT
      Not Valid After:  2005 Jul 18th, 21:11:11 GMT
   Public Key Info:
      Public Modulus  (n) (2048 bits): C575A...A5
      Public Exponent (e) (  24 bits): 010001
   Extensions:
      Subject Alternative Names:
              DNS = example-protect.oracle.com
      Key Usage: DigitalSignature KeyEncipherment
```

```
   [CRITICAL]
CRL Distribution Points:
   Full Name:
      URI = #Ihttp://www.oracle.com/pki/pkismica.crl#i
      DN = <CN= Oracle CA (Cl B), O=Oracle>
   CRL Issuer:
   Authority Key ID:
   Key ID:              4F ... 6B
   SubjectKeyID:        A5 ... FD
   Certificate Policies
   Authority Information Access
```

CRL Distribution Points エントリに注目してください。URI エントリは、この機関の CRL が Web 上にあることを示しています。DN エントリは、CRL が LDAP サーバー上にあることを示しています。一度、IKE がアクセスすると、CRL は将来に備えてキャッシュに格納されます。

CRL にアクセスするには、配布ポイントまで到達する必要があります。

2 中央の配布ポイントから **CRL** にアクセスするには、次のメソッドのうちの **1** つを選択します。

- **URI** を使用します。

  キーワード use_http をホストの /etc/inet/ike/config ファイルに追加します。たとえば、ike/config ファイルは次のようになります。

  ```
  # Use CRL from organization's URI
  use_http
  ...
  ```

- **Web** プロキシを使用します。

  キーワード proxy を ike/config ファイルに追加します。キーワード proxy は、次のように引数として URL を取ります。

  ```
  # Use own web proxy
  proxy "http://proxy1:8080"
  ```

- **LDAP** サーバーを使用します。

  ホストの /etc/inet/ike/config ファイルの ldap-list キーワードに LDAP サーバーの名前を指定します。LDAP サーバーの名前は、使用する機関にたずねてください。ike/config ファイルのエントリは次のようになります。

  ```
  # Use CRL from organization's LDAP
  ldap-list "ldap1.oracle.com:389,ldap2.oracle.com"
  ...
  ```

IKE は CRL を取り出し、証明書の期限が切れるまで CRL を保持します。

例 23–6 CRL をローカルの certrldb データベースに貼り付ける

使用する機関の証明書に一元的な配布ポイントが含まれていない場合は、機関の CRL を手動でローカルの `certrldb` データベースに追加できます。機関の説明に従って CRL をファイルに抽出し、それを `ikecert certrldb -a` コマンドでデータベースに追加します。

```
# ikecert certrldb -a < Oracle.Cert.CRL
```

移動体システム用の IKE の構成 (タスクマップ)

次の表に、中央サイトにリモートからログインするシステムを処理するように、IKE を構成する手順を示します。

タスク	説明	参照先
オフサイトから中央サイトへ通信します	遠隔地のシステムが中央サイトと通信できるようにします。遠隔地のシステムは移動体システムの可能性もあります。	634 ページの「遠隔地のシステム用に IKE を構成する方法」
移動体システムからのトラフィックを受信する中央システムで CA の公開証明書と IKE を使用します	固定 IP アドレスを持たないシステムからの IPsec トラフィックを受信するゲートウェイシステムを構成します。	例 23–7
固定 IP アドレスを持たないシステムで CA の公開証明書と IKE を使用します	中央サイト (会社の本社など) とのトラフィックを保護するように、移動体システムを構成します。	例 23–8
移動体システムからのトラフィックを受信する中央システムで自己署名付き証明書と IKE を使用します	移動体システムから IPsec トラフィックを受信するように、ゲートウェイシステムを自己署名付き証明書で構成します。	例 23–9
固定 IP アドレスを持たないシステムで自己署名付き証明書と IKE を使用します	中央サイトとのトラフィックを保護するように、移動体システムを自己署名付き証明書で構成します。	例 23–10

移動体システム用の IKE の構成

適切に構成することで、ホームオフィスやノートブックから IPsec と IKE を使用して、会社の中央コンピュータと通信できます。公開鍵認証方法と結びついたブランケット IPsec ポリシーを使用すると、遠隔地のシステムは中央システムとのトラフィックを保護できます。

▼ 遠隔地のシステム用に IKE を構成する方法

ソースと宛先を識別するために、IPsec と IKE は一意の ID を必要とします。一意の IP アドレスを持たない遠隔地のシステムまたは移動体システムの場合、別の種類の ID を使用する必要があります。システムを一意に識別するために、`DNS`、`DN`、または `email` などの ID の種類を使用できます。

一意の IP アドレスを持つ遠隔地のシステムまたは移動体システムで、別の種類の ID で構成するようにします。たとえば、システムが NAT 越しに中央システムに接続しようとした場合、そのシステムの一意なアドレスは使用されません。NAT ボックスが任意の IP アドレスを割り当てるため、中央システムは認識できません。

事前共有鍵は固定 IP アドレスを必要とするため、事前共有鍵も移動体システム用の認証メカニズムとしては機能しません。自己署名付き証明書 (または PKI からの証明書) を使用すると、移動体システムは中央サイトと通信できます。

1 システムコンソール上で、**Primary Administrator** の役割を引き受けるか、スーパーユーザーになります。

Primary Administrator 役割には、Primary Administrator プロファイルが含まれます。役割を作成してユーザーに役割を割り当てるには、『Oracle Solaris の管理: 基本管理』の第 2 章「Solaris 管理コンソールの操作 (タスク)」を参照してください。

注 – リモートログインすると、セキュリティー上重要なトラフィックが盗聴される恐れがあります。何らかの方法でリモートログインを保護していても、システムのセキュリティーがリモートログインセッションレベルに低下します。セキュリティー保護されたリモートログインには、`ssh` コマンドを使用してください。

---

2 移動体システムを認識するように、中央システムを構成します。

a. **`ipsecinit.conf`** ファイルを設定します。

中央システムには、IP アドレスの広い範囲を許可するポリシーを必要とします。そのあと、IKE ポリシーの証明書で接続システムが合法であることを確認します。

```
# /etc/inet/ipsecinit.conf on central
# Keep everyone out unless they use this IPsec policy:
{} ipsec {encr_algs aes encr_auth_algs sha1 sa shared}
```

b. **`ike.config`** ファイルを設定します。

DNS は中央システムを識別します。証明書を使用して、システムを認証します。

```
## /etc/inet/ike/ike.config on central
# Global parameters
#
# Find CRLs by URI, URL, or LDAP
# Use CRL from organization's URI
use_http
```

```
#
# Use web proxy
proxy "http://somecache.domain:port/"
#
# Use LDAP server
ldap_server   "ldap-server1.domain.org,ldap2.domain.org:port"
#
# List CA-signed certificates
cert_root    "C=US, O=Domain  Org, CN=Domain STATE"
#
# List self-signed certificates - trust server and enumerated others
#cert_trust    "DNS=central.domain.org"
#cert_trust    "DNS=mobile.domain.org"
#cert_trust    "DN=CN=Domain Org STATE (CLASS), O=Domain Org
#cert_trust    "email=root@central.domain.org"
#cert_trust    "email=user1@mobile.domain.org"
#

# Rule for mobile systems with certificate
{
  label "Mobile systems with certificate"
  local_id_type DNS

# CA's public certificate ensures trust,
# so allow any remote_id and any remote IP address.
  remote_id ""
  remote_addr 0.0.0.0/0

p2_pfs 5

p1_xform
{auth_method rsa_sig oakley_group 5 encr_alg blowfish auth_alg sha1 }
}
```

3 各移動体システムにログインして、中央システムを見つけるように構成します。

a. **/etc/hosts** ファイルを設定します。

/etc/hosts ファイルは、移動体システムのアドレスを必要としませんが、提供することは可能です。このファイルは、中央システムの公開 IP アドレスを含んでいる必要があります。

```
# /etc/hosts on mobile
central 192.xxx.xxx.x
```

b. **ipsecinit.conf** ファイルを設定します。

移動体システムは、公開 IP アドレスで中央システムを見つける必要があります。システムは同じ IPsec ポリシーで構成する必要があります。

```
# /etc/inet/ipsecinit.conf on mobile
# Find central
{raddr 192.xxx.xxx.x} ipsec {encr_algs aes encr_auth_algs sha1 sa shared}
```

c. **ike.config** ファイルを設定します。

識別子は IP アドレスであってはなりません。移動体システムに有効な識別子は次のとおりです。

- DN=*ldap-directory-name*
- DNS=*domain-name-server-address*
- email=*email-address*

証明書を使用して、移動体システムを認証します。

```
## /etc/inet/ike/ike.config on mobile
# Global parameters
#
# Find CRLs by URI, URL, or LDAP
# Use CRL from organization's URI
use_http
#
# Use web proxy
proxy "http://somecache.domain:port/"
#
# Use LDAP server
ldap_server   "ldap-server1.domain.org,ldap2.domain.org:port"
#
# List CA-signed certificates
cert_root    "C=US, O=Domain  Org, CN=Domain STATE"
#
# Self-signed certificates - trust me and enumerated others
#cert_trust    "DNS=mobile.domain.org"
#cert_trust    "DNS=central.domain.org"
#cert_trust    "DN=CN=Domain Org STATE (CLASS), O=Domain Org
#cert_trust    "email=user1@domain.org"
#cert_trust    "email=root@central.domain.org"
#
# Rule for off-site systems with root certificate
{
    label "Off-site mobile with certificate"
    local_id_type DNS

# NAT-T can translate local_addr into any public IP address
# central knows me by my DNS

    local_id "mobile.domain.org"
    local_addr 0.0.0.0/0

# Find central and trust the root certificate
    remote_id "central.domain.org"
    remote_addr 192.xxx.xxx.x

p2_pfs 5

p1_xform
{auth_method rsa_sig oakley_group 5 encr_alg blowfish auth_alg sha1 }
}
```

4 **IKE** の構成をカーネルに読み込みます。

- **Solaris 10 4/09** リリース以降では、**ike** サービスを使用可能にします。

  ```
  # svcadm enable svc:/network/ipsec/ike
  ```

- **Solaris 10 4/09** リリースより前のリリースを実行している場合は、システムをリブートします。

  ```
  # init 6
  ```

  あるいは、in.iked デーモンを停止および起動します。

例 23–7 移動体システムからの IPsec トラフィックを受信するための中央コンピュータの構成

IKE は、NAT ボックス越しのネゴシエーションを開始できます。しかし、IKE の理想的な設定は NAT ボックスをはさまないことです。次の例では、CA の公開証明書は移動体システムと中央システムに格納されています。中央システムは NAT 越しのシステムからの IPsec ネゴシエーションを受け入れます。main1 は、遠隔地のシステムからの接続を受け入れることができる会社のシステムです。遠隔地のシステムを設定する方法については、例 23–8 を参照してください。

```
## /etc/hosts on main1
main1 192.168.0.100

## /etc/inet/ipsecinit.conf on main1
# Keep everyone out unless they use this IPsec policy:
{} ipsec {encr_algs aes encr_auth_algs sha1 sa shared}

## /etc/inet/ike/ike.config on main1
# Global parameters
#
# Find CRLs by URI, URL, or LDAP
# Use CRL from organization's URI
use_http
#
# Use web proxy
proxy "http://cache1.domain.org:8080/"
#
# Use LDAP server
ldap_server   "ldap1.domain.org,ldap2.domain.org:389"
#
# List CA-signed certificate
cert_root "C=US, O=ExamplePKI Inc, OU=PKI-Example, CN=Example PKI"
#
# Rule for off-site systems with root certificate
{
  label "Off-site system with root certificate"
  local_id_type DNS
  local_id "main1.domain.org"
  local_addr 192.168.0.100

# CA's public certificate ensures trust,
# so allow any remote_id and any remote IP address.
  remote_id ""
  remote_addr 0.0.0.0/0

p2_pfs 5
```

```
p1_xform
{auth_method rsa_sig oakley_group 5 encr_alg blowfish auth_alg sha1}
p1_xform
{auth_method rsa_sig oakley_group 5 encr_alg aes auth_alg sha1}
p1_xform
{auth_method rsa_sig oakley_group 5 encr_alg blowfish auth_alg sha1}
p1_xform
{auth_method rsa_sig oakley_group 5 encr_alg aes auth_alg sha1}
}
```

例 23–8 NAT 越しのシステムの IPsec による構成

次の例では、CA の公開証明書は移動体システムと中央システムに格納されています。mobile1 は、家から会社の本社に接続しています。インターネットサービスプロバイダ (ISP) ネットワークは NAT ボックスを使用しているため、ISP は mobile1 に非公開アドレスを割り当てることができます。NAT ボックスは、非公開アドレスを公開 IP アドレスに変換します。この公開アドレスは、ほかの ISP ネットワークノードと共有されます。企業の本社は NAT を越えません。企業の本社のコンピュータを設定する方法については、例 23–7を参照してください。

```
## /etc/hosts on mobile1
mobile1 10.1.3.3
main1 192.168.0.100

## /etc/inet/ipsecinit.conf on mobile1
# Find main1
{raddr 192.168.0.100} ipsec {encr_algs aes encr_auth_algs sha1 sa shared}

## /etc/inet/ike/ike.config on mobile1
# Global parameters
#
# Find CRLs by URI, URL, or LDAP
# Use CRL from organization's URI
use_http
#
# Use web proxy
proxy "http://cache1.domain.org:8080/"
#
# Use LDAP server
ldap_server   "ldap1.domain.org,ldap2.domain.org:389"
#
# List CA-signed certificate
cert_root "C=US, O=ExamplePKI Inc, OU=PKI-Example, CN=Example PKI"
#
# Rule for off-site systems with root certificate
{
  label "Off-site mobile1 with root certificate"
  local_id_type DNS
  local_id "mobile1.domain.org"
  local_addr 0.0.0.0/0

# Find main1 and trust the root certificate
  remote_id "main1.domain.org"
  remote_addr 192.168.0.100
```

```
p2_pfs 5

p1_xform
{auth_method rsa_sig oakley_group 5 encr_alg blowfish auth_alg sha1 }
}
```

例 23–9 移動体システムからの自己署名付き証明書の受け入れ

次の例では、自己署名付き証明書が発行されており、移動体システムと中央システムに格納されています。main1 は、遠隔地のシステムからの接続を受け入れることができる会社のシステムです。オフサイトシステムを設定する方法については、例 23–10 を参照してください。

```
## /etc/hosts on main1
main1 192.168.0.100

## /etc/inet/ipsecinit.conf on main1
# Keep everyone out unless they use this IPsec policy:
{} ipsec {encr_algs aes encr_auth_algs sha1 sa shared}

## /etc/inet/ike/ike.config on main1
# Global parameters
#
# Self-signed certificates - trust me and enumerated others
cert_trust    "DNS=main1.domain.org"
cert_trust    "jdoe@domain.org"
cert_trust    "user2@domain.org"
cert_trust    "user3@domain.org"
#
# Rule for off-site systems with trusted certificate
{
  label "Off-site systems with trusted certificates"
  local_id_type DNS
  local_id "main1.domain.org"
  local_addr 192.168.0.100

# Trust the self-signed certificates
# so allow any remote_id and any remote IP address.
  remote_id ""
  remote_addr 0.0.0.0/0

p2_pfs 5

p1_xform
{auth_method rsa_sig oakley_group 5 encr_alg blowfish auth_alg sha1 }
}
```

例 23–10 自己署名付き証明書による中央システムとの接続

次の例では、mobile1 は家から会社の本社に接続しています。自己署名付き証明書が発行されており、移動体システムと中央システムに格納されています。ISP ネットワークは NAT ボックスを使用しているため、ISP は mobile1 に非公開アドレスを割り

当てることができます。NAT ボックスは、非公開アドレスを公開 IP アドレスに変換します。この公開アドレスは、ほかの ISP ネットワークノードと共有されます。企業の本社は NAT を越えません。企業の本社のコンピュータを設定する方法については、例 23–9 を参照してください。

```
## /etc/hosts on mobile1
mobile1 10.1.3.3
main1 192.168.0.100

## /etc/inet/ipsecinit.conf on mobile1
# Find main1
{raddr 192.168.0.100} ipsec {encr_algs aes encr_auth_algs sha1 sa shared}

## /etc/inet/ike/ike.config on mobile1
# Global parameters

# Self-signed certificates - trust me and the central system
cert_trust    "jdoe@domain.org"
cert_trust    "DNS=main1.domain.org"
#
# Rule for off-site systems with trusted certificate
{
  label "Off-site mobile1 with trusted certificate"
  local_id_type email
  local_id "jdoe@domain.org"
  local_addr 0.0.0.0/0

# Find main1 and trust the certificate
  remote_id "main1.domain.org"
  remote_addr 192.168.0.100

p2_pfs 5

p1_xform
{auth_method rsa_sig oakley_group 5 encr_alg blowfish auth_alg sha1 }
}
```

接続したハードウェアを検出するための IKE の構成 (タスクマップ)

次の表に、接続したハードウェアを IKE に伝える手順を示します。IKE がハードウェアを使用できるようにするには、接続したハードウェアを IKE に伝える必要があります。ハードウェアを使用する手順については、615 ページの「公開鍵証明書による IKE の構成」を参照してください。

注 – オンチップハードウェアについては、IKE に通知する必要はありません。たとえば、UltraSPARC T2 プロセッサには暗号化促進機能があります。オンチップアクセラレータを検出するよう IKE を構成する必要はありません。

タスク	説明	参照先
IKE キーの操作を Sun Crypto Accelerator 1000 ボードで行います	IKE を PKCS #11 ライブラリにリンクする	641 ページの「Sun Crypto Accelerator 1000 ボードを検出するように IKE を構成する方法」
IKE キーの操作とキーの格納を Sun Crypto Accelerator 4000 ボードで行います	IKE を PKCS #11 ライブラリにリンクして、接続されたハードウェアの名前のリストを表示する	642 ページの「Sun Crypto Accelerator 4000 ボードを検出するように IKE を構成する方法」

接続したハードウェアを検出するように IKE を構成する

公開鍵証明書は、接続されたハードウェアに格納することもできます。Sun Crypto Accelerator 1000 ボードが提供するのはストレージのみです。Sun Crypto Accelerator 4000 および Sun Crypto Accelerator 6000 ボードによってストレージが提供され、公開鍵の操作をシステムからこのボードにオフロードできます。

▼ Sun Crypto Accelerator 1000 ボードを検出するように IKE を構成する方法

始める前に　次の手順では、Sun Crypto Accelerator 1000 ボードがシステムに接続されていると仮定します。さらに、ボードに必要なソフトウェアがすでにインストールされ、構成されているものとします。手順については、Sun Crypto Accelerator 1000 Board Version 2.0 のインストールおよびユーザーズガイド (`http://download.oracle.com/docs/cd/E19412-01/819-0425-11/819-0425-11.pdf`)を参照してください。

1. システムコンソール上で、**Primary Administrator** の役割を引き受けるか、スーパーユーザーになります。

   Primary Administrator 役割には、Primary Administrator プロファイルが含まれます。役割を作成してユーザーに役割を割り当てるには、『Oracle Solaris の管理: 基本管理』の第 2 章「Solaris 管理コンソールの操作 (タスク)」を参照してください。

   注 – リモートログインすると、セキュリティー上重要なトラフィックが盗聴される恐れがあります。何らかの方法でリモートログインを保護していても、システムのセキュリティーがリモートログインセッションレベルに低下します。セキュリティー保護されたリモートログインには、`ssh` コマンドを使用してください。

   ---

2. **PKCS #11** ライブラリがリンクされていることを確認します。

   PKCS #11 ライブラリがリンクされていることを確認するには、次のコマンドを実行します。

```
# ikeadm get stats
Phase 1 SA counts:
Current:   initiator:          0   responder:          0
```

```
Total:     initiator:          0   responder:          0
Attempted: initiator:          0   responder:          0
Failed:    initiator:          0   responder:          0
           initiator fails include 0 time-out(s)
PKCS#11 library linked in from /usr/lib/libpkcs11.so
#
```

3 **Solaris 10 1/06:** このリリース以降では、ソフトトークンキーストアにキーを格納できます。

暗号化フレームワークが提供するキーストアについては、cryptoadm(1M) のマニュアルページを参照してください。キーストアを使用する例については、Example 23–11 を参照してください。

▼ Sun Crypto Accelerator 4000 ボードを検出するように IKE を構成する方法

始める前に 次の手順では、Sun Crypto Accelerator 4000 ボードがシステムに接続されていると仮定します。さらに、ボードに必要なソフトウェアがすでにインストールされ、構成されているものとします。手順については、Sun Crypto Accelerator 4000 Board Version 1.1 のインストールおよびユーザーズガイド (http://download.oracle.com/docs/cd/E19877-01/817-3693-10/817-3693-10.pdf)を参照してください。

1 システムコンソール上で、**Primary Administrator** の役割を引き受けるか、スーパーユーザーになります。

Primary Administrator 役割には、Primary Administrator プロファイルが含まれます。役割を作成してユーザーに役割を割り当てるには、『Oracle Solaris の管理: 基本管理』の第 2 章「Solaris 管理コンソールの操作 (タスク)」を参照してください。

注 – リモートログインすると、セキュリティー上重要なトラフィックが盗聴される恐れがあります。何らかの方法でリモートログインを保護していても、システムのセキュリティーがリモートログインセッションレベルに低下します。セキュリティー保護されたリモートログインには、ssh コマンドを使用してください。

---

2 **PKCS #11** ライブラリがリンクされていることを確認します。

IKE はライブラリのルーチンを使用して、Sun Crypto Accelerator 4000 ボード上でキーの生成および格納処理を行います。PKCS #11 ライブラリがリンクされていることを確認するには、次のコマンドを実行します。

```
$ ikeadm get stats
...
PKCS#11 library linked in from /usr/lib/libpkcs11.so
$
```

注 – Sun Crypto Accelerator 4000 ボードは、RSA で最大 2048 ビットのキーをサポートします。DSA の場合、このボードは最大 1024 ビットの鍵をサポートします。

3 接続された **Sun Crypto Accelerator 4000** ボードのトークン **ID** を見つけます。

```
$ ikecert tokens
Available tokens with library "/usr/lib/libpkcs11.so":

"Sun Metaslot                          "
```

ライブラリは、32 文字のトークン ID (キーストア名 とも呼ぶ) を戻します。この例では、`ikecert` コマンドに `Sun Metaslot` トークンを使用すると、IKE 鍵を格納および高速化できます。

トークンを使用する手順については、627 ページの「ハードウェアで公開鍵証明書を生成および格納する方法」を参照してください。

`ikecert` コマンドにより、後続スペースが自動的に付加されます。

例 23–11 メタスロットトークンの検索と使用

トークンは、ディスク、接続されたボード、または暗号化フレームワークが提供するソフトトークンキーストアに格納できます。次に、ソフトトークンキーストアのトークン ID の例を示します。

```
$ ikecert tokens
Available tokens with library "/usr/lib/libpkcs11.so":

"Sun Metaslot                          "
```

ソフトトークンキーストアのパスフレーズを作成する方法については、`pktool(1)` のマニュアルページを参照してください。

次に、ソフトトークンキーストアに証明書を追加するコマンドの例を示します。`Sun.Metaslot.cert` は、CA 証明書を格納しているファイルです。

```
# ikecert certdb -a -T "Sun Metaslot" < Sun.Metaslot.cert
Enter PIN for PKCS#11 token:      Type user:passphrase
```

▼ Sun Crypto Accelerator 6000 ボードを検出するように IKE を構成する方法

始める前に 次の手順では、Sun Crypto Accelerator 6000 ボードがシステムに接続されていると仮定します。さらに、ボードに必要なソフトウェアがすでにインストールされ、構成されているものとします。手順については、Sun Crypto Accelerator 6000 Board Version 1.1 User's Guide (`http://download.oracle.com/docs/cd/E19321-01/820-4144-12/820-4144-12.pdf`)を参照してください。

1 システムコンソール上で、**Primary Administrator** の役割を引き受けるか、スーパーユーザーになります。

Primary Administrator 役割には、Primary Administrator プロファイルが含まれます。役割を作成してユーザーに役割を割り当てるには、『Oracle Solaris の管理: 基本管理』の第 2 章「Solaris 管理コンソールの操作 (タスク)」を参照してください。

注 – リモートログインすると、セキュリティー上重要なトラフィックが盗聴される恐れがあります。何らかの方法でリモートログインを保護していても、システムのセキュリティーがリモートログインセッションレベルに低下します。セキュリティー保護されたリモートログインには、`ssh` コマンドを使用してください。

---

2 **PKCS #11** ライブラリがリンクされていることを確認します。

IKE はライブラリのルーチンを使用して、Sun Crypto Accelerator 6000 ボード上で鍵の生成および鍵の格納を処理します。PKCS #11 ライブラリがリンクされていることを確認するには、次のコマンドを実行します。

```
$ ikeadm get stats
...
PKCS#11 library linked in from /usr/lib/libpkcs11.so
$
```

3 接続された **Sun Crypto Accelerator 6000** ボードのトークン **ID** を見つけます。

```
$ ikecert tokens
Available tokens with library "/usr/lib/libpkcs11.so":

"Sun Metaslot                    "
```

ライブラリは、32 文字のトークン ID (キーストア名 とも呼ぶ) を戻します。この例では、`ikecert` コマンドに `Sun Metaslot` トークンを使用すると、IKE 鍵を格納および高速化できます。

トークンを使用する手順については、627 ページの「ハードウェアで公開鍵証明書を生成および格納する方法」を参照してください。

`ikecert` コマンドにより、後続スペースが自動的に付加されます。

例 23–12 メタスロットトークンの検索と使用

トークンは、ディスク、接続されたボード、または暗号化フレームワークが提供するソフトトークンキーストアに格納できます。次に、ソフトトークンキーストアのトークン ID の例を示します。

```
$ ikecert tokens
Available tokens with library "/usr/lib/libpkcs11.so":

"Sun Metaslot                    "
```

ソフトトークンキーストアのパスフレーズを作成する方法については、`pktool(1)` のマニュアルページを参照してください。

次に、ソフトトークンキーストアに証明書を追加するコマンドの例を示します。Sun.Metaslot.cert は、CA 証明書を格納しているファイルです。

```
# ikecert certdb -a -T "Sun Metaslot" < Sun.Metaslot.cert
Enter PIN for PKCS#11 token:       Type user:passphrase
```

IKE 転送パラメータの変更 (タスクマップ)

次の表に、IKE 用の転送パラメータを構成する手順を示します。

タスク	説明	参照先
鍵ネゴシエーションをより効率的にします。	鍵ネゴシエーションパラメータを変更します。	646 ページの「フェーズ 1 IKE 鍵ネゴシエーションの持続時間を変更する方法」
転送で遅延を許可するように鍵ネゴシエーションを構成します。	鍵ネゴシエーションパラメータを長くします。	例 23–13
すばやく成功したり、すばやく障害を見つけるように鍵ネゴシエーションを構成します。	鍵ネゴシエーションパラメータを短くします。	例 23–14

IKE 転送パラメータの変更

IKE が鍵ネゴシエーションを行うとき、転送速度がネゴシエーションの成功に影響します。通常、IKE 転送パラメータのデフォルト値を変更する必要はありません。しかし、非常に悪い回線で鍵ネゴシエーションを最適化したり、問題を再現したりするときには、転送パラメータの値を変更してもかまいません。

持続時間を長くすると、信頼性の低い転送回線で鍵ネゴシエーションを行うことができます。初期試行が成功するためには、特定のパラメータを長くします。初期試行が成功しない場合、後続の試行の間を空けることによって、ネゴシエーション全体が成功する時間を提供できます。

持続時間を短くすると、信頼性の高い転送回線で鍵ネゴシエーションを行うことができます。 持続時間が短くなると、ネゴシエーションが失敗したときに素早く再試行するため、ネゴシエーション全体の速度が上がります。問題を診断するときにも、ネゴシエーションの速度を上げると、障害をすばやく再現できます。持続時間を短くすると、フェーズ 1 SA が自分の寿命に使用できるようになります。

▼ フェーズ 1 IKE 鍵ネゴシエーションの持続時間を変更する方法

1 システムコンソール上で、**Primary Administrator** の役割を引き受けるか、スーパーユーザーになります。

Primary Administrator 役割には、Primary Administrator プロファイルが含まれます。役割を作成してユーザーに役割を割り当てるには、『Oracle Solaris の管理: 基本管理』の第 2 章「Solaris 管理コンソールの操作 (タスク)」を参照してください。

注 – リモートログインすると、セキュリティー上重要なトラフィックが盗聴される恐れがあります。何らかの方法でリモートログインを保護していても、システムのセキュリティーがリモートログインセッションレベルに低下します。セキュリティー保護されたリモートログインには、`ssh` コマンドを使用してください。

---

2 各システムのグローバル転送パラメータのデフォルト値を変更します。

システムごとに、`/etc/inet/ike/config` ファイルのフェーズ 1 持続時間パラメータを変更します。

```
### ike/config file on      system

## Global parameters
#
## Phase 1 transform defaults
#
#expire_timer      300
#retry_limit         5
#retry_timer_init    0.5 (integer or float)
#retry_timer_max    30   (integer or float)
```

`expire_timer` 完了していない IKE フェーズ 1 ネゴシエーションを残しておく秒数。この時間が過ぎると、ネゴシエーションの試行は削除されます。デフォルトは 30 秒です。

`retry_limit` 再転送の最大数。この回数を過ぎると、IKE ネゴシエーションは中断されます。デフォルトは 5 回です。

`retry_timer_init` 再転送の間隔の初期値。`retry_timer_max` 値に到達するまで、再転送ごとに、その間隔は 2 倍にされます。デフォルトは 0.5 秒です。

`retry_timer_max` 再転送の間隔の最大値。再転送の間隔は、この値より大きくはなりません。デフォルトは 30 秒です。

3 変更した構成をカーネルに読み込みます。

- **Solaris 10 4/09** リリース以降では、**ike** サービスをリフレッシュします。

  ```
  # svcadm refresh svc:/network/ipsec/ike
  ```

- **Solaris 10 4/09** リリースより前のリリースを実行している場合は、システムをリブートします。

  ```
  # init 6
  ```

  あるいは、in.iked デーモンを停止および起動します。

例 23–13 IKE フェーズ 1 ネゴシエーションの時間を長くする

次の例では、システムと IKE ピアはトラフィックが多い転送回線で接続されています。オリジナルの設定は、ファイルのコメント行にあります。新しい設定は、ネゴシエーションの時間を長くしています。

```
### ike/config file on partym
## Global Parameters
#
## Phase 1 transform defaults
#expire_timer   300
#retry_limit      5
#retry_timer_init 0.5 (integer or float)
#retry_timer_max 30   (integer or float)
#
expire_timer  600
retry_limit  10
retry_timer_init  2.5
retry_timer_max  180
```

例 23–14 IKE フェーズ 1 ネゴシエーションの時間を短くする

次の例では、システムと IKE ピアはトラフィックが少ない高速回線で接続されています。オリジナルの設定は、ファイルのコメント行にあります。新しい設定は、ネゴシエーションの時間を短くしています。

```
### ike/config file on partym
## Global Parameters
#
## Phase 1 transform defaults
#expire_timer   300
#retry_limit      5
#retry_timer_init 0.5 (integer or float)
#retry_timer_max 30   (integer or float)
#
expire_timer  120
retry_timer_init  0.20
```

◆ ◆ ◆ 第 24 章

インターネット鍵交換 (リファレンス)

この章では、IKE に関する次のリファレンス情報について説明します。

IKE の実装方法については、第 23 章「IKE の構成 (タスク)」を参照してください。概要については、第 22 章「インターネット鍵交換 (概要)」を参照してください。

IKE サービス

svc:/network/ipsec/ike:default サービス –サービス管理機能 (SMF) では、IKE を管理するための ike サービスが提供されています。デフォルトでは、このサービスは無効になっています。このサービスを有効にする前に、IKE 構成ファイル /etc/inet/ike/config を作成する必要があります。

次の ike サービスのプロパティーは構成可能です。

- config_file プロパティー – IKE 構成ファイルの場所です。初期値は /etc/inet/ike/config です。
- debug_level プロパティー – in.iked デーモンのデバッグレベルです。初期値は op (動作)です。指定可能な値については、ikeadm(1M) のマニュアルページの「オブジェクトタイプ」に記載されているデバッグレベルの表を参照してください。
- admin_privilege プロパティー – in.iked デーモンの特権レベルです。初期値は base です。ほかの値は modkeys と keymat です。詳細は、651 ページの「ikeadm コマンド」を参照してください。

SMF の詳細については、『Oracle Solaris の管理: 基本管理』の第 18 章「サービスの管理 (概要)」を参照してください。smf(5)、svcadm(1M)、および svccfg(1M) のマニュアルページも参照してください。

IKE デーモン

in.iked デーモンは、Oracle Solaris システム上で IPsec の暗号化キーの管理を自動化します。また、同じプロトコルを実行するリモートシステムとのネゴシエーションを行い、認証された鍵情報が、保護された方法でセキュリティーアソシエーション (SA) に提供されます。そのデーモンは、セキュリティー保護された通信を行うすべてのシステムで実行する必要があります。

デフォルトでは、svc:/network/ipsec/ike:default サービスは有効になっていません。/etc/inet/ike/config ファイルを構成し、ike サービスを有効にしたら、システムブート時に in.iked デーモンが実行されます。

IKE デーモンを実行すると、システムは、フェーズ 1 交換でそのピア IKE エンティティーに対してそのシステム自体を認証します。そのピアは、認証方式として IKE ポリシーファイルに定義されています。そのあと、デーモンはフェーズ 2 のキーが設定します。ポリシーファイルで指定した時間間隔で、IKE キーが自動的にリフレッシュされます。in.iked デーモンを実行すると、ネットワークからの着信 IKE 要求と PF_KEY ソケット経由のアウトバウンドトラフィックの要求を待機します。詳細は、pf_key(7P) のマニュアルページを参照してください。

2 つのコマンドが IKE デーモンをサポートします。ikeadm コマンドを使用すると、IKE ポリシーの表示および一時的な変更を行うことができます。IKE ポリシーを永続的に変更するには、ike サービスのプロパティーを変更する必要があります。IKE サービスのプロパティーを変更するには、546 ページの「IKE および IPsec サービスを管理する方法」を参照してください。

ikecert コマンドを実行すると、公開鍵データベースを表示および変更できます。このコマンドでは、ローカルデータベース ike.privatekeys と publickeys を管理します。公開鍵の操作とハードウェア上の公開鍵のストレージも管理します。

IKE 構成ファイル

IKE 構成ファイル /etc/inet/ike/config は、IPsec ポリシーファイル /etc/inet/ipsecinit.conf の中で保護されているインタフェースの鍵を管理します。

IKE での鍵管理には、ルールとグローバルパラメータが関係します。IKE ルールは、その鍵情報で保護するシステムやネットワークを識別します。さらに、ルールは認証方式も指定します。グローバルパラメータには、接続されたハードウェアアクセラレータへのパスなどがあります。IKE ポリシーファイルの例については、

604 ページの「事前共有鍵による IKE の構成 (タスクマップ)」を参照してください。IKE ポリシーエントリの例と説明については、ike.config(4) のマニュアルページを参照してください。

IKE がサポートする IPsec SA は、IPsec 構成ファイル /etc/inet/ipsecinit.conf のポリシーに従って IP データグラムを保護します。IPsec SA の作成時に perfect forward security (PFS) を使用するかどうかは、IKE ポリシーファイルで決まります。

/etc/inet/ike/config ファイルには、RSA Security Inc. PKCS #11 Cryptographic Token Interface (Cryptoki) に準拠して実装されているライブラリへのパスを含めることができます。IKE は、この PKCS #11 ライブラリを使って、アクセラレータおよび鍵ストレージハードウェアにアクセスします。

ike/config ファイルのセキュリティーに関する注意点は、ipsecinit.conf ファイルのセキュリティーと同様です。詳細は、588 ページの「ipsecinit.conf と ipsecconf のセキュリティーについて」を参照してください。

ikeadm コマンド

ikeadm コマンドを実行すると、次のことができます。

- IKE デーモンプロセスの要素の表示
- IKE デーモンに渡すパラメータの変更
- フェーズ 1 交換時の SA 作成に関する統計情報の表示
- IKE プロセスのデバッグ
- IKE 状態の要素の表示
- IKE デーモンのプロパティーの変更
- フェーズ 1 交換時の SA 作成に関する統計情報の表示
- IKE プロトコル交換のデバッグ

このコマンドのオプションの例と詳しい説明については、ikeadm(1M) のマニュアルページを参照してください。

実行する IKE デーモンの特権レベルにより、表示および変更可能な IKE デーモンの要素が決まります。使用可能な特権レベルは 3 つあります。

base レベル	鍵情報を表示したり変更したりすることはできません。base レベルはデフォルトの特権レベルです。
modkeys レベル	事前共有鍵の削除、変更、追加ができます。
keymat レベル	ikeadm コマンドで実際の鍵情報を表示できます。

特権を一時的に変更する場合は、ikeadm コマンドを使用できます。永続的に変更する場合は、ike サービスの admin_privilege プロパティーを変更します。手順については、546 ページの「IKE および IPsec サービスを管理する方法」を参照してください。

ikeadm コマンドのセキュリティーについては、ipseckey コマンドのセキュリティーと同様です。詳細は、590 ページの「ipseckey におけるセキュリティーについて」を参照してください。

IKE 事前共有鍵ファイル

事前共有鍵を手動で作成すると、鍵は、/etc/inet/secret ディレクトリのファイルに格納されます。ike.preshared ファイルに Internet Security Association and Key Management Protocol (ISAKMP) SA の事前共有鍵が含まれ、ipseckeys ファイルに IPsec SA の事前共有鍵が含まれます。これらのファイルは 0600 で保護されます。secret ディレクトリは 0700 で保護されます。

- ike.preshared ファイルは、事前共有鍵を必要とする ike/config ファイルの構成時に作成します。IKE 認証用として、ike.preshared ファイルに ISAKMP SA の鍵情報を入力します。フェーズ 1 交換の認証に事前共有鍵を使用するため、このファイルを in.iked デーモンの開始前に有効にする必要があります。
- ipseckeys ファイルには、IPsec SA の鍵情報が含まれています。手動によるファイルの管理の例については、538 ページの「IPsec セキュリティーアソシエーションを手動で作成する方法」を参照してください。IKE デーモンでは、このファイルを使用しません。IKE によって IPsec SA に対して生成される鍵情報は、カーネルに保存されます。

IKE 公開鍵のデータベースおよびコマンド

ikecert コマンドを実行すると、ローカルシステムの公開鍵データベースを操作できます。このコマンドは、ike/config ファイルが公開鍵証明書を要求するときに使用します。IKE ではそれらのデータベースを使用してフェーズ 1 交換を認証するため、in.iked デーモンを起動する前に、それらのデータベースに必要な情報が含まれていなければなりません。3 つのサブコマンド certlocal、certdb、certrldb をそれぞれ実行して、3 つのデータベースを処理します。

ikecert コマンドは鍵の格納処理も行います。鍵は、ディスク、接続された Sun Crypto Accelerator 6000 または Sun Crypto Accelerator 4000 ボード、またはソフトトークンキーストアに格納できます。ソフトトークンキーストアは、ハードウェアデバイスと通信するために、暗号化フレームワークのメタスロットを使用しているときに使用できます。ikecert コマンドは、PKCS #11 ライブラリを使用して鍵の格納場所を見つけます。

- **Solaris 10 1/06**: このリリース以降では、このライブラリを指定する必要はありません。デフォルトでは、PKCS #11 ライブラリは /usr/lib/libpkcs11.so です。

- **Solaris 10**: このリリースでは、PKCS #11 エントリを指定する必要があります。それ以外の場合、ikecert コマンドの -T オプションは機能しません。エントリは次のようになります。

```
pkcs11_path "/opt/SUNWconn/cryptov2/lib/libvpkcs11.so"
```

詳細は、ikecert(1M) のマニュアルページを参照してください。メタスロットとソフトトークンキーストアについては、cryptoadm(1M) のマニュアルページを参照してください。

ikecert tokens コマンド

tokens 引数を使用すると、使用可能なトークン ID がリストされます。トークン ID により、ikecert certlocal コマンドと ikecert certdb コマンドは、公開鍵証明書と証明書要求を生成します。証明書と証明書リクエストも、暗号化フレームワークによってソフトトークンキーストアに格納するか、接続された Sun Crypto Accelerator 6000 または Sun Crypto Accelerator 4000 ボードに格納することができます。ikecert コマンドは、PKCS #11 ライブラリを使用して証明書の格納場所を見つけます。

ikecert certlocal コマンド

certlocal サブコマンドは非公開鍵データベースを管理します。このサブコマンドを選択すると、非公開鍵の追加、表示、および削除を行うことができます。また、自己署名付き証明書または証明書要求のいずれかを作成できます。-ks オプションを選択すると、自己署名付き証明書が作成されます。-kc オプションを選択すると、証明書要求が作成されます。鍵はシステムの /etc/inet/secret/ike.privatekeys ディレクトリに格納されます。-T オプションを指定した場合は、システムに接続されたハードウェアに格納されます。

非公開鍵を作成する場合は、ikecert certlocal コマンドへのサブコマンドに関連するエントリが ike/config ファイルに存在しなければなりません。ikecert オプションと ike/config エントリの対応を次の表に示します。

表 24–1 ikecert オプションと ike/config エントリの対応表

ikecert オプション	ike/config エントリ	説明
-A *subject-alternate-name*	cert_trust *subject-alternate-name*	証明書を一意に識別するニックネーム。指定可能な値は IP アドレス、電子メールアドレス、およびドメイン名です。
-D *X.509-distinguished-name*	*X.509-distinguished-name*	国 (C)、組織名 (ON)、組織単位 (OU)、共通名 (CN) を含む認証局のフルネーム。

表 24–1 ikecert オプションと ike/config エントリの対応表 (続き)

ikecert オプション	ike/config エントリ	説明
-t dsa-sha1	auth_method dsa_sig	RSAよりもわずかに遅い認証方式。
-t rsa-md5 および -t rsa-sha1	auth_method rsa_sig	DSAよりもわずかに速い認証方式。 RSA 公開鍵は、最大ペイロードを暗号化するのに十分な長さが必要。通常、X.509 識別名などの ID ペイロードが最大ペイロードになります。
-t rsa-md5 および -t rsa-sha1	auth_method rsa_encrypt	RSA 暗号化により、IKE にある ID が不正侵入者から保護されますが、IKE ピアには互いの公開鍵の認識が要求されます。
-T	pkcs11_path	PKCS #11 ライブラリは、Sun Crypto Accelerator 1000 ボード、Sun Crypto Accelerator 6000 ボード、および Sun Crypto Accelerator 4000 ボード上の鍵アクセラレータを処理します。また、このライブラリは、Sun Crypto Accelerator 6000 および Sun Crypto Accelerator 4000 ボード上の鍵ストレージを処理するトークンも提供します。

ikecert certlocal -kc コマンドを指定して証明書要求を実行する場合、そのコマンド出力を PKI 機関または認証局 (CA) に送信します。会社が独自の PKI を運営している場合は、出力を PKI 管理者に送信します。PKI 機関、CA、または PKI の管理者はこれに基づいて証明書を作成します。PKI または CA から返された証明書は、certdb サブコマンドに渡されます。PKI から返された証明書失効リスト (CRL) は、certrldb サブコマンドに渡されます。

ikecert certdb コマンド

certdb サブコマンドは、公開鍵データベースを管理します。そのサブコマンドを選択すると、公開鍵と証明書を追加、表示、および削除できます。また、リモートシステムで ikecert certlocal -ks コマンドを実行して作成された証明書を入力として受け入れます。手順については、616 ページの「自己署名付き公開鍵証明書により IKE を構成する方法」を参照してください。さらに、PKI または CA から受信する証明書も入力として受け入れます。手順については、621 ページの「CA からの署名付き証明書により IKE を構成する方法」を参照してください。

証明書と公開鍵は、システムの /etc/inet/ike/publickeys ディレクトリに格納されます。-T オプションを指定した場合、証明書、非公開鍵、公開鍵は、システムに接続されたハードウェアに格納されます。

ikecert certrldb コマンド

certrldb サブコマンドは、証明書失効リスト (CRL) データベース /etc/inet/ike/crls を管理します。CRL データベースには、公開鍵の失効リストが保存されています。よって、このリストには、すでに有効でない証明書が明記されます。PKI によって CRL が提供されるときに、ikecert certrldb コマンドを指定して CRL データベースにその CRL を格納します。手順については、631 ページの「証明書失効リストを処理する方法」を参照してください。

/etc/inet/ike/publickeys ディレクトリ

/etc/inet/ike/publickeys ディレクトリの複数のファイルまたは「スロット」には、公開鍵と非公開鍵のペアの公開部分とその証明書が含まれています。このディレクトリは 0755 で保護されています。ikecert certdb コマンドを使用して、そのディレクトリを読み込みます。-T オプションは、鍵を publickeys ディレクトリではなく Sun Crypto Accelerator 6000 または Sun Crypto Accelerator 4000 ボード上に格納します。

スロットには、別のシステムで生成された証明書の X.509 識別名がエンコードされた形式で含まれます。自己署名付き証明書を使用する場合、そのコマンドへの入力として、リモートシステムの管理者から受信する証明書を使用します。CA からの証明書を使用する場合、CA から受け取る 2 つの署名付き証明書をこのデータベースに格納します。CA に送信した証明書署名要求に基づいた証明書を格納します。また、CA の証明書も格納します。

/etc/inet/secret/ike.privatekeys ディレクトリ

/etc/inet/secret/ike.privatekeys ディレクトリには、公開非公開鍵ペアの一部である非公開鍵ファイルが格納されています。このディレクトリは 0700 で保護されています。ikecert certlocal コマンドを実行して、ike.privatekeys ディレクトリを読み込みます。非公開鍵は、ペアとなる公開鍵、自己署名付き証明書や CA が格納されてから有効になります。ペアとなる公開鍵は、/etc/inet/ike/publickeys ディレクトリか、Sun Crypto Accelerator 6000 または Sun Crypto Accelerator 4000 ボードに格納されます。

/etc/inet/ike/crls ディレクトリ

/etc/inet/ike/crls ディレクトリには、証明書失効リスト (CRL) ファイルが含まれています。各ファイルは、/etc/inet/ike/publickeys ディレクトリにある公開鍵証明書ファイルに対応しています。PKI 機関により、それらの証明書の CRL が提供されます。ikecert certrldb コマンドを使用して、そのデータベースを読み込むことができます。

◆ ◆ ◆ 第 25 章

Oracle Solaris の IP フィルタ (概要)

この章では、Oracle Solaris の機能の 1 つである IP フィルタの概要を説明します。IP フィルタを使用したタスクについては、第 26 章「IP フィルタ (タスク)」を参照してください。

この章では、次の内容について説明します。

IP フィルタの新機能

この節では、IP フィルタの新しい機能について説明します。

新機能の完全な一覧や各 Oracle Solaris リリースの説明については、『Oracle Solaris 10 1/13 の新機能』を参照してください

パケットフィルタリングのパケットフィルタフック

Solaris 10 7/07 リリース以降では、Oracle Solaris でのネットワークパケットのフィルタリングにパケットフィルタリングフックが使用されるようになりました。この機能は、システム管理に次のような利点をもたらします。

- パケットフィルタリングフックにより、IP フィルタの構成が簡単になります。
- ゾーン間のパケットフィルタリングがサポートされるようになりました。
- フィルタリングフックを使用すると IP フィルタのパフォーマンスが向上します。

これらのフックの詳細については、669 ページの「パケットフィルタリングフック」を参照してください。パケットフィルタリングフックに関するタスクについては、第 26 章「IP フィルタ (タスク)」を参照してください。

IP フィルタの IPv6 パケットフィルタリング

Solaris 6/06 OS: ネットワークインフラストラクチャーの全部または一部が IPv6 で構成するシステム管理者のために、IP フィルタが IPv6 パケットフィルタリングに対応できるように拡張されています。IPv6 パケットフィルタリングでは、発信元または宛先の IPv6 アドレス、IPv6 アドレスを含むプール、および IPv6 拡張ヘッダーに基づいて、パケットを取り出すことができます。

`ipf` コマンドと `ipfstat` コマンドを IPv6 に使用できるように、`-6` オプションが追加されています。`ipmon` コマンドと `ippool` コマンドのコマンド行インタフェースに変更はありませんが、これらのコマンドは IPv6 もサポートしています。`ipmon` コマンドは、IPv6 パケットのログを記録できるように拡張されており、`ippool` コマンドは、IPv6 アドレスのプールへの格納をサポートしています。

詳細は、「IPv6 と IP フィルタ」を参照してください。IPv6 パケットフィルタリングに関するタスクについては、第 26 章「IP フィルタ (タスク)」を参照してください。

さらに、IP フィルタのネットワークアドレス変換 (NAT) 機能での IPv6 をサポートしています。NAT については、667 ページの「IP フィルタの NAT 機能の使用」を参照してください。

IP フィルタとは

Oracle Solaris の IP フィルタ機能は、OS のSunScreen ファイアウォールの代替となるものです。SunScreen ファイアウォール同様、IP フィルタは、ステートフルパケットフィルタリングとネットワークアドレス変換 (NAT) を行います。IP フィルタには、ステートレスパケットフィルタリングと、アドレスプールの作成および管理を行う機能もあります。

パケットのフィルタリングは、ネットワークベースの攻撃に対する基本的な保護を提供します。IP フィルタは、IP アドレス、ポート、プロトコル、ネットワークインタフェース、およびトラフィックの転送方向によって、フィルタリングを行うことができます。また、発信元 IP アドレス、宛先 IP アドレス、IP アドレスの範囲、またはアドレスプールによってもフィルタリングを行うことができます。

IP フィルタは、オープンソースの IP フィルタソフトウェアから派生したものです。オープンソースの IP フィルタのライセンス契約の条件、作者、および著作権宣言文を参照するためのデフォルトパスは、`/usr/lib/ipf/IPFILTER.LICENCE` です。Oracle Solaris がデフォルト以外の場所にインストールされている場合は、所定のパスを修正して、インストールした場所にあるファイルにアクセスします。

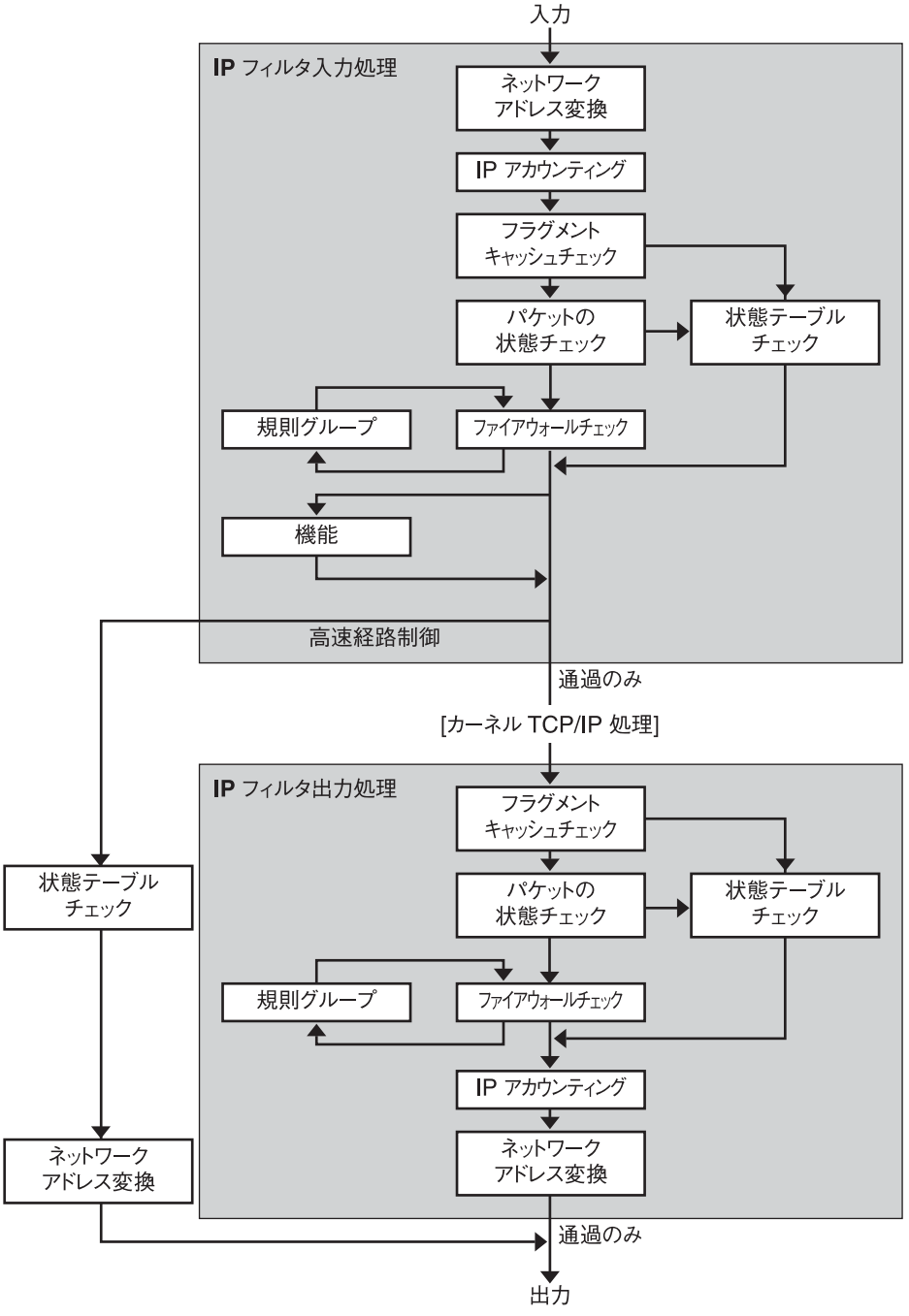
オープンソースの IP フィルタの情報ソース

Darren Reed によるオープンソースの IP フィルタソフトウェアのホームページは、`http://coombs.anu.edu.au/~avalon/ip-filter.html` にあります。このサイトには、チュートリアル「IP Filter Based Firewalls HOWTO」(Brendan Conoboy および Erik Fichtner 著、2002 年) へのリンクなど、オープンソースの IP フィルタに関する情報が含まれています。このチュートリアルは、BSD UNIX 環境でファイアウォールを作成する方法を手順ごとに説明しています。このチュートリアルは BSD UNIX 環境向けに書かれていますが、IP フィルタの構成にも関連しています。

IP フィルタのパケット処理

IP フィルタは、パケットが処理されるときに一連の手順を実行します。次の図は、パケット処理の段階と、フィルタが TCP/IP プロトコルスタックとどのように統合されるかを示しています。

図 25–1 パケット処理の順序



パケット処理には次の手順が含まれます。

- ネットワークアドレス変換 **(NAT)**

  プライベート IP アドレスを異なる公開アドレスに変換するか、複数のプライベートアドレスの別名として単一の公開アドレスを使用します。NAT を使用すると、組織に既存のネットワークがあり、インターネットにアクセスする必要がある場合に、IP アドレスが枯渇する問題を解決できます。

- **IP** アカウンティング

  入力と出力の規則を個別に設定し、通過するバイト数を記録できます。規則に一致する数に達するたびに、パケットのバイト数を規則に追加し、段階的な統計を収集できます。

- フラグメントキャッシュチェック

  現在のトラフィック内の次のパケットがフラグメントの場合、前のパケットが許可されると、状態テーブルと規則のチェックが迂回され、そのフラグメントも許可されます。

- パケットの状態チェック

  規則に `keep state` が含まれている場合、指定されたセッション内のすべてのパケットは、規則で `pass` または `block` のどちらが指定されているかに応じて自動的に通されるかブロックされます。

- ファイアウォールチェック

  入力と出力の規則は個別に設定が可能で、パケットが IP フィルタを通過してカーネルの TCP/IP ルーチン内に受信したり、またはネットワーク上に送信されることを許可するかどうかを決定できます。

- グループ

  グループを使用すると、ツリー形式で規則セットを作成できます。

- 機能

  機能とは、実行されるアクションです。`block`、 `pass`、`literal`、および `send ICMP response` などの機能を実行できます。

- 高速経路制御

  高速ルートは、パケットを経路制御のための UNIX IP スタックに渡さないように IP Filter に指示し、TTL の減少を防ぎます。

- **IP** 認証

  認証されたパケットが、ファイアウォールループを 1 回だけ通過するようにして、重複した処理を防止します。

IP フィルタの使用ガイドライン

- IP フィルタは SMF サービス svc: /network/pfil と svc: /network/ipfilter によって管理されます。SMF の完全な概要については、『Oracle Solaris の管理: 基本管理』の第 18 章「サービスの管理 (概要)」を参照してください。SMF に関連するステップごとの手順については、『Oracle Solaris の管理: 基本管理』の第 19 章「サービスの管理 (タスク)」を参照してください。
- IP フィルタでは、構成ファイルを直接編集する必要があります。
- IP フィルタは Oracle Solaris の一部としてインストールされます。デフォルトでは、インストール直後は、IP フィルタはアクティブではありません。フィルタリングを構成するには、構成ファイルを編集し、手動で IP フィルタをアクティブにする必要があります。フィルタリングは、システムをリブートするか、`ifconfig` コマンドでインタフェースを plumb することによってアクティブ化できます。詳細は、`ifconfig(1M)` のマニュアルページを参照してください。IP フィルタを有効にするためのタスクについては、675 ページの「IP フィルタの構成」を参照してください。
- IP フィルタを管理するには、IP Filter Management の権利プロファイルを持つ役割またはスーパーユーザーになれなければなりません。IP Filter Management の権利プロファイルは、ユーザーが作成した役割に割り当てることができます。役割の作成と役割のユーザーへの割り当てについては、『Solaris のシステム管理: セキュリティーサービス』の「RBAC の構成 (タスクマップ)」を参照してください。
- IP ネットワークマルチパス (IPMP) は、ステートレスフィルタリングだけをサポートしています。

  IPMP グループに送受信されるトラフィックに対して IP フィルタでステートレスフィルタリングを実行する場合は、`ipmp_hook_emulation` パラメータを設定する必要があります。デフォルトでは、このパラメータはゼロ (`0`) に設定されており、これは IP フィルタが IPMP グループに属する物理インタフェースに対してトラフィックのステートフルパケットインスペクションを実行できないことを意味します。IPMP パケットフィルタリングを有効にするには、次のコマンドを発行します。

  ```
  ndd -set /dev/ip ipmp_hook_emulation 1
  ```
- Oracle Solaris クラスタソフトウェアは、スケーラブルサービス用の IP フィルタによるフィルタリングはサポートしませんが、フェイルオーバーサービス用の IP フィルタはサポートします。IP フィルタをクラスタ内で構成するときのガイドラインおよび制限については、『Oracle Solaris Cluster ソフトウェアのインストール』の「Oracle Solaris OS の機能制限」を参照してください。
- ゾーン間のフィルタリングは、IP フィルタ規則が実装されているゾーンが、システム上のほかのゾーンの仮想ルーターとして機能する場合にかぎりサポートされます。

IP フィルタの構成ファイルの使用

IP フィルタを使用すると、ファイアウォールサービスまたはネットワークアドレス変換 (NAT) が利用できるようになります。IP フィルタは、ロード可能な構成ファイルを使用して実装できます。IP フィルタには、/etc/ipf というディレクトリがあります。/etc/ipf ディレクトリには、ipf.conf、ipnat.conf および ippool.conf と呼ばれる構成ファイルを作成して、保存できます。 /etc/ipf ディレクトリに保存されている場合、これらのファイルは、ブートプロセスで自動的にロードされます。構成ファイルを別の場所に保存して、これらのファイルを手動でロードすることも可能です。構成ファイルの例については、709 ページの「IP フィルタ構成ファイルの作成と編集」を参照してください。

IP フィルタの規則セットの使用

ファイアウォールを管理するために、IP フィルタを使用して、ネットワークトラフィックをフィルタリングするための規則セットを指定します。次の種類の規則セットを作成できます。

- パケットフィルタリング規則セット
- ネットワークアドレス変換 (NAT) 規則セット

さらに、まとまった IP アドレスを参照するために、アドレスプールを作成することもできます。作成したプールは、あとで規則セット内で使用できます。アドレスプールは、規則処理を速めるために役立ちます。また、アドレスプールによって、大きなまとまりのアドレスをより簡単に管理できます。

IP フィルタのパケットのフィルタリング機能の使用

パケットフィルタリング規則セットを使用して、パケットのフィルタリングを設定します。ipf コマンドで、パケットフィルタリング規則セットを処理します。ipf コマンドの詳細については、ipf(1M) コマンドを参照してください。

パケットフィルタリング規則は、ipf コマンドによってコマンド行で作成することも、パケットフィルタリングの構成ファイル内で作成することもできます。ブート時にパケットフィルタリング規則をロードする場合は、パケットフィルタリング規則を保存する /etc/ipf/ipf.conf という構成ファイルを作成します。 ブート時にパケットフィルタリング規則をロードしない場合は、適当な場所に ipf.conf ファイルを保存し、 ipf コマンドによってパケットフィルタリングを手動でアクティブ化します。

IP フィルタには、アクティブ規則セットと非アクティブ規則セットの 2 つのパケットフィルタリング規則セットを維持管理することができます。大部分の場合、作業ではアクティブ規則セットを使用します。ただし、ipf -I コマンドを使用

すると、コマンドアクションを非アクティブ規則リストに適用できます。非アクティブ規則リストは、ユーザーが選択しない限り、IP フィルタによって使用されることはありません。非アクティブ規則リストによって、アクティブなパケットのフィルタリングに影響を与えずに、規則を保存できます。

IP フィルタは、構成された規則リストの最初から最後まで規則を処理してから、パケットの通過またはブロックを行います。IP フィルタは、パケットを通過させるかどうかを決めるフラグを維持管理しています。フラグは、規則セット全体を調べ、最後に一致した規則を基にパケットを通過させるか、ブロックするかを決定します。

このプロセスには、2 つの例外があります。最初の例外は、パケットが `quick` キーワードを含む規則に一致した場合です。規則が `quick` キーワードを含む場合は、その規則に対する処理が実行され、それ以降の規則はチェックされません。2 番目の例外は、パケットが `group` キーワードを含む規則に一致した場合です。パケットがグループに一致すると、グループでタグ付けされた規則だけがチェックされます。

パケットのフィルタリング規則の構成

パケットのフィルタリング規則を作成するには、次の構文を使用します。

action `[in|out]` *option keyword, keyword...*

1. 各規則がアクションを開始します。IP フィルタは、パケットが規則に適合する場合、アクションを実行します。次の一覧に、パケットに対して実行される一般的なアクションを示します。

   `block` パケットはフィルタを通過できません。

   `pass` パケットはフィルタを通過します。

   `log` パケットをロギングしますが、パケットをブロックするか、通過させるかの決定は行いません。ログを参照するには、`ipmon` コマンドを使用します。

   `count` フィルタの統計にパケットを含めます。統計を参照するには、`ipfstat` コマンドを使用します。

   `skip` *number* フィルタは *number* フィルタリング規則をスキップします。

   `auth` パケット情報を確認するユーザープログラムが実行するパケット認証を要求します。このプログラムは、パケットを通過させるか、ブロックするかを決定します。

2. アクション後の出力は、`in` または `out` のはずです。ユーザーの選択により、パケットのフィルタリング規則が、受信パケットと発信パケットのどちらに適用されるのかが決定されます。

3. 次に、オプションの一覧からオプションを選択します。複数のオプションを使用する場合は、次の順序で使用してください。

`log`	規則が最後に一致した規則の場合、パケットをロギングします。ログを参照するには、`ipmon` コマンドを使用します。
`quick`	パケットが一致した場合、`quick` オプションを含む規則を実行します。これ以上の規則チェックは行われません。
`on` *interface-name*	パケットが指定したインタフェースを出入りする場合だけ、規則を適用します。
`dup-to` *interface-name*	パケットをコピーし、*interface-name* 上の複製を任意で指定した IP アドレスに送信します。
`to` *interface-name*	パケットを *interface-name* のアウトバウンドキューに移動します。

4. オプションの指定後、パケットが規則に一致するかどうかを決定するさまざまなキーワードを選択できます。次のキーワードは、以下の順序で使用してください。

注 – デフォルトでは、構成ファイルのいずれの規則にも一致しないパケットは、すべてフィルタを通過します。

`tos`	16 進数または 10 進数の整数で表されたサービスタイプの値を基に、パケットをフィルタリングします。
`ttl`	生存期間の値を基に、パケットの一致を取ります。パケットに保存されている生存期間の値は、破棄される前にパケットがネットワーク上に存在できる期間を示します。
`proto`	特定のプロトコルに対して一致を取ります。`/etc/protocols` ファイルに指定されている任意のプロトコル名を使用したり、そのプロトコルを表す 10 進数の数を指定したりできます。キーワード `tcp/udp` は、TCP または UDP パケットとの一致を取るために使用できます。
`from/to/all/ any`	発信元 IP アドレス、宛先 IP アドレス およびポート番号のいずれか、 またはすべてに対して一致を取ります。`all` キーワードは、すべての発信元からのパケットおよびすべての宛先へのパケットを受諾するために使用します。

`with`	パケットに関連する指定された属性に対して一致を取ります。オプションがない場合にパケットを一致させるには、キーワードの前に `not` または `no` と記述します。
`flags`	設定されている TCP フラグを基にフィルタリングを行う TCP で使用します。TCP フラグについては、`ipf(4)` のマニュアルページを参照してください。
`icmp-type`	ICMP のタイプによってフィルタリングを行います。このキーワードは `proto` オプションが `icmp` に設定されているときに使用され、`flags` オプションが指定されているときは使用されません。
`keep` *keep-options*	保存しておくパケットの情報を決定します。使用可能な *keep-options* には、`state` オプションと `frags` オプションなどがあります。`state` オプションは、セッションに関する情報を、TCP、UDP、および ICMP パケットで保存できます。`frags` オプションは、パケットのフラグメントに関する情報を保存し、後のフラグメントにその情報を適用します。*keep-options* は、一致したパケットをアクセス制御リストのチェックなしで、通過させます。
`head` *number*	番号 *number* で指定されるフィルタリング規則に対して、新しいグループを作成します。
`group` *number*	デフォルトグループではなく、グループ番号 *number* のグループに規則を追加します。ほかのグループを指定しない場合は、すべてのフィルタリング規則がグループ 0 に保存されます。

次の例は、規則を作成するためにパケットのフィルタリング規則構文をまとめる方法を示しています。IP アドレス `192.168.0.0/16` からの受信トラフィックをブロックするには、規則リストに次の規則を含めます。

```
block in quick from 192.168.0.0/16 to any
```

パケットフィルタリング規則を記述するときの詳細な文法および構文については、`ipf(4)` のマニュアルページを参照してください。パケットのフィルタリングに関連するタスクについては、690 ページの「IP フィルタのパケットフィルタリング規則セットの管理」を参照してください。この例で示す IP アドレススキーム (`192.168.0.0/16`) の説明については、第 2 章「TCP/IP ネットワークの計画 (手順)」を参照してください。

IP フィルタの NAT 機能の使用

NAT は、発信元 IP アドレスと宛先 IP アドレスをほかのインターネットアドレスまたはイントラネットアドレスに変換するマッピング規則を設定します。これらの規則は、受信 IP パケットまたは発信 IP パケットの 発信元アドレスおよび宛先アドレスを変更し、パケットを送信します。また、NAT を使用して、あるポートから別のポートにトラフィックの方向を変更することもできます。NAT は、パケットに修正または方向の変更が行われても、パケットの完全性を維持します。

`ipnat` コマンドは、NAT 規則リストを処理するために使用します。`ipnat` コマンドの詳細については、`ipnat(1M)` コマンドを参照してください。

NAT 規則は、 `ipnat` コマンドを使用してコマンド行で作成することもNAT 構成ファイルで作成することもできます。NAT 構成規則は、`ipnat.conf` ファイルに保存されます。ネットワークアドレス変換を使用する場合は、`ipnat.conf` ファイルを作成します。 ブート時に NAT 規則をロードしない場合は、適当な場所に `ipnat.conf` ファイルを保存し、`ipnat` コマンドによってパケットフィルタリングを手動でアクティブ化します。

NAT 規則は IPv4 と IPv6 アドレス両方に適用できます。しかし、1 つの規則で両方のアドレスの種類を指定することはできません。アドレスの種類ごとに規則を別個に設定する必要があります。IPv6 アドレスを含む NAT 規則では、`mapproxy` および `rdrproxy` NAT コマンドを同時に使用することはできません。

NAT 規則の構成

次の構文で NAT 規則を作成します。

command interface-name parameters

1. 各規則の冒頭には、次のコマンドのいずれかが記述されています。

`map`	ある IP アドレスまたはネットワークを規制のないラウンドロビン方式で別の IP アドレスまたはネットワークにマッピングします。
`rdr`	ある IP アドレスとポートのペアから別の IP アドレスとポートのペアにパケットの方向を変更します。
`bimap`	外部 IP アドレスと内部 IP アドレス間で双方向の NAT を確立します。
`map-block`	静的 IP アドレスをベースにした変換を確立します。このコマンドは、アドレスを指定の範囲に変換するアルゴリズムに基づいています。

2. このコマンドのあとには、`hme0` などのインタフェース名を記述します。

3. 次に、NAT 構成を決定するさまざまなパラメータを選択します。次に、この種のパラメータの例をいくつか挙げます。

   `ipmask` ネットワークマスクを指定します。

   `dstipmask` `ipmask` が変換されるアドレスを指定します。

   `mapport` ポート番号の範囲と `tcp`、`udp` または `tcp/udp` プロトコルを指定します。

次の例は、NAT 規則を作成するために NAT 規則の構文をまとめる方法を示しています。発信元アドレスが `192.168.1.0/24` のデバイス `de0` から発信されるパケットを書き換え、外部に対して発信元アドレスが `10.1.0.0/16` であることを示すには、NAT 規則セットに次の規則を含めます。

```
map de0 192.168.1.0/24 -> 10.1.0.0/16
```

次の規則は IPv6 アドレスに適用されます。

```
map ppp0 fec0:1::/64 -> 2000:1:2::/72 portmap tcp/udp 1025:65000
map-block ppp0 fe80:0:0:209::/64 -> 209:1:2::/72 ports auto
rdr ce0 209::ffff:fe13:e43e port 80 -> fec0:1::e,fec0:1::f port 80 tcp round-robin
```

NAT 規則を記述する際の詳細な文法と形式については、`ipnat(4)` のマニュアルページを参照してください。

IP フィルタのアドレスプール機能の使用

アドレスプールは、アドレスとネットマスクのペアのまとまりに名前付けを行います。アドレスプールは、IP アドレスと規則の一致を取るために必要な時間を短縮します。また、アドレスプールによって、大きなまとまりのアドレスをより簡単に管理できます。

アドレスプール構成規則は、`ippool.conf` ファイルに保存されます。ブート時にアドレスプール規則をロードする場合は、アドレスプールの規則を保存する `/etc/ipf/ippool.conf` というファイルを作成します。 ブート時に アドレスプール規則をロードしない場合は、適当な場所に `ippool.conf` ファイルを保存し、`ippool` コマンドによってパケットフィルタリングを手動でアクティブ化します。

アドレスプールの構成

次の構文でアドレスプールを作成します。

`table role =` *role-name* `type =` *storage-format* `number =` *reference-number*

`table` 複数のアドレスへの参照を定義します。

role　IP フィルタでのプールの役割を指定します。この時点で、参照できる役割は ipf だけです。

type　プールの保存形式を指定します。

number　フィルタリング規則が使用する参照番号を指定します。

たとえば、アドレスが 10.1.1.1 および 10.1.1.2 でネットワークが 192.16.1.0 のグループをプール番号 13 で参照する場合、アドレスプールの構成ファイルに次の規則を含めます。

```
table role = ipf type = tree number = 13
{ 10.1.1.1/32, 10.1.1.2/32, 192.168.1.0/24 };
```

次に、フィルタリング規則のプール番号 13 を参照するには、次の例のような規則を構築します。

```
pass in from pool/13 to any
```

なお、プールへの参照を含む規則ファイルをロードする前に、プールファイルをロードする必要があります。プールファイルをロードしていない場合、次の出力のようにプールは未定義となります。

```
# ipfstat -io
empty list for ipfilter(out)
block in from pool/13(!) to any
```

プールをあとで追加しても、そのプールの追加によってカーネルの規則セットが更新されることはありません。そのプールを参照する規則ファイルも再ロードする必要があります。

パケットフィルタリング規則を記述するときの詳細な文法および構文については、ippool(4) のマニュアルページを参照してください。

パケットフィルタリングフック

Solaris 10 7/07 リリース以降では、pfil モジュールの代わりにパケットフィルタリングフックを使用して IP フィルタを有効にします。以前の Solaris リリースでは、IP フィルタを設定するには、追加手順として pfil モジュールの構成が必要でした。この追加の構成が必要であったため間違いが発生しやすく、IP フィルタが正しく動作しない場合がありました。pfil STREAMS モジュールが IP とデバイスドライバの間に挿入されることも、パフォーマンス低下の原因となっていました。さらに、pfil モジュールはゾーン間のパケット傍受を実行できませんでした。

パケットフィルタリングフックを使用すると、IP フィルタを有効にする手順が簡素化されます。これらのフックを介して、IP フィルタはルーティング前 (入力) およびルーティング後 (出力) のフィルタタップを使用して、Oracle Solaris システムに対する入出力パケットフローを制御できます。

パケットフィルタリングフックによって pfil モジュールは不要になります。したがって、このモジュールに関連する次のコンポーネントも削除されます。

- pfil ドライバ
- pfil デーモン
- svc:/network/pfil SMF サービス

IP フィルタを有効にするためのタスクについては、第 26 章「IP フィルタ (タスク)」を参照してください。

IP フィルタと pfil STREAMS モジュール

注 –

pfil モジュールが IP フィルタに使用されるのは、次の Solaris リリースのみです。

- Solaris 10 3/05 リリース
- Solaris 10 1/06 リリース
- Solaris 10 6/06 リリース
- Solaris 10 11/06 リリース

Solaris 10 7/07 リリース以降では、pfil モジュールはパケットフィルタリングフックで置き換えられ、IP フィルタには使用されなくなりました。

---

pfil STREAMS モジュールは、IP フィルタを有効にするために必要です。ただし、IP フィルタには、モジュールを各インタフェースに転送する自動メカニズムはありません。そのかわり、pfil STREAMS モジュールは SMF サービス svc:/network/pfil によって管理されます。ネットワークインタフェースでフィルタリングをアクティブにするには、まず pfil.ap ファイルを構成します。そのあと svc:/network/pfil サービスをアクティブ化して、pfil STREAMS モジュールをネットワークインタフェースに転送します。STREAMS モジュールを有効にするには、システムをリブートするか、ifconfig コマンドを使用して、フィルタリングする各ネットワークインタフェースを unplumb したあと、再度 plumb します。IPv6 パケットフィルタリング機能をアクティブ化するには、inet6 バージョンのインタフェースを plumb する必要があります。

ネットワークインタフェースの pfil モジュールが見つからない場合、SMF サービスは保守状態になります。この状態をもたらすもっとも一般的な原因

は、/etc/ipf/pfil.ap ファイルが正しく編集されていないことです。サービスが保守モードになると、フィルタのログファイルにそのことが記録されます。

IP フィルタのアクティブ化に関連するタスクについては、675 ページの「IP フィルタの構成」を参照してください。

IP フィルタ用の IPv6

Solaris 6/06 リリース以降の IP フィルタでは、IPv6 がサポートされています。IPv6 パケットフィルタリングでは、発信元または宛先の IPv6 アドレス、IPv6 アドレスを含むプール、および IPv6 拡張ヘッダーに基づいて、パケットを取り出すことができます。

IPv6 は、多くの点で IPv4 に似ています。ただし、これら 2 つの IP バージョンは、ヘッダーとパケットサイズが異なっています。IP フィルタでは、これらは重要な要素です。IPv6 パケットには、「ジャンボグラム」と呼ばれる、65,535 バイトより大きなデータグラムが含まれています。IP フィルタでは、IPv6 ジャンボグラムはサポートされていません。IPv6 のその他の機能の詳細については、72 ページの「IPv6 の主な特長」を参照してください。

注 – ジャンボグラムの詳細については、Internet Engineering Task Force (IETF) の RFC 2675、『IPv6 Jumbograms』のドキュメントを参照してください。 http://www.ietf.org/rfc/rfc2675.txt

---

IPv6 に関連する IP フィルタのタスクは、IPv4 とほとんど変わりません。もっとも大きな違いは、特定のコマンドで -6 オプションを使用することです。ipf コマンドと ipfstat コマンドには、IPv6 パケットフィルタリングを使用するために、-6 オプションが用意されています。IPv6 パケットフィルタリング規則をロードおよびフラッシュするときは、ipf コマンドで -6 オプションを使用します。IPv6 統計を表示するときは、ipfstat コマンドに -6 オプションを使用します。ipmon コマンドと ippool コマンドでも IPv6 がサポートされますが、IPv6 をサポートするためのオプションは指定しません。ipmon コマンドは、IPv6 パケットのログを記録できるように拡張されています。ippool コマンドでは、IPv6 アドレスのプールをサポートしています。IPv4 アドレスまたは IPv6 アドレスのみのプールを作成したり、IPv4 アドレスと IPv6 アドレスを同じプールに含めたりできます。

ipf6.conf ファイルを使用して、IPv6 用のパケットフィルタリング規則セットを作成できます。デフォルトでは、ipf6.conf 構成ファイルは /etc/ipf ディレクトリに含まれています。ほかのフィルタリング構成ファイルと同様に、/etc/ipf ディレクトリに保存されている ipf6.conf ファイルは、ブート時に自動的にロードされます。作成した IPv6 構成ファイルを別の場所に保存し、そのファイルを手動でロードすることもできます。

IPv6 用のパケットフィルタリング規則を設定したら、inet6 バージョンのインタフェースを plumb して IPv6 パケットフィルタリング機能をアクティブにしてください。

IPv6 の詳細については、第 3 章「IPv6 の紹介 (概要)」を参照してください。IP フィルタに関連するタスクについては、第 26 章「IP フィルタ (タスク)」を参照してください。

IP フィルタのマニュアルページ

次の表に IP フィルタに関するマニュアルページのドキュメントを示します。

マニュアルページ	説明
ipf(1M)	ipf コマンドを実行して次のタスクを行う ■ パケットフィルタリング規則セットの処理 ■ フィルタリングの無効化と有効化 ■ 統計のリセットと現在のインタフェースステータスリストとカーネル内インタフェースリストの再同期化
ipf(4)	IP フィルタパケットのフィルタリング規則を作成するための文法と構文を含む。
ipfilter(5)	オープンソースの IP フィルタのライセンス情報を提供する
ipfs(1M)	ipfs コマンドを実行して、NAT 情報と状態テーブル情報を保存し、リブート後に復元する
ipfstat(1M)	ipfstat コマンドを実行して、パケット処理の統計を検索し、表示する
ipmon(1M)	ipmon コマンドを使用してログデバイスを開き、パケットのフィルタリングと NAT の両方に対してロギングされたパケットを参照する
ipnat(1M)	ipnat コマンドを実行して次のタスクを行う ■ NAT 規則の処理 ■ NAT 統計の検索と表示
ipnat(4)	NAT 規則を作成するための文法と構文を含む
ippool(1M)	ippool コマンドを実行して、アドレスプールの作成と管理を行う
ippool(4)	IP フィルタアドレスプールを作成するための文法と構文を含む。

マニュアルページ	説明
ndd(1M)	pfil STREAMS モジュールの現在のフィルタリングパラメータおよび調整可能なパラメータの現在値を表示する

◆ ◆ ◆ 第 26 章

26 IP フィルタ (タスク)

この章では、タスクの手順をステップごとに説明します。IP フィルタの概要情報は、第 25 章「Oracle Solaris の IP フィルタ (概要)」 を参照してください。

この章では、次の内容について説明します。

IP フィルタの構成

次のタスクマップに、IP フィルタの構成タスクに関連する手順を示します。

表 26–1 IP フィルタの構成 (タスクマップ)

タスク	説明	参照先
最初に IP フィルタを有効にします。	IP フィルタは、デフォルトでは無効です。IP フィルタを手動で有効にするか /etc/ipf/ ディレクトリにある構成ファイルを使用して、システムをリブートする必要があります。Solaris 10 7/07 リリース以降では、pfil モジュールの代わりにパケットフィルタリングフックで IP フィルタを有効にします。	676 ページの「IP フィルタを有効にする方法」

表 26–1 IP フィルタの構成 (タスクマップ) (続き)

タスク	説明	参照先
IP フィルタを再度有効にします。	IP フィルタが非アクティブ化された、または無効になった場合は、システムをリブートするか、ipf コマンドを使用して、IP フィルタを再度有効にできます。	677 ページの「IP フィルタを再度有効にする方法」
ループバックフィルタリングを有効にします。	オプションとして、ループバックフィルタリングを有効にすると、ゾーン間のトラフィックのフィルタリングなどを行うことができます。	678 ページの「ループバックフィルタリングを有効にする方法」

▼ IP フィルタを有効にする方法

Solaris 10 7/07 以降の OS を実行しているシステムで IP フィルタを有効にするには、次の手順に従います。Solaris 10 7/07 OS より前の Solaris 10 を実行しているシステムで IP フィルタを有効にする場合は、682 ページの「pfil モジュールの使用」を参照してください。

1 **IP Filter Management の権利プロファイルを持つ役割またはスーパーユーザーになります。**

IP Filter Management の権利プロファイルは、ユーザーが作成した役割に割り当てることができます。役割の作成と役割のユーザーへの割り当てについては、『Solaris のシステム管理: セキュリティーサービス』の「RBAC の構成 (タスクマップ)」を参照してください。

2 **パケットフィルタリング構成ファイルを作成します。**

パケットフィルタリング規則セットには、IP フィルタが使用するパケットフィルタリング規則が含まれています。ブート時にパケットフィルタリング規則をロードする場合は、/etc/ipf/ipf.conf ファイルを編集して IPv4 パケットフィルタリングを実装します。IPv6 パケットフィルタリング規則には /etc/ipf/ipf6.conf ファイルを使用します。ブート時にパケットフィルタリング規則をロードしない場合は、適当な場所に ipf.conf ファイルを保存し、パケットフィルタリングを手動でアクティブ化します。パケットのフィルタリングについては、663 ページの「IP フィルタのパケットのフィルタリング機能の使用」を参照してください。構成ファイルの操作については、709 ページの「IP フィルタ構成ファイルの作成と編集」を参照してください。

3 **(オプション) ネットワークアドレス変換 (NAT) 構成ファイルを作成します。**

注 – ネットワークアドレス変換 (NAT) では、IPv6 はサポートされていません。

ネットワークアドレス変換を使用する場合は、ipnat.conf ファイルを作成します。ネットワークアドレス変換を使用する場合は、ipnat.conf ファイルを作成します。 ブート時に NAT 規則をロードしない場合は、適当な場所に ipnat.conf ファイルを保存し、NAT 規則を手動でアクティブ化します。

NAT については、667 ページの「IP フィルタの NAT 機能の使用」を参照してください。

4 (オプション) アドレスプール構成ファイルを作成します。

ひとかたまりのアドレスを単一のアドレスプールとして参照する場合は、ipool.conf ファイルを作成します。ブート時にアドレスプール構成ファイルをロードする場合は、アドレスプールを保存する /etc/ipf/ippool.conf というファイルを作成します。ブート時にアドレスプール構成ファイルをロードしない場合は、適当な場所に ippool.conf ファイルを保存し、規則を手動でアクティブ化します。

アドレスプールは、IPv4 アドレスだけまたは IPv6 アドレスだけを含むことができます。IPv4 アドレスと IPv6 アドレスの両方を含むこともできます。

アドレスプールの詳細については、668 ページの「IP フィルタのアドレスプール機能の使用」を参照してください。

5 (オプション) ループバックトラフィックのフィルタリングを有効にします。

システムに構成されているゾーン間のトラフィックのフィルタリングを行う場合は、ループバックフィルタリングを有効にする必要があります。678 ページの「ループバックフィルタリングを有効にする方法」を参照してください。ゾーンに適用する適切な規則セットも必ず定義してください。

6 IP フィルタをアクティブにします。

```
# svcadm enable network/ipfilter
```

▼ IP フィルタを再度有効にする方法

パケットフィルタリングが一時的に有効でない場合は、再度有効にできます。

1 IP Filter Management の権利プロファイルを持つ役割またはスーパーユーザーになります。

IP Filter Management の権利プロファイルは、ユーザーが作成した役割に割り当てることができます。役割の作成と役割のユーザーへの割り当てについては、『Solaris のシステム管理: セキュリティーサービス』の「RBAC の構成 (タスクマップ)」を参照してください。

2 次のいずれかの方法で、IP フィルタを有効にし、フィルタリングをアクティブ化します。

- マシンをリブートします。

  ```
  # reboot
  ```

  注 – IP フィルタが有効になっているときに次のファイルが存在する場合は、リブート後にそれらのファイルがロードされます。 `/etc/ipf/ipf.conf` ファイル (IPv6 を使用している場合は `/etc/ipf/ipf6.conf` ファイル) または `/etc/ipf/ipnat.conf` ファイル。

- 次の一連のコマンドを実行して、IP フィルタを有効にし、フィルタリングをアクティブにします。

  a. IP フィルタを有効にします。

     ```
     # ipf -E
     ```

  b. パケットフィルタリングのアクティブ化

     ```
     # ipf -f filename
     ```

  c. (オプション) NAT のアクティブ化

     ```
     # ipnat -f filename
     ```

     注 – ネットワークアドレス変換 (NAT) では、IPv6 はサポートされていません。

▼ ループバックフィルタリングを有効にする方法

注 – ループバックトラフィックをフィルタリングできるのは、システムで Solaris 10 7/07 リリース以降が実行されている場合のみです。以前の Oracle Solaris 10 リリースでは、ループバックフィルタリングはサポートされません。

1 **IP Filter Management の権利プロファイルを持つ役割またはスーパーユーザーになります。**

IP Filter Management の権利プロファイルは、ユーザーが作成した役割に割り当てることができます。役割の作成と役割のユーザーへの割り当てについては、『Solaris のシステム管理: セキュリティーサービス』の「RBAC の構成 (タスクマップ)」を参照してください。

2 **IP フィルタが実行中の場合は、停止させます。**

```
# svcadm disable network/ipfilter
```

3 **`/etc/ipf.conf` ファイルまたは `/etc/ipf6.conf` ファイルを編集して、ファイルの先頭に次の行を追加します。**

```
set intercept_loopback true;
```

この行は、ファイル内で定義されるすべての IP フィルタ規則よりも前に置く必要があります。ただし、この行の前にコメントを挿入することはできます。次に例を示します。

```
#
# Enable loopback filtering to filter between zones
#
set intercept_loopback true;
#
# Define policy
#
block in all
block out all
<other rules>
...
```

4 **IP フィルタを起動します。**

```
# svcadm enable network/ipfilter
```

5 ループバックフィルタリングのステータスを確認するには、次のコマンドを使用します。

```
# ipf -T ipf_loopback
ipf_loopback    min 0   max 0x1 current 1
#
```

ループバックフィルタリングが無効になっている場合、このコマンドは次の出力を生成します。

```
ipf_loopback    min 0   max 0x1 current 0
```

IP フィルタの非アクティブ化と無効化

次のような場合、パケットフィルタリングと NAT を非アクティブ化または無効にしたほうがよいこともあります。

- テスト目的
- 問題の原因が IP フィルタにあると考えられる場合のシステムのトラブルシューティングを行う

次のタスクマップで、IP フィルタの機能を非アクティブ化または無効にする手順を示します。

表 26–2 IP フィルタの非アクティブ化と無効化 (タスクマップ)

タスク	説明	手順
パケットフィルタリングの非アクティブ化	`ipf` コマンドでパケットフィルタリングを非アクティブにする	680 ページの「パケットフィルタリングを非アクティブにする方法」

表 26–2 IP フィルタの非アクティブ化と無効化 (タスクマップ) (続き)

タスク	説明	手順
NAT の非アクティブ化	`ipnat` コマンドで NAT を非アクティブにする	681 ページの「NAT を非アクティブにする方法」
パケットフィルタリングと NAT の無効化	`ipf` コマンドでパケットフィルタリングと NAT を無効にする	681 ページの「パケットフィルタリングを無効にする方法」

▼ パケットフィルタリングを非アクティブにする方法

次の手順は、パケットフィルタリング規則をアクティブなフィルタリング規則セットから消去することによって、IP フィルタのパケットフィルタリングを非アクティブにします。この手順では、IP フィルタは無効になりません。規則を規則セットに追加することによって、IP フィルタを再度アクティブ化することもできます。

1 **IP Filter Management の権利プロファイルを持つ役割またはスーパーユーザーになります。**

IP Filter Management の権利プロファイルは、ユーザーが作成した役割に割り当てることができます。役割の作成と役割のユーザーへの割り当てについては、『Solaris のシステム管理: セキュリティーサービス』の「RBAC の構成 (タスクマップ)」を参照してください。

2 **次のいずれかの方法で IP フィルタ規則を非アクティブにします。**

- アクティブな規則セットをカーネルから削除します。

```
# ipf -Fa
```

このコマンドは、すべてのパケットフィルタリング規則を非アクティブにします。

- 受信パケットのフィルタリング規則を削除します。

```
# ipf -Fi
```

このコマンドは、 受信パケットのパケットフィルタリング規則を非アクティブにします。

- 送信パケットのフィルタリング規則を削除します。

```
# ipf -Fo
```

このコマンドは、 送信パケットのパケットフィルタリング規則を非アクティブにします。

▼ NAT を非アクティブにする方法

次の手順では、NAT 規則をアクティブな NAT 規則セットから消去することで、IP フィルタの NAT 規則を非アクティブにします。この手順では、IP フィルタは無効になりません。規則を規則セットに追加することによって、IP フィルタを再度アクティブ化することもできます。

1 **IP Filter Management** の権利プロファイルを持つ役割またはスーパーユーザーになります。

IP Filter Management の権利プロファイルは、ユーザーが作成した役割に割り当てることができます。役割の作成と役割のユーザーへの割り当てについては、『Solaris のシステム管理: セキュリティーサービス』の「RBAC の構成 (タスクマップ)」を参照してください。

2 **NAT** をカーネルから削除します。

```
# ipnat -FC
```

`-C` オプションは、現在の NAT 規則リストのすべてのエントリを削除します。`-F` オプションは、現在アクティブな NAT マッピングを示す現在の NAT 変換テーブルのすべてのアクティブなエントリを削除します。

▼ パケットフィルタリングを無効にする方法

この手順を実行すると、パケットフィルタリングと NAT の両方がカーネルから削除されます。この手順を使用する場合は、パケットフィルタリングと NAT を再度有効にするには、IP フィルタを再度有効にする必要があります。詳細については、677 ページの「IP フィルタを再度有効にする方法」を参照してください。

1 **IP Filter Management** の権利プロファイルを持つ役割またはスーパーユーザーになります。

IP Filter Management の権利プロファイルは、ユーザーが作成した役割に割り当てることができます。役割の作成と役割のユーザーへの割り当てについては、『Solaris のシステム管理: セキュリティーサービス』の「RBAC の構成 (タスクマップ)」を参照してください。

2 パケットフィルタリングを無効にし、すべてのパケットがネットワークを通過できるようにします。

```
# ipf –D
```

注 – `ipf -D` コマンドは、規則セットから規則を消去します。フィルタリングを再び有効にするときには、規則セットに規則を追加する必要があります。

---

pfil モジュールの使用

この節では、pfil STREAMS モジュールを使用して IP フィルタをアクティブまたは非アクティブにする方法と、pfil 統計を参照する方法について説明します。これらの手順は、次の Solaris リリースのいずれかを実行しているシステムでのみ適用できます。

- Solaris 10 3/05 リリース
- Solaris 10 1/06 リリース
- Solaris 10 6/06 リリース
- Solaris 10 11/06 リリース

次のタスクマップに、pfil モジュールを構成するためのタスクを示します。

表 26–3　pfil モジュールの使用 (作業マップ)

タスク	説明	参照先
IP フィルタの有効化	IP フィルタは、デフォルトでは無効です。IP フィルタを手動で有効にするか /etc/ipf/ ディレクトリにある構成ファイルを使用して、システムをリブートする必要があります。	682 ページの「以前の Solaris リリースで IP フィルタを有効にする方法」
NIC でのパケットフィルタリングのアクティブ化	pfil モジュールを構成して、NIC でのパケットフィルタリングをアクティブにします	685 ページの「NIC でパケットフィルタリングをアクティブにする方法」
NIC の IP フィルタの非アクティブ化	NIC を削除し、すべてのパケットが NIC を通過できるようにします。	686 ページの「NIC の IP フィルタを非アクティブにする方法」
pfil 統計の参照	pfil モジュールの統計を参照すると、ndd コマンドによる IP フィルタのトラブルシューティングに役立ちます。	688 ページの「IP フィルタの pfil 統計を参照する方法」

▼ 以前の Solaris リリースで IP フィルタを有効にする方法

IP フィルタは Oracle Solaris とともにインストールされます。ただし、パケットフィルタ処理はデフォルトでは有効になっていません。次の手順で IP フィルタをアクティブ化します。

注 – システムで Solaris 10 7/07 リリース以降が実行されている場合は、676 ページの「IP フィルタを有効にする方法」の、パケットフィルタリングフックを使用する手順に従ってください。

1 **IP Filter Management** の権利プロファイルを持つ役割またはスーパーユーザーになります。

IP Filter Management の権利プロファイルは、ユーザーが作成した役割に割り当てることができます。役割の作成と役割のユーザーへの割り当てについては、『Solaris のシステム管理: セキュリティーサービス』の「RBAC の構成 (タスクマップ)」を参照してください。

2 適当なファイルエディタを起動して、 **/etc/ipf/pfil.ap** ファイルを編集します。

このファイルには、ホスト上のネットワークインタフェースカード (NIC) の名前が含まれています。デフォルトでは、名前はコメントとされます。フィルタリングを実行するネットワークトラフィックのデバイス名をコメントから外してください。システムの NIC の名前が含まれていない場合は、NIC を指定する行を追加してください。

```
# vi /etc/ipf/pfil.ap
# IP Filter pfil autopush setup
#
# See autopush(1M) manpage for more information.
#
# Format of the entries in this file is:
#
#major  minor lastminor modules

#le     -1      0       pfil
#qe     -1      0       pfil
hme     -1      0       pfil (Device has been uncommented for filtering)
#qfe    -1      0       pfil
#eri    -1      0       pfil
#ce     -1      0       pfil
#bge    -1      0       pfil
#be     -1      0       pfil
#vge    -1      0       pfil
#ge     -1      0       pfil
#nf     -1      0       pfil
#fa     -1      0       pfil
#ci     -1      0       pfil
#el     -1      0       pfil
#ipdptp -1      0       pfil
#lane   -1      0       pfil
#dmfe   -1      0       pfil
```

3 **network/pfil** サービスインスタンスを再起動することによって、**/etc/ipf/pfil.ap** ファイルの変更内容を有効にします。

```
# svcadm restart network/pfil
```

4 パケットフィルタリング構成ファイルを作成します。

パケットフィルタリング規則セットには、IP フィルタが使用するパケットフィルタリング規則が含まれています。ブート時にパケットフィルタリング規則をロードする場合は、`/etc/ipf/ipf.conf` ファイルを編集して IPv4 パケットフィルタリングを実装します。IPv6 パケットフィルタリング規則には `/etc/ipf/ipf6.conf` ファイルを使用します。ブート時にパケットフィルタリング規則をロードしない場合は、適当な場所に ipf.conf ファイルを保存し、パケットフィルタリングを手動でアクティブ化します。パケットのフィルタリングについては、663 ページの「IP フィルタのパケットのフィルタリング機能の使用」を参照してください。構成ファイルの操作については、709 ページの「IP フィルタ構成ファイルの作成と編集」を参照してください。

5 (オプション) ネットワークアドレス変換 (NAT) 構成ファイルを作成します。

注 – ネットワークアドレス変換 (NAT) では、IPv6 はサポートされていません。

ネットワークアドレス変換を使用する場合は、`ipnat.conf` ファイルを作成します。ネットワークアドレス変換を使用する場合は、`ipnat.conf` ファイルを作成します。 ブート時に NAT 規則をロードしない場合は、適当な場所に `ipnat.conf` ファイルを保存し、NAT 規則を手動でアクティブ化します。

NAT については、667 ページの「IP フィルタの NAT 機能の使用」を参照してください。

6 (オプション) アドレスプール構成ファイルを作成します。

ひとかたまりのアドレスを単一のアドレスプールとして参照する場合は、`ipool.conf` ファイルを作成します。ブート時にアドレスプール構成ファイルをロードする場合は、アドレスプールを保存する `/etc/ipf/ippool.conf` というファイルを作成します。ブート時にアドレスプール構成ファイルをロードしない場合は、適当な場所に `ippool.conf` ファイルを保存し、規則を手動でアクティブ化します。

アドレスプールは、IPv4 アドレスだけまたは IPv6 アドレスだけを含むことができます。IPv4 アドレスと IPv6 アドレスの両方を含むこともできます。

アドレスプールの詳細については、668 ページの「IP フィルタのアドレスプール機能の使用」を参照してください。

7 次のいずれかの方法で IP フィルタを有効にします。

- IP フィルタを有効にして、マシンをリブートします。

```
# svcadm enable network/ipfilter
# reboot
```

注 – NIC で ifconfig unplumb コマンドと ifconfig plumb コマンドを安全に使用できない場合は、リブートが必要です。

- ifconfig unplumb コマンドと ifconfig plumb コマンドを使用して、NIC を有効にします。次に、IP フィルタを有効にします。IPv6 パケットフィルタリングを実装するには、inet6 バージョンのインタフェースが plumb されている必要があります。

```
# ifconfig hme0 unplumb
# ifconfig hme0 plumb 192.168.1.20 netmask 255.255.255.0 up
# ifconfig hme0 inte6 unplumb
# ifconfig hme0 inet6 plumb fec3:f849::1/96 up
# svcadm enable network/ipfilter
```

ifconfig コマンドの詳細については、ifconfig(1M) のマニュアルページを参照してください。

▼ NIC でパケットフィルタリングをアクティブにする方法

/etc/ipf/ipf.conf ファイル (IPv6 使用時は /etc/ipf/ipf6.conf ファイル) が存在する場合は、IP フィルタはブート時に有効になります。IP フィルタを有効にしたあと、NIC でフィルタリングを有効にする必要がある場合は、次の手順で行います。

1. **IP Filter Management の権利プロファイルを持つ役割またはスーパーユーザーになります。**

   IP Filter Management の権利プロファイルは、ユーザーが作成した役割に割り当てることができます。役割の作成と役割のユーザーへの割り当てについては、『Solaris のシステム管理: セキュリティーサービス』の「RBAC の構成 (タスクマップ)」を参照してください。

2. **適当なファイルエディタを起動して、 /etc/ipf/pfil.ap ファイルを編集します。**

   このファイルには、ホスト上の NIC の名前が含まれています。デフォルトでは、名前はコメントとされます。フィルタリングを実行するネットワークトラフィックのデバイス名をコメントから外してください。システムの NIC の名前が含まれていない場合は、NIC を指定する行を追加してください。

```
# vi /etc/ipf/pfil.ap
# IP Filter pfil autopush setup
#
# See autopush(1M) manpage for more information.
#
# Format of the entries in this file is:
#
#major  minor lastminor modules
```

```
#le     -1      0       pfil
#qe     -1      0       pfil
hme     -1      0       pfil (Device has been uncommented for filtering)
#qfe    -1      0       pfil
#eri    -1      0       pfil
#ce     -1      0       pfil
#bge    -1      0       pfil
#be     -1      0       pfil
#vge    -1      0       pfil
#ge     -1      0       pfil
#nf     -1      0       pfil
#fa     -1      0       pfil
#ci     -1      0       pfil
#el     -1      0       pfil
#ipdptp -1      0       pfil
#lane   -1      0       pfil
#dmfe   -1      0       pfil
```

3 **network/pfil** サービスインスタンスを再起動することによって、**/etc/ipf/pfil.ap** ファイルの変更内容を有効にします。

```
# svcadm restart network/pfil
```

4 次の方法のいずれかで **NIC** を有効にします。

- マシンをリブートします。

  ```
  # reboot
  ```

  注 – NIC で `ifconfig unplumb` コマンドと `ifconfig plumb` コマンドを安全に使用できない場合は、リブートが必要です。

- `unplumb` と `plumb` オプションを指定して `ifconfig` コマンドを実行し、フィルタリングする NIC を有効にします。IPv6 パケットフィルタリングを実装するには、`inet6` バージョンの各インタフェースが plumb されている必要があります。

  ```
  # ifconfig hme0 unplumb
  # ifconfig hme0 plumb 192.168.1.20  netmask 255.255.255.0  up
  # ifconfig hme0 inet6 unplumb
  # ifconfig hme0 inet6 plumb fec3:f840::1/96 up
  ```

  `ifconfig` コマンドの詳細については、`ifconfig(1M)` のマニュアルページを参照してください。

▼ NIC の IP フィルタを非アクティブにする方法

NIC でフィルタリングパケットを停止するには、次の手順を実行します。

1 **IP Filter Management** の権利プロファイルを持つ役割またはスーパーユーザーになります。

IP Filter Management の権利プロファイルは、ユーザーが作成した役割に割り当てることができます。役割の作成と役割のユーザーへの割り当てについては、『Solaris のシステム管理: セキュリティーサービス』の「RBAC の構成 (タスクマップ)」を参照してください。

2 適当なファイルエディタを起動して、 **/etc/ipf/pfil.ap** ファイルを編集します。

このファイルには、ホスト上の NIC の名前が含まれています。ネットワークトラフィックのフィルタリングに使用されていた NIC は、コメントから外されます。ネットワークトラフィックのフィルタリングで使用しなくなったデバイスの名前をコメントにします。

```
# vi /etc/ipf/pfil.ap
# IP Filter pfil autopush setup
#
# See autopush(1M) manpage for more information.
#
# Format of the entries in this file is:
#
#major  minor lastminor modules

#le     -1      0       pfil
#qe     -1      0       pfil
#hme    -1      0       pfil (Commented-out device no longer filters network traffic)
#qfe    -1      0       pfil
#eri    -1      0       pfil
#ce     -1      0       pfil
#bge    -1      0       pfil
#be     -1      0       pfil
#vge    -1      0       pfil
#ge     -1      0       pfil
#nf     -1      0       pfil
#fa     -1      0       pfil
#ci     -1      0       pfil
#el     -1      0       pfil
#ipdptp -1      0       pfil
#lane   -1      0       pfil
#dmfe   -1      0       pfil
```

3 次の方法のいずれかで **NIC** を非アクティブにします。

- マシンをリブートします。

```
# reboot
```

注 – NIC で `ifconfig unplumb` コマンドと `ifconfig plumb` コマンドを安全に使用できない場合は、リブートが必要です。

- unplumb と plumb オプションを指定して ifconfig コマンドを実行し、NIC を非アクティブにします。IPv6 パケットフィルタリングを非アクティブにするには、inet6 バージョンの各インタフェースを unplumb する必要があります。次の手順を実行します。システムのサンプルデバイスは hme です。

  a. 無効にするデバイスのメジャー番号を特定します。

  ```
  # grep hme /etc/name_to_major
  hme 7
  ```

  b. hme0 の現在の autopush 構成を表示します。

  ```
  # autopush -g -M 7 -m 0
     Major     Minor     Lastminor       Modules
         7       ALL           -          pfil
  ```

  c. その autopush 構成を削除します。

  ```
  # autopush -r -M 7 -m 0
  ```

  d. デバイスを開いて IP アドレスを割り当てます。

  ```
  # ifconfig hme0 unplumb
  # ifconfig hme0 plumb 192.168.1.20  netmask 255.255.255.0  up
  # ifconfig hme0 inet6 unplumb
  # ifconfig hme0 inet6 plumb fec3:f840::1/96 up
  ```

  ifconfig コマンドの詳細については、ifconfig(1M) のマニュアルページを参照してください。

▼ IP フィルタの pfil 統計を参照する方法

IP フィルタをトラブルシューティングするときに pfil 統計を参照できます。

1 **IP Filter Management** の権利プロファイルを持つ役割またはスーパーユーザーになります。

IP Filter Management の権利プロファイルは、ユーザーが作成した役割に割り当てることができます。役割の作成と役割のユーザーへの割り当てについては、『Solaris のシステム管理: セキュリティーサービス』の「RBAC の構成 (タスクマップ)」を参照してください。

2 **pfil** 統計の参照

```
# ndd -get /dev/pfil qif_status
```

例 26–1 IP フィルタの pfil 統計の参照

次の例は、**pfil** 統計の参照方法を示しています。

```
# ndd -get /dev/pfil qif_status
ifname ill q OTHERQ num sap hl nr nw bad copy copyfail drop notip nodata
   notdata
```

```
QIF6 0 300011247b8 300011248b0 6 806 0 4 9 0 0 0 0 0 0 0 0
dmfe1 3000200a018 30002162a50 30002162b48 5 800 14 171 13681 0 0 0 0 0 0 0
```

IP フィルタ規則セットの操作

次のタスクマップに、IP フィルタの規則セットに関連するタスクを示します。

表 26–4　IP フィルタ規則セットの操作 (タスクマップ)

タスク	説明	手順
IP フィルタのパケットフィルタリング規則セットの管理、参照、変更		690 ページの「IP フィルタのパケットフィルタリング規則セットの管理」
	アクティブなパケットフィルタリング規則セットを参照する	690 ページの「アクティブなパケットフィルタリング規則セットを参照する方法」
	アクティブでないパケットフィルタリング規則セットを参照する	691 ページの「アクティブでないパケットフィルタリング規則セットを参照する方法」
	別のアクティブな規則セットをアクティブにする	691 ページの「別のパケットフィルタリング規則セット、または更新されたパケットフィルタリング規則セットをアクティブにする方法」
	規則セットを削除する	693 ページの「パケットフィルタリング規則セットを削除する方法」
	規則セットへ規則を追加する	694 ページの「アクティブなパケットフィルタリング規則セットに規則を追加する方法」
		695 ページの「アクティブでないパケットフィルタリング規則セットに規則を追加する方法」
	アクティブな規則セットとアクティブでない規則セット間を移動する	695 ページの「アクティブなパケットフィルタリング規則セットとアクティブでないパケットフィルタリング規則セットを切り替える方法」
	アクティブでない規則セットをカーネルから削除する	697 ページの「カーネルからアクティブでないパケットフィルタリング規則セットを削除する方法」

表 26–4 IP フィルタ規則セットの操作 (タスクマップ) (続き)

タスク	説明	手順
IP フィルタの NAT 規則の管理、参照、変更		697 ページの「IP フィルタ用 NAT 規則の管理」
	アクティブな NAT 規則を参照する	698 ページの「アクティブな NAT 規則を参照する方法」
	NAT 規則を削除する	698 ページの「NAT 規則を削除する方法」
	NAT 規則へさらに規則を追加する	699 ページの「NAT 規則に規則を追加する方法」
IP フィルタのアドレスプールの管理、参照、変更		700 ページの「IP フィルタのアドレスプールの管理」
	アクティブなアドレスプールを参照する	700 ページの「アクティブなアドレスプールを参照する方法」
	アドレスプールを削除する	700 ページの「アドレスプールを削除する方法」
	アドレスプールへさらに規則を追加する	701 ページの「規則をアドレスプールに追加する方法」

IP フィルタのパケットフィルタリング規則セットの管理

有効な場合、アクティブなパケットフィルタリング規則セットとアクティブでないパケットフィルタリング規則セットの両方をカーネルに置くことができます。アクティブな規則セットによって、受信パケットと送信パケットに対して実行するフィルタリングが決まります。アクティブでない規則セットでも規則を格納します。アクティブでない規則セットは、アクティブな規則セットにしない限り、使用されることはありません。アクティブなパケットフィルタリング規則セットとアクティブでないパケットフィルタリング規則セットの両方を管理、参照、変更できます。

▼ アクティブなパケットフィルタリング規則セットを参照する方法

1 **IP Filter Management** の権利プロファイルを持つ役割またはスーパーユーザーになります。

IP Filter Management の権利プロファイルは、ユーザーが作成した役割に割り当てることができます。役割の作成と役割のユーザーへの割り当てについては、『Solaris のシステム管理: セキュリティーサービス』の「RBAC の構成 (タスクマップ)」を参照してください。

2 カーネルにロードされているアクティブなパケットフィルタリング規則セットを参照します。

```
# ipfstat -io
```

例 26–2 アクティブなパケットフィルタリング規則セットの参照

次の例は、カーネルにロードされたアクティブなパケットフィルタリング規則セットからの出力を示しています。

```
# ipfstat -io
empty list for ipfilter(out)
pass in quick on dmfe1 from 192.168.1.0/24 to any
pass in all
block in on dmfe1 from 192.168.1.10/32 to any
```

▼ アクティブでないパケットフィルタリング規則セットを参照する方法

1 **IP Filter Management** の権利プロファイルを持つ役割またはスーパーユーザーになります。

IP Filter Management の権利プロファイルは、ユーザーが作成した役割に割り当てることができます。役割の作成と役割のユーザーへの割り当てについては、『Solaris のシステム管理: セキュリティーサービス』の「RBAC の構成 (タスクマップ)」を参照してください。

2 アクティブでないパケットフィルタリング規則セットを参照します。

```
# ipfstat -I -io
```

例 26–3 アクティブでないパケットフィルタリング規則セットの参照

次の例は、アクティブでないパケットフィルタリング規則セットからの出力を示しています。

```
# ipfstat -I -io
pass out quick on dmfe1 all
pass in quick on dmfe1 all
```

▼ 別のパケットフィルタリング規則セット、または更新されたパケットフィルタリング規則セットをアクティブにする方法

次のいずれかのタスクを実行する場合には、ここで示す手順を実行します。

- IP フィルタが現在使用しているパケットフィルタリング規則セット以外のパケットフィルタリング規則セットをアクティブにする
- 新規更新されたフィルタリング規則セットを再読み込みする

1 **IP Filter Management** の権利プロファイルを持つ役割またはスーパーユーザーになります。

IP Filter Management の権利プロファイルは、ユーザーが作成した役割に割り当てることができます。役割の作成と役割のユーザーへの割り当てについては、『Solaris のシステム管理: セキュリティーサービス』の「RBAC の構成 (タスクマップ)」を参照してください。

2 次の手順から 1 つを選択します。

- まったく異なる規則セットをアクティブにする場合は、別個のファイルを選択し、そこに新規規則セットを作成します。
- 規則セットを含む構成ファイルを編集して、現在の規則セットを更新します。

3 現在の規則セットを削除し、新しい規則セットをロードします。

```
# ipf -Fa -f filename
```

filename には、新しい規則セットを含む新規ファイル、またはアクティブな規則セットを含む更新されたファイルを指定できます。

アクティブな規則セットがカーネルから削除されます。*filename* ファイル内の規則がアクティブな規則セットになります。

注 – 現在の構成ファイルの再読み込みをしていても、このコマンドを実行する必要があります。それ以外の場合、以前の規則セットがアクティブであり続けるため、更新した構成ファイル内の変更された規則セットが適用されません。

更新した規則セットをロードするのに `ipf -D` や `svcadm restart` などのコマンドを使わないでください。これらのコマンドは、新しい規則セットをロードする前にファイアウォールを無効にするため、ネットワークが危険にさらされます。

例 26–4 別のパケットフィルタリング規則セットのアクティブ化

次の例は、あるパケットフィルタリング規則セットを、別の構成ファイル `/etc/ipf/ipf.conf` 内の別のパケットフィルタリング規則セットに置換する方法を示しています。

```
# ipfstat -io
empty list for ipfilter(out)
pass in quick on dmfe all
# ipf -Fa -f /etc/ipf/ipf.conf
# ipfstat -io
empty list for ipfilter(out)
block in log quick from 10.0.0.0/8 to any
```

例 26–5 更新したパケットフィルタリング規則セットの再読み込み

次の例は、現在アクティブでこれから更新するパケットフィルタリング規則セットを再読み込みする方法を示しています。この例で使用しているファイルは、/etc/ipf/ipf.conf です。

```
# ipfstat -io (Optional)
empty list for ipfilter (out)
block in log quick from 10.0.0.0/8 to any

(Edit the /etc/ipf/ipf.conf configuration file.)

# ipf -Fa -f /etc/ipf/ipf.conf
# ipfstat -io (Optional)
empty list for ipfilter (out)
block in log quick from 10.0.0.0/8 to any
block in quick on elx10 from 192.168.0.0/12 to any
```

▼ パケットフィルタリング規則セットを削除する方法

1. **IP Filter Management** の権利プロファイルを持つ役割またはスーパーユーザーになります。

   IP Filter Management の権利プロファイルは、ユーザーが作成した役割に割り当てることができます。役割の作成と役割のユーザーへの割り当てについては、『Solaris のシステム管理: セキュリティーサービス』の「RBAC の構成 (タスクマップ)」を参照してください。

2. 規則セットを削除します。

   ```
   # ipf -F [a|i|o]
   ```

   -a すべてのフィルタリング規則を規則セットから削除します。

   -i 受信パケットのフィルタリング規則を削除します。

   -o 送信パケットのフィルタリング規則を削除します。

例 26–6 パケットフィルタリング規則セットの削除

次の例は、すべてのフィルタリング規則をアクティブなフィルタリング規則セットから削除する方法を示しています。

```
# ipfstat -io
block out log on dmf0 all
block in log quick from 10.0.0.0/8 to any
# ipf -Fa
# ipfstat -io
empty list for ipfilter(out)
empty list for ipfilter(in)
```

▼ アクティブなパケットフィルタリング規則セットに規則を追加する方法

1. **IP Filter Management の権利プロファイルを持つ役割またはスーパーユーザーになります。**

   IP Filter Management の権利プロファイルは、ユーザーが作成した役割に割り当てることができます。役割の作成と役割のユーザーへの割り当てについては、『Solaris のシステム管理: セキュリティーサービス』の「RBAC の構成 (タスクマップ)」を参照してください。

2. **次のいずれかの方法で規則をアクティブな規則セットに追加します。**

   - `ipf -f -` コマンドを使用して、コマンド行で、規則セットに規則を追加します。

     ```
     # echo "block in on dmfe1 proto tcp from 10.1.1.1/32 to any" | ipf -f -
     ```

   - 次のコマンドを実行します。

     a. 適当なファイルに規則セットを作成します。

     b. 作成しておいた規則をアクティブな規則セットに追加します。

        ```
        # ipf -f filename
        ```

        filename の規則がアクティブな規則セットの最後に追加されます。IP フィルタは「最後に一致した規則を採用する」アルゴリズムを使用するため、`quick` キーワードを使用しないかぎり、追加した規則によってフィルタリングの優先順位が決まります。パケットが `quick` キーワードを含む規則に一致する場合は、その規則に対する処理が実行され、それ以降の規則はチェックされません。

例 26–7 アクティブなパケットフィルタリング規則セットへの規則の追加

次の例は、コマンド行から、アクティブなパケットフィルタリング規則セットに規則を追加する方法を示しています。

```
# ipfstat -io
empty list for ipfilter(out)
block in log quick from 10.0.0.0/8 to any
# echo "block in on dmfe1 proto tcp from 10.1.1.1/32 to any" | ipf -f -
# ipfstat -io
empty list for ipfilter(out)
block in log quick from 10.0.0.0/8 to any
block in on dmfe1 proto tcp from 10.1.1.1/32 to any
```

▼ アクティブでないパケットフィルタリング規則セットに規則を追加する方法

1. **IP Filter Management** の権利プロファイルを持つ役割またはスーパーユーザーになります。

   IP Filter Management の権利プロファイルは、ユーザーが作成した役割に割り当てることができます。役割の作成と役割のユーザーへの割り当てについては、『Solaris のシステム管理: セキュリティーサービス』の「RBAC の構成 (タスクマップ)」を参照してください。

2. 適当なファイルに規則セットを作成します。

3. 作成しておいた規則をアクティブでない規則セットに追加します。

   # ipf -I -f *filename*

   filename の規則がアクティブでない規則セットの最後に追加されます。IP フィルタは「最後に一致した規則を採用する」アルゴリズムを使用するため、`quick` キーワードを使用しないかぎり、追加した規則によってフィルタリングの優先順位が決まります。パケットが `quick` キーワードを含む規則に一致する場合は、その規則に対する処理が実行され、それ以降の規則はチェックされません。

例 26–8 アクティブでない規則セットへの規則の追加

次の例は、ファイルからアクティブでない規則セットに規則を追加する方法を示しています。

```
# ipfstat -I -io
pass out quick on dmfe1 all
pass in quick on dmfe1 all
# ipf -I -f /etc/ipf/ipf.conf
# ipfstat -I -io
pass out quick on dmfe1 all
pass in quick on dmfe1 all
block in log quick from 10.0.0.0/8 to any
```

▼ アクティブなパケットフィルタリング規則セットとアクティブでないパケットフィルタリング規則セットを切り替える方法

1. **IP Filter Management** の権利プロファイルを持つ役割またはスーパーユーザーになります。

   IP Filter Management の権利プロファイルは、ユーザーが作成した役割に割り当てることができます。役割の作成と役割のユーザーへの割り当てについては、『Solaris のシステム管理: セキュリティーサービス』の「RBAC の構成 (タスクマップ)」を参照してください。

2 アクティブな規則セットとアクティブでない規則セットを切り替えます。

```
# ipf -s
```

このコマンドを使用すると、カーネル内のアクティブな規則セットとアクティブでない規則セットを切り替えることができます。なお、アクティブでない規則セットが空の場合は、パケットフィルタリングは行われません。

例 26–9 アクティブなパケットフィルタリング規則セットとアクティブでないパケットフィルタリング規則セットの切り替え

次の例は、ipf - s コマンドの使用によって、どのようにアクティブでない規則セットがアクティブな規則セットになり、アクティブな規則セットがアクティブでない規則セットになるのかを示しています。

- ipf -s コマンドを実行する前に、ipfstat -I - io コマンドからの出力でアクティブでない規則セット内の規則が示されます。ipfstat -io コマンドからの出力は、アクティブな規則セットの規則を示します。

```
# ipfstat -io
empty list for ipfilter(out)
block in log quick from 10.0.0.0/8 to any
block in on dmfe1 proto tcp from 10.1.1.1/32 to any
# ipfstat -I -io
pass out quick on dmfe1 all
pass in quick on dmfe1 all
block in log quick from 10.0.0.0/8 to any
```

- ipf -s コマンドの実行後、ipfstat -I -io コマンドおよび ipfstat -io コマンドからの出力によって、2 つの規則セットの内容が切り替えられたことが示されます。

```
# ipf -s
Set 1 now inactive
# ipfstat -io
pass out quick on dmfe1 all
pass in quick on dmfe1 all
block in log quick from 10.0.0.0/8 to any
# ipfstat -I -io
empty list for inactive ipfilter(out)
block in log quick from 10.0.0.0/8 to any
block in on dmfe1 proto tcp from 10.1.1.1/32 to any
```

▼ カーネルからアクティブでないパケットフィルタリング規則セットを削除する方法

1 **IP Filter Management** の権利プロファイルを持つ役割またはスーパーユーザーになります。

IP Filter Management の権利プロファイルは、ユーザーが作成した役割に割り当てることができます。役割の作成と役割のユーザーへの割り当てについては、『Solaris のシステム管理: セキュリティーサービス』の「RBAC の構成 (タスクマップ)」を参照してください。

2 全削除コマンドで、アクティブでない規則セットを指定します。

```
# ipf -I -Fa
```

このコマンドが、アクティブでない規則セットをカーネルから消去します。

注 – 続けて `ipf -s` を実行すると、空のアクティブでない規則セットがアクティブな規則セットになります。アクティブな規則セットが空の場合は、フィルタリングが行われません。

例 26–10 カーネルからのアクティブでないパケットフィルタリング規則セットの削除

次の例は、すべての規則が削除されるように、アクティブでないパケットフィルタリング規則セットを消去する方法を示しています。

```
# ipfstat -I -io
empty list for inactive ipfilter(out)
block in log quick from 10.0.0.0/8 to any
block in on dmfe1 proto tcp from 10.1.1.1/32 to any
# ipf -I -Fa
# ipfstat -I -io
empty list for inactive ipfilter(out)
empty list for inactive ipfilter(in)
```

IP フィルタ用 NAT 規則の管理

次の手順で NAT 規則を管理、参照および変更します。

▼ アクティブな NAT 規則を参照する方法

1. **IP Filter Management** の権利プロファイルを持つ役割またはスーパーユーザーになります。

   IP Filter Management の権利プロファイルは、ユーザーが作成した役割に割り当てることができます。役割の作成と役割のユーザーへの割り当てについては、『Solaris のシステム管理: セキュリティーサービス』の「RBAC の構成 (タスクマップ)」を参照してください。

2. アクティブな **NAT** 規則を参照します。

   ```
   # ipnat -l
   ```

例 26–11 アクティブな NAT 規則の参照

次の例は、アクティブな NAT 規則セットからの出力を示しています。

```
# ipnat -l
List of active MAP/Redirect filters:
map dmfe0 192.168.1.0/24 -> 20.20.20.1/32

List of active sessions:
```

▼ NAT 規則を削除する方法

1. **IP Filter Management** の権利プロファイルを持つ役割またはスーパーユーザーになります。

   IP Filter Management の権利プロファイルは、ユーザーが作成した役割に割り当てることができます。役割の作成と役割のユーザーへの割り当てについては、『Solaris のシステム管理: セキュリティーサービス』の「RBAC の構成 (タスクマップ)」を参照してください。

2. 現在の **NAT** 規則を削除します。

   ```
   # ipnat -C
   ```

例 26–12 NAT 規則の削除

次の例は、現在の NAT 規則のエントリを削除する方法を示してます。

```
# ipnat -l
List of active MAP/Redirect filters:
map dmfe0 192.168.1.0/24 -> 20.20.20.1/32

List of active sessions:
# ipnat -C
1 entries flushed from NAT list
```

```
# ipnat -l
List of active MAP/Redirect filters:

List of active sessions:
```

▼ NAT 規則に規則を追加する方法

1. **IP Filter Management** の権利プロファイルを持つ役割またはスーパーユーザーになります。

   IP Filter Management の権利プロファイルは、ユーザーが作成した役割に割り当てることができます。役割の作成と役割のユーザーへの割り当てについては、『Solaris のシステム管理: セキュリティーサービス』の「RBAC の構成 (タスクマップ)」を参照してください。

2. 次のいずれかの方法で規則をアクティブな規則セットに追加します。

   - `ipnat -f -` コマンドを使用して、コマンド行で、NAT 規則セットに規則を追加します。

     ```
     # echo "map dmfe0 192.168.1.0/24 -> 20.20.20.1/32" | ipnat -f -
     ```

   - 次のコマンドを実行します。

     a. 適当なファイルに追加の NAT 規則を作成します。

     b. 作成しておいた規則をアクティブな NAT 規則に追加します。

        ```
        # ipnat -f filename
        ```

        filename の規則がアクティブな NAT 規則の最後に追加されます。

例 26–13 NAT 規則セットへの規則の追加

次の例は、コマンド行から、NAT 規則セットに規則を追加する方法を示しています。

```
# ipnat -l
List of active MAP/Redirect filters:

List of active sessions:
# echo "map dmfe0 192.168.1.0/24 -> 20.20.20.1/32" | ipnat -f -
# ipnat -l
List of active MAP/Redirect filters:
map dmfe0 192.168.1.0/24 -> 20.20.20.1/32

List of active sessions:
```

IP フィルタのアドレスプールの管理

次の手順でアドレスプールを管理、参照および変更します。

▼ アクティブなアドレスプールを参照する方法

1 **IP Filter Management** の権利プロファイルを持つ役割またはスーパーユーザーになります。

IP Filter Management の権利プロファイルは、ユーザーが作成した役割に割り当てることができます。役割の作成と役割のユーザーへの割り当てについては、『Solaris のシステム管理: セキュリティーサービス』の「RBAC の構成 (タスクマップ)」を参照してください。

2 アクティブなアドレスプールを参照します。

```
# ippool -l
```

例 26–14 アクティブなアドレスプールの参照

次の例は、アクティブなアドレスプールの内容を参照する方法を示しています。

```
# ippool -l
table role = ipf type = tree number = 13
        { 10.1.1.1/32, 10.1.1.2/32, 192.168.1.0/24; };
```

▼ アドレスプールを削除する方法

1 **IP Filter Management** の権利プロファイルを持つ役割またはスーパーユーザーになります。

IP Filter Management の権利プロファイルは、ユーザーが作成した役割に割り当てることができます。役割の作成と役割のユーザーへの割り当てについては、『Solaris のシステム管理: セキュリティーサービス』の「RBAC の構成 (タスクマップ)」を参照してください。

2 現在のアドレスプールのエントリを削除します。

```
# ippool -F
```

例 26–15 アドレスプールの削除

次の例は、アドレスプールを削除する方法を示しています。

```
# ippool -l
table role = ipf type = tree number = 13
        { 10.1.1.1/32, 10.1.1.2/32, 192.168.1.0/24; };
```

```
# ippool -F
1 object flushed
# ippool -l
```

▼ 規則をアドレスプールに追加する方法

1. **IP Filter Management** の権利プロファイルを持つ役割またはスーパーユーザーになります。

   IP Filter Management の権利プロファイルは、ユーザーが作成した役割に割り当てることができます。役割の作成と役割のユーザーへの割り当てについては、『Solaris のシステム管理: セキュリティーサービス』の「RBAC の構成 (タスクマップ)」を参照してください。

2. 次のいずれかの方法で規則をアクティブな規則セットに追加します。

   - `ippool-f -` コマンドを使用して、コマンド行で、規則セットに規則を追加します。

     ```
     # echo "table role = ipf type = tree number = 13
     {10.1.1.1/32, 10.1.1.2/32, 192.168.1.0/24};" | ippool -f -
     ```

   - 次のコマンドを実行します。

     a. 適当なファイルに追加のアドレスプールを作成します。

     b. 作成しておいた規則をアクティブなアドレスプールに追加します。

        ```
        # ippool -f filename
        ```

        filename の規則がアクティブなアドレスプールの最後に追加されます。

例 26–16 アドレスプールへの規則の追加

次の例は、コマンド行から、アドレスプール規則セットにアドレスプールを追加する方法を示しています。

```
# ippool -l
table role = ipf type = tree number = 13
        { 10.1.1.1/32, 10.1.1.2/32, 192.168.1.0/24; };
# echo "table role = ipf type = tree number = 100
 {10.0.0.0/32, 172.16.1.2/32, 192.168.1.0/24};" | ippool -f -
# ippool -l
table role = ipf type = tree number = 100
        { 10.0.0.0/32, 172.16.1.2/32, 192.168.1.0/24; };
table role = ipf type = tree number = 13
        { 10.1.1.1/32, 10.1.1.2/32, 192.168.1.0/24; };
```

IP フィルタの統計および情報の表示

表 26–5 IP フィルタの統計および情報の表示 (タスクマップ)

タスク	説明	手順
状態テーブルの参照	`ipfstat` コマンドで、パケットフィルタリングに関する情報を取得する	702 ページの「IP フィルタの状態テーブルを参照する方法」
状態統計の参照	`ipfstat -s` コマンドでパケット状態情報の統計を参照する	703 ページの「IP フィルタの状態統計を参照する方法」
NAT 統計の参照	`ipnat -s` コマンドで NAT 統計を参照する	704 ページの「IP フィルタの NAT 統計を参照する方法」
アドレスプール統計の参照	`ippool - s` コマンドでアドレスプール統計を参照する	704 ページの「IP フィルタのアドレスプール統計情報を表示する方法」

▼ IP フィルタの状態テーブルを参照する方法

1 **IP Filter Management** の権利プロファイルを持つ役割またはスーパーユーザーになります。

IP Filter Management の権利プロファイルは、ユーザーが作成した役割に割り当てることができます。役割の作成と役割のユーザーへの割り当てについては、『Solaris のシステム管理: セキュリティーサービス』の「RBAC の構成 (タスクマップ)」を参照してください。

2 状態テーブルを参照します。

```
# ipfstat
```

注 – `-t` オプションを使用すると、状態テーブルをトップユーティリティー形式で参照できます。

例 26–17 IP フィルタの状態テーブルの参照

次の例は、状態テーブルの参照方法を示しています。

```
# ipfstat
bad packets:            in 0    out 0
 input packets:         blocked 160 passed 11 nomatch 1 counted 0 short 0
output packets:         blocked 0 passed 13681 nomatch 6844 counted 0 short 0
 input packets logged:  blocked 0 passed 0
output packets logged:  blocked 0 passed 0
```

```
 packets logged:        input 0 output 0
 log failures:          input 0 output 0
fragment state(in):     kept 0  lost 0
fragment state(out):    kept 0  lost 0
packet state(in):       kept 0  lost 0
packet state(out):      kept 0  lost 0
ICMP replies:   0       TCP RSTs sent:  0
Invalid source(in):     0
Result cache hits(in):  152     (out):  6837
IN Pullups succeeded:   0       failed: 0
OUT Pullups succeeded:  0       failed: 0
Fastroute successes:    0       failures:       0
TCP cksum fails(in):    0       (out):  0
IPF Ticks:      14341469
Packet log flags set: (0)
        none
```

▼ IP フィルタの状態統計を参照する方法

1 **IP Filter Management** の権利プロファイルを持つ役割またはスーパーユーザーになります。

IP Filter Management の権利プロファイルは、ユーザーが作成した役割に割り当てることができます。役割の作成と役割のユーザーへの割り当てについては、『Solaris のシステム管理: セキュリティーサービス』の「RBAC の構成 (タスクマップ)」を参照してください。

2 状態統計を参照します。

```
# ipfstat -s
```

例 26–18 IP フィルタの状態統計の参照

次の例は、状態統計の参照方法を示しています。

```
# ipfstat -s
IP states added:
        0 TCP
        0 UDP
        0 ICMP
        0 hits
        0 misses
        0 maximum
        0 no memory
        0 max bucket
        0 active
        0 expired
        0 closed
State logging enabled

State table bucket statistics:
        0 in use
```

```
0.00% bucket usage
0 minimal length
0 maximal length
0.000 average length
```

▼ IP フィルタの NAT 統計を参照する方法

1 **IP Filter Management** の権利プロファイルを持つ役割またはスーパーユーザーになります。

IP Filter Management の権利プロファイルは、ユーザーが作成した役割に割り当てることができます。役割の作成と役割のユーザーへの割り当てについては、『Solaris のシステム管理: セキュリティーサービス』の「RBAC の構成 (タスクマップ)」を参照してください。

2 **NAT** 統計の参照

```
# ipnat -s
```

例 **26–19** IP フィルタの NAT 統計の参照

次の例は、NAT 統計の参照方法を示しています。

```
# ipnat -s
mapped  in      0       out     0
added   0       expired 0
no memory       0       bad nat 0
inuse   0
rules   1
wilds   0
```

▼ IP フィルタのアドレスプール統計情報を表示する方法

1 **IP Filter Management** の権利プロファイルを持つ役割またはスーパーユーザーになります。

IP Filter Management の権利プロファイルは、ユーザーが作成した役割に割り当てることができます。役割の作成と役割のユーザーへの割り当てについては、『Solaris のシステム管理: セキュリティーサービス』の「RBAC の構成 (タスクマップ)」を参照してください。

2 アドレスプール統計の参照

```
# ippool -s
```

例 26–20 IP フィルタのアドレスプール統計の参照

次の例は、アドレスプール統計の参照方法を示しています。

```
# ippool -s
Pools:  3
Hash Tables:    0
Nodes:  0
```

IP フィルタ用ログファイルの操作

表 26–6 IP フィルタログファイルの操作 (タスクマップ)

タスク	説明	手順
ログファイルの作成	IP フィルタログファイルを別個に作成する	705 ページの「IP フィルタのログファイルを設定する方法」
ログファイルの参照	`ipmon` コマンドで、状態、NAT、通常のログファイルを参照する	706 ページの「IP フィルタのログファイルを参照する方法」
パケットログバッファーの消去	`ipmon -F` コマンドでパケットログバッファーの内容を削除する	707 ページの「パケットログファイルを消去する方法」
ロギングされたパケットのファイルへの保存	あとで参照できるようにロギングされたパケットをファイルに保存する	708 ページの「ロギングされたパケットをファイルに保存する方法」

▼ IP フィルタのログファイルを設定する方法

デフォルトでは、IP フィルタのログ情報はすべて `syslogd` ファイルに記録されます。デフォルトログファイルに記録される可能性のあるほかのデータとは別個に記録するため、IP フィルタのトラフィック情報を記録するログファイルを設定することをお勧めします。次の手順を実行します。

1 **IP Filter Management の権利プロファイルを持つ役割またはスーパーユーザーになります。**

IP Filter Management の権利プロファイルは、ユーザーが作成した役割に割り当てることができます。役割の作成と役割のユーザーへの割り当てについては、『Solaris のシステム管理: セキュリティーサービス』の「RBAC の構成 (タスクマップ)」を参照してください。

2 **`/etc/syslog.conf` を編集して、次の 2 行を追加します。**

```
# Save IP Filter log output to its own file
local0.debug            /var/log/log-name
```

注 – 2 番目の行の `local0.debug` と `/var/log/`*log-name* との区切りには、Space バーではなく Tab キーを必ず使用してください。

3 新規ログファイルを作成します。

```
# touch /var/log/log-name
```

4 **system-log** サービスを再起動します。

```
# svcadm restart system-log
```

例 **26–21** IP フィルタログの作成

次の例は、`ipmon.log` を作成して IP フィルタ情報をアーカイブする方法を示しています。

`/etc/syslog.conf` に、次の記述を追加します。

```
# Save IP Filter log output to its own file
local0.debug             /var/log/ipmon.log
```

コマンド行で、次のコマンドを実行します。

```
# touch /var/log/ipmon.log
# svcadm restart system-log
```

▼ IP フィルタのログファイルを参照する方法

始める前に IP フィルタデータの記録用に、ログファイルを別個に作成することをお勧めします。705 ページの「IP フィルタのログファイルを設定する方法」を参照してください。

1 **IP Filter Management** の権利プロファイルを持つ役割またはスーパーユーザーになります。

IP Filter Management の権利プロファイルは、ユーザーが作成した役割に割り当てることができます。役割の作成と役割のユーザーへの割り当てについては、『Solaris のシステム管理: セキュリティーサービス』の「RBAC の構成 (タスクマップ)」を参照してください。

2 状態、**NAT**、または通常のログファイルを参照します。ログファイルを参照するには、適切なオプションと共に次のコマンドを入力してください。

```
# ipmon -o [S|N|I] filename
```

`S` 状態ログファイルを表示します。

`N` NAT ログファイルを表示します。

I　通常の IP ログファイルを表示します。

状態、NAT、および通常のログファイルをすべて表示するには、すべてのオプションを使用します。

```
# ipmon -o SNI filename
```

- 最初に **ipmon** デーモンを手動で停止した場合は、次のコマンドを使って状態、**NAT**、および **IP** フィルタログファイルを表示することもできます。

  ```
  # ipmon -a filename
  ```

注 – ipmon デーモンが実行中の場合は、ipmon -a 構文を使用しないでください。通常、このデーモンは、システムのブート時に自動的に起動されます。ipmon -a コマンドを実行すると、ipmon の別のコピーも開かれます。この場合、両方のコピーが同じログ情報を読み取るため、一方だけが特定のログメッセージを取得します。

ログファイルの参照については、ipmon(1M) のマニュアルページを参照してください。

例 26–22　IP フィルタのログファイルの参照

次の例は、/var/ipmon.log からの出力を示しています。

```
# ipmon -o SNI /var/ipmon.log
02/09/2004 15:27:20.606626 hme0 @0:1 p 129.146.157.149 ->
129.146.157.145 PR icmp len 20 84 icmp echo/0 IN
```

または

```
# pkill ipmon
# ipmon -aD /var/ipmon.log
02/09/2004 15:27:20.606626 hme0 @0:1 p 129.146.157.149 ->
129.146.157.145 PR icmp len 20 84 icmp echo/0 IN
```

▼ パケットログファイルを消去する方法

1. **IP Filter Management** の権利プロファイルを持つ役割またはスーパーユーザーになります。

   IP Filter Management の権利プロファイルは、ユーザーが作成した役割に割り当てることができます。役割の作成と役割のユーザーへの割り当てについては、『Solaris のシステム管理: セキュリティーサービス』の「RBAC の構成 (タスクマップ)」を参照してください。

2 パケットログバッファーの消去

```
# ipmon -F
```

例 **26–23** パケットログファイルの消去

次の例は、ログファイルが削除されたときの出力を示しています。ログファイルに何も保存されていない場合も、この例のようなレポートが出力されます。

```
# ipmon -F
0 bytes flushed from log buffer
0 bytes flushed from log buffer
0 bytes flushed from log buffer
```

▼ ロギングされたパケットをファイルに保存する方法

1 **IP Filter Management** の権利プロファイルを持つ役割またはスーパーユーザーになります。

IP Filter Management の権利プロファイルは、ユーザーが作成した役割に割り当てることができます。役割の作成と役割のユーザーへの割り当てについては、『Solaris のシステム管理: セキュリティーサービス』の「RBAC の構成 (タスクマップ)」を参照してください。

2 ロギングされたパケットをファイルへ保存します。

```
# cat /dev/ipl > filename
```

`Control-C` を入力して、コマンド行のプロンプトに戻って、このプロシージャーを中断するまで、パケットは *filename* ファイルに継続的にロギングされます。

例 **26–24** ファイルへのロギングされたパケットの保存

次の例は、ロギングされたパケットがファイルに保存されたときの結果を表します。

```
# cat /dev/ipl > /tmp/logfile
^C#

# ipmon -f /tmp/logfile
02/09/2004 15:30:28.708294 hme0 @0:1 p 129.146.157.149,33923 ->
  129.146.157.145,23 PR tcp len 20 52 -S IN
02/09/2004 15:30:28.708708 hme0 @0:1 p 129.146.157.149,33923 ->
  129.146.157.145,23 PR tcp len 20 40 -A IN
02/09/2004 15:30:28.792611 hme0 @0:1 p 129.146.157.149,33923 ->
  129.146.157.145,23 PR tcp len 20 70 -AP IN
02/09/2004 15:30:28.872000 hme0 @0:1 p 129.146.157.149,33923 ->
```

```
 129.146.157.145,23 PR tcp len 20 40 -A IN
02/09/2004 15:30:28.872142 hme0 @0:1 p 129.146.157.149,33923 ->
  129.146.157.145,23 PR tcp len 20 43 -AP IN
02/09/2004 15:30:28.872808 hme0 @0:1 p 129.146.157.149,33923 ->
  129.146.157.145,23 PR tcp len 20 40 -A IN
02/09/2004 15:30:28.872951 hme0 @0:1 p 129.146.157.149,33923 ->
  129.146.157.145,23 PR tcp len 20 47 -AP IN
02/09/2004 15:30:28.926792 hme0 @0:1 p 129.146.157.149,33923 ->
  129.146.157.145,23 PR tcp len 20 40 -A IN
.
.
(output truncated)
```

IP フィルタ構成ファイルの作成と編集

規則セットとアドレスプールを作成および変更するには、構成ファイルを直接編集する必要があります。構成ファイルは、次のような標準的な UNIX 構文規則に従っています。

- シャープ記号 (#) は、コメントを含む行を示します。
- 規則とコメントは、同一の行に共存できます。
- 規則を読みやすくするために、不要な空白を使用できます。
- 複数行に渡って規則を記述できます。行の最後のバックスラッシュ (\) は、規則が次の行に続いていることを示します。

▼ IP フィルタの構成ファイルを作成する方法

次の手順では、次のファイルの設定方法を説明します。

- パケットフィルタリングの構成ファイル
- NAT 規則の構成ファイル
- アドレスプールの構成ファイル

1. **IP Filter Management** の権利プロファイルを持つ役割またはスーパーユーザーになります。

   IP Filter Management の権利プロファイルは、ユーザーが作成した役割に割り当てることができます。役割の作成と役割のユーザーへの割り当てについては、『Solaris のシステム管理: セキュリティーサービス』の「RBAC の構成 (タスクマップ)」を参照してください。

2. 適当なファイルエディタを起動します。構成したい機能の構成ファイルを作成または編集します。

- パケットフィルタリング規則の構成ファイルを作成するには、ipf.conf ファイルを編集します。

  IP フィルタは、ipf.conf ファイルに保存したパケットフィルタリング規則を使用します。パケットフィルタリングの規則ファイルを /etc/ipf/ipf.conf ファイルに保存した場合は、システムのブート時に、このファイルがロードされます。フィルタリング規則をブート時にロードしない場合は、このファイルを別の場所に保存します。次に、ipf コマンドで規則をアクティブにします (691 ページの「別のパケットフィルタリング規則セット、または更新されたパケットフィルタリング規則セットをアクティブにする方法」を参照)。

  パケットフィルタリング規則を作成する方法については、663 ページの「IP フィルタのパケットのフィルタリング機能の使用」を参照してください。

  注 – ipf.conf ファイルが空の場合は、フィルタリングは行われません。空の ipf.conf ファイルは、次のような規則セットを持っているのと同じです。

  ```
  pass in all
  pass out all
  ```

- NAT 規則の構成ファイルを作成するには、 ipnat.conf ファイルを編集します。

  IP フィルタは、ipnat.conf ファイルに保存した NAT 規則を使用します。NAT の規則ファイルを /etc/ipf/ipnat.conf ファイルに保存した場合は、システムのブート時に、このファイルがロードされます。NAT 規則をブート時にロードしない場合は、適当な場所に ipnat.conf ファイルを保存します。次に、 ipnat コマンドで NAT 規則をアクティブ化します。

  NAT 用の規則の作成については、667 ページの「IP フィルタの NAT 機能の使用」を参照してください。

- アドレスプールの構成ファイルを作成するには、ippool.conf ファイルを編集します。

  IP フィルタは、ippool.conf ファイルに保存したアドレスのプールを使用します。アドレスプールの規則ファイルを /etc/ipf/ippool.conf ファイルに保存した場合は、システムのブート時に、このファイルがロードされます。アドレスプールをブート時にロードしない場合は、適当な場所に ippool.conf ファイルを保存します。次に、ippool コマンドでアドレスプールをアクティブ化できます。

  アドレスプールの作成については、668 ページの「IP フィルタのアドレスプール機能の使用」を参照してください。

IP フィルタの構成ファイルの例

次の例は、フィルタリング構成で使用されるパケットフィルタリング規則を示しています。

例 26–25 IP フィルタのホスト構成

この例は、elxl ネットワークインタフェースを備えたホストマシンの構成を示しています。

```
# pass and log everything by default
pass in log on bge0 all
pass out log on bge0 all

# block, but don't log, incoming packets from other reserved addresses
block in quick on bge0 from 10.0.0.0/8 to any
block in quick on bge0 from 172.16.0.0/12 to any

# block and log untrusted internal IPs. 0/32 is notation that replaces
# address of the machine running Solaris IP Filter.
block in log quick from 192.168.1.15 to <thishost>
block in log quick from 192.168.1.43 to <thishost>

# block and log X11 (port 6000) and remote procedure call
# and portmapper (port 111) attempts
block in log quick on bge0 proto tcp from any to bge0/32 port = 6000 keep state
block in log quick on bge0 proto tcp/udp from any to bge0/32 port = 111 keep state
```

この規則セットは、最初の段階では、すべてのデータが elxl インタフェースを出入りできる制限なしの規則です。2 番目の規則セットは、プライベートアドレス空間 10.0.0.0 および 172.16.0.0 からの受信パケットがファイアウォールの中に入るのをブロックします。次の規則セットは、ホストマシンからの特定の内部アドレスをブロックします。そして、最後の規則セットは、ポート 6000 およびポート 111 から受信されるパケットをブロックします。

例 26–26 IP フィルタのサーバー構成

この例は、Web サーバーとして機能するホストマシンの構成を示しています。このマシンには、eri ネットワークインタフェースがあります。

```
# web server with an eri interface
# block and log everything by default; then allow specific services
# group 100 - inbound rules
# group 200 - outbound rules
# (0/32) resolves to our IP address)
*** FTP proxy ***


# block short packets which are packets fragmented too short to be real.
block in log quick all with short
```

例 **26–26** IP フィルタのサーバー構成 *(続き)*

```
# block and log inbound and outbound by default, group by destination
block in log on eri0 from any to any head 100
block out log on eri0 from any to any head 200


# web rules that get hit most often
pass in quick on eri0 proto tcp from any \
to eri0/32 port = http flags S keep state group 100
pass in quick on eri0 proto tcp from any \
to eri0/32 port = https flags S keep state group 100


# inbound traffic - ssh, auth
pass in quick on eri0 proto tcp from any \
to eri0/32 port = 22 flags S keep state group 100
pass in log quick on eri0 proto tcp from any \
to eri0/32 port = 113 flags S keep state group 100
pass in log quick on eri0 proto tcp from any port = 113 \
to eri0/32 flags S keep state group 100


# outbound traffic - DNS, auth, NTP, ssh, WWW, smtp
pass out quick on eri0 proto tcp/udp from eri0/32 \
to any port = domain flags S keep state group 200
pass in quick on eri0 proto udp from any port = domain to eri0/32 group 100

pass out quick on eri0 proto tcp from eri0/32 \
to any port = 113 flags S keep state group 200
pass out quick on eri0 proto tcp from eri0/32 port = 113 \
to any flags S keep state group 200

pass out quick on eri0 proto udp from eri0/32 to any port = ntp group 200
pass in quick on eri0 proto udp from any port = ntp to eri0/32 port = ntp group 100

pass out quick on eri0 proto tcp from eri0/32 \
to any port = ssh flags S keep state group 200

pass out quick on eri0 proto tcp from eri0/32 \
to any port = http flags S keep state group 200
pass out quick on eri0 proto tcp from eri0/32 \
to any port = https flags S keep state group 200

pass out quick on eri0 proto tcp from eri0/32 \
to any port = smtp flags S keep state group 200


# pass icmp packets in and out
pass in quick on eri0 proto icmp from any to eri0/32  keep state group 100
pass out quick on eri0 proto icmp from eri0/32 to any keep state group 200


# block and ignore NETBIOS packets
block in quick on eri0 proto tcp from any \
to any port = 135 flags S keep state group 100
```

例 26–26 IP フィルタのサーバー構成 *(続き)*

```
block in quick on eri0 proto tcp from any port = 137 \
to any flags S keep state group 100
block in quick on eri0 proto udp from any to any port = 137 group 100
block in quick on eri0 proto udp from any port = 137 to any group 100

block in quick on eri0 proto tcp from any port = 138 \
to any flags S keep state group 100
block in quick on eri0 proto udp from any port = 138 to any group 100

block in quick on eri0 proto tcp from any port = 139 to any flags S keep state
group 100
block in quick on eri0 proto udp from any port = 139 to any group 100
```

例 26–27 IP フィルタのルーター構成

この例は、内部インタフェース ce0 と外部インタフェース ce1 を備えるルーターの構成を示しています。

```
# internal interface is ce0 at 192.168.1.1
# external interface is ce1 IP obtained via DHCP
# block all packets and allow specific services
*** NAT ***
*** POOLS ***


# Short packets which are fragmented too short to be real.
block in log quick all with short


# By default, block and log everything.
block in log on ce0 all
block in log on ce1 all
block out log on ce0 all
block out log on ce1 all


# Packets going in/out of network interfaces that aren't on the loopback
# interface should not exist.
block in log quick on ce0 from 127.0.0.0/8 to any
block in log quick on ce0 from any to 127.0.0.0/8
block in log quick on ce1 from 127.0.0.0/8 to any
block in log quick on ce1 from any to 127.0.0.0/8


# Deny reserved addresses.
block in quick on ce1 from 10.0.0.0/8 to any
block in quick on ce1 from 172.16.0.0/12 to any
block in log quick on ce1 from 192.168.1.0/24 to any
block in quick on ce1 from 192.168.0.0/16 to any


# Allow internal traffic
pass in quick on ce0 from 192.168.1.0/24 to 192.168.1.0/24
pass out quick on ce0 from 192.168.1.0/24 to 192.168.1.0/24
```

例 **26–27** IP フィルタのルーター構成 *(続き)*

```
# Allow outgoing DNS requests from our servers on .1, .2, and .3
pass out quick on ce1 proto tcp/udp from ce1/32 to any port = domain keep state
pass in quick on ce0 proto tcp/udp from 192.168.1.2 to any port = domain keep state
pass in quick on ce0 proto tcp/udp from 192.168.1.3 to any port = domain keep state


# Allow NTP from any internal hosts to any external NTP server.
pass in quick on ce0 proto udp from 192.168.1.0/24 to any port = 123 keep state
pass out quick on ce1 proto udp from any to any port = 123 keep state


# Allow incoming mail
pass in quick on ce1 proto tcp from any to ce1/32 port = smtp keep state
pass in quick on ce1 proto tcp from any to ce1/32 port = smtp keep state
pass out quick on ce1 proto tcp from 192.168.1.0/24 to any port = smtp keep state


# Allow outgoing connections: SSH, WWW, NNTP, mail, whois
pass in quick on ce0 proto tcp from 192.168.1.0/24 to any port = 22 keep state
pass out quick on ce1 proto tcp from 192.168.1.0/24 to any port = 22 keep state

pass in quick on ce0 proto tcp from 192.168.1.0/24 to any port = 80 keep state
pass out quick on ce1 proto tcp from 192.168.1.0/24 to any port = 80 keep state
pass in quick on ce0 proto tcp from 192.168.1.0/24 to any port = 443 keep state
pass out quick on ce1 proto tcp from 192.168.1.0/24 to any port = 443 keep state

pass in quick on ce0 proto tcp from 192.168.1.0/24 to any port = nntp keep state
block in quick on ce1 proto tcp from any to any port = nntp keep state
pass out quick on ce1 proto tcp from 192.168.1.0/24 to any port = nntp keep state

pass in quick on ce0 proto tcp from 192.168.1.0/24 to any port = smtp keep state

pass in quick on ce0 proto tcp from 192.168.1.0/24 to any port = whois keep state
pass out quick on ce1 proto tcp from any to any port = whois keep state


# Allow ssh from offsite
pass in quick on ce1 proto tcp from any to ce1/32 port = 22 keep state


# Allow ping out
pass in quick on ce0 proto icmp all keep state
pass out quick on ce1 proto icmp all keep state


# allow auth out
pass out quick on ce1 proto tcp from ce1/32 to any port = 113 keep state
pass out quick on ce1 proto tcp from ce1/32 port = 113 to any keep state


# return rst for incoming auth
block return-rst in quick on ce1 proto tcp from any to any port = 113 flags S/SA
```

例 **26–27** IP フィルタのルーター構成 *(続き)*

```
# log and return reset for any TCP packets with S/SA
block return-rst in log on ce1 proto tcp from any to any flags S/SA


# return ICMP error packets for invalid UDP packets
block return-icmp(net-unr) in proto udp all
```

パート V

IPMP

このパートでは、IP マルチパス (IPMP) を紹介し、IPMP を管理するための作業について説明します。IPMP は、同じリンクに接続されたシステムインタフェースの障害検出およびフェイルオーバーを行います。

◆ ◆ ◆ 第 27 章

IPMP の紹介 (概要)

IP ネットワークマルチパス (IPMP) は、同一の IP リンク上に複数のインタフェースを保持するシステムで、物理インタフェースの障害検出および透過的なネットワークアクセスフェイルオーバーを提供します。IPMP も、複数のインタフェースを保持するシステムについて、パケットの負荷分散を提供します。

この章では、次の内容について説明します。

IPMP 構成タスクについては、第 28 章「IPMP の管理 (タスク)」を参照してください。

IPMP を使用しなければならない理由

IPMP は、複数の物理インタフェースを持つシステムの信頼性、可用性、およびネットワークパフォーマンスを向上させます。物理インタフェースまたはそのインタフェースに接続しているネットワークハードウェアでは、ときどき障害が発生したり、メンテナンスが必要になったりすることがあります。従来、このような場合には、障害が発生したインタフェースに関連するどの IP アドレスを使用しても、システムに接続できなくなっていました。さらに、これらの IP アドレスを使用する既存のシステム接続も妨害されていました。

IPMP を使用すると、1 つ以上の物理インタフェースを IP マルチパスグループ、つまり「*IPMP グループ*」に構成できます。IPMP を構成すると、IPMP グループのインタフェースに障害が発生していないかどうかをシステムが自動的に監視します。グ

ループ内のインタフェースで障害が発生した場合、またはインタフェースがメンテナンスのために取り外された場合、IPMP は、障害が発生したインタフェースの IP アドレスを自動的に移行、つまり「障害を迂回」します。フェイルオーバーされたアドレスは、障害が発生したインタフェースの IPMP グループ内の機能中のインタフェースが受け取ります。IPMP のフェイルオーバー機能は、接続を保持し、既存の接続の切断を防止します。さらに、IPMP は、ネットワークトラフィックを自動的に IPMP グループ内のインタフェースのセットに分散することによって、ネットワークパフォーマンス全体を向上させます。このプロセスは「負荷分散」と呼ばれます。

Oracle Solaris IPMP コンポーネント

Oracle Solaris IPMP には、次のソフトウェアが含まれます。

- in.mpathd(1M) のマニュアルページに詳しく説明されている in.mpathd デーモン。
- /etc/default/mpathd 構成ファイル。これについても、in.mpathd(1M) のマニュアルページに説明があります。
- ifconfig(1M) のマニュアルページに説明がある IPMP 構成の ifconfig オプション。

マルチパスデーモン in.mpathd

in.mpathd デーモンはインタフェース障害を検出し、障害経路の迂回や回復した経路への復帰に対するさまざまな手続を実装します。in.mpathd は、障害または修復を検出すると、ioctl を送信して、フェイルオーバーまたは回復した経路への復帰を実行します。ioctl を実行する ip カーネルモジュールは、ネットワークアクセスのフェイルオーバーを、透過的かつ自動的に行います。

注 – ネットワークインタフェースカードの同じセットで IPMP を使用している間は、代替パスを使用しないでください。同様に、代替パスを使用している間は、IPMP を使用しないでください。異なるインタフェースセットの場合は、代替パスと IPMP を同時に使用できます。代替パスについては、『Sun Enterprise Server Alternate Pathing 2.3.1 User Guide』を参照してください。

---

in.mpathd デーモンは、IPMP グループの一部であるインタフェースすべてに検査信号を送信して、障害と回復を検出します。in.mpathd デーモンも、グループに属する各インタフェースで RUNNING フラグを監視することによって障害や回復を検出します。詳細は、in.mpathd(1M) のマニュアルページを参照してください。

注 – IPMP データアドレスを管理するために DHCP はサポートされていません。これらのアドレスに対して DHCP を使用しようとすると、DHCP は最終的にこれらのアドレスの制御を放棄します。データアドレスには DHCP を使用しないでください。

IPMP の用語と概念

この節では、このドキュメントの IPMP の章を通して使用される用語と概念を紹介します。

IP リンク

IPMP の用語では、「*IP* リンク」は、ノードがインターネットプロトコル群のデータリンク層で通信を行う通信設備またはメディアです。IP リンクのタイプには、単純な Ethernet、ブリッジ Ethernet、ハブ、または ATM (Asynchronous Transfer Mode) ネットワークなどがあります。IP リンクは、1 つ以上の IPv4 サブネット番号、および適用できる場合は、1 つ以上の IPv6 サブネット接頭辞を持つことができます。同じサブネット番号またはネットワーク接頭辞を複数の IP リンクに割り当てることはできません。ATM LANE では、IP リンクは 1 つのエミュレートされた LAN (Local Area Network) です。ARP (Address Resolution Protocol) を使用する場合、ARP プロトコルの範囲は、1 つの IP リンクです。

注 – RFC 2460、『Internet Protocol, Version 6 (IPv6) Specification』などのその他の IP 関連文書では、「*IP* リンク」の代わりに「リンク」という用語を使用します。パート 6 では、IEEE 802 との混乱を避けるために、「*IP* リンク」という用語を使用します。IEEE 802 では、「リンク」は、Ethernet ネットワークインタフェースカード (NIC) から Ethernet スイッチへの一本の配線を指します

物理インタフェース

「物理インタフェース」は、IP リンクにシステム接続を提供します。この接続は、しばしばデバイスドライバや NIC として実装されます。システムが同じリンクに複数のインタフェースを接続している場合は、IPMP を構成して、インタフェースの 1 つで障害が発生した場合に障害を迂回させることができます。物理インタフェースについては、727 ページの「IPMP インタフェースの構成」を参照してください。

ネットワークインタフェースカード

「ネットワークインタフェースカード」(NIC) は、システム内に組み込むことができるネットワークアダプタです。また、NIC は、システムから IP リンクへのインタフェースとして機能する別のカードの場合もあります。複数の物理インタフェースを持つ NIC もあります。たとえば、`qfe` NIC は、`qfe0` - `qfe3` などの 4 つのインタフェースを持つことができます。

IPMP グループ

IP マルチパスグループ、つまり「*IPMP*」グループは、同じ IPMP グループ名で構成された同じシステム上の 1 つ以上の物理インタフェースで構成されます。IPMP グループ内のインタフェースは、同じ IP リンクに接続してください。同じ (空文字以外の) 文字列の IPMP グループ名は、グループ内のすべてのインタフェースを識別します。NIC のタイプが同じである限り、違った速度の NIC インタフェースを同じ IPMP グループ内に配置できます。たとえば、100M ビット Ethernet NIC のインタフェースと 1G ビット Ethernet NIC のインタフェースを同じグループに構成できます。別の例として、2 つの 100M ビット Ethernet NIC を持っているとします。インタフェースのどちらかを 10M ビットに下げて構成しても、この 2 つのインタフェースを同じ IPMP グループに配置できます。

メディアタイプの異なる 2 つのインタフェースを 1 つの IPMP グループに配置することはできません。たとえば、ATM インタフェースを Ethernet インタフェースと同じグループに配置することはできません。

障害検出とフェイルオーバー

「障害検出」は、インターネット層デバイスへのインタフェースまたはインタフェースからのパスが機能しなくなったことの検出プロセスです。IPMP は、インタフェースでの障害を検出する機能をシステムに提供します。

IPMP は、次のタイプの通信障害を検出します。

- インタフェースの送信または受信パスで障害が発生した。
- IP リンクとのインタフェースの接続が停止した。
- スイッチ上のポートがパケットを送信または受信していない。
- IPMP グループ内の物理インタフェースがシステムのブート時に存在しない。

障害を検出すると、IPMP はフェイルオーバーを開始します。「フェイルオーバー」は、障害が発生したインタフェースから同じグループ内の機能中の物理インタフェースにネットワークアクセスを切り替える自動プロセスです。ネットワークアクセスには、IPv4 のユニキャスト、マルチキャスト、およびブロードキャストと、IPv6 のユニキャストとマルチキャストが含まれます。フェイルオーバーは、IPMP グループ内に複数のインタフェースを構成している場合のみ実行できます。フェイルオーバープロセスにより、ネットワークへのアクセスは中断することなく継続されます。

回復の検出と回復した経路への復帰

「回復の検出」は、障害後、 NIC または NIC からインターネット層デバイスへのパスが正しく機能し始めたときの検出プロセスです。NIC が回復されたことを検出すると、IPMP は、ネットワークアクセスを回復されたインタフェースに戻すプロセスである「回復した経路への復帰」を実行します。回復検出が行われるのは、回復した経路への復帰が有効になっている場合のみです。詳細については、731 ページの「物理インタフェースの回復検出」を参照してください。

ターゲットシステム

検査信号ベースの障害検出は、「ターゲットシステム」を使用して、インタフェースの状態を判断します。各ターゲットシステムは、IPMP グループのメンバーと同じ IP リンクに接続します。ローカルシステムの `in.mpathd` デーモンは、ICMP 検査信号メッセージを各ターゲットシステムに送信します。検査信号メッセージは、IPMP グループ内の各インタフェースの状態を判断するのに役立ちます。

検査信号ベースの障害検出での対象システムの使用については、730 ページの「検査信号ベースの障害検出」を参照してください。

出力負荷の分散

IPMP を構成すると、出力ネットワークパケットは、パケットの順番に影響を与えることなく、複数の NIC に分散されます。このプロセスは、「負荷分散」として知られています。負荷分散の結果、より高いスループットを達成できます。ただし、負荷分散が行われるのは、データが複数の接続を経由して複数の標識に送信される場合だけです。

動的再構成 (DR)

「動的再構成」(DR) は、既存の操作にほとんど、またはまったく影響を与えることなく、システムを実行しながらシステムを再構成する機能です。Sun プラットフォームの一部は、DR をサポートしていません。Sun プラットフォームの一部は、特定のタイプのハードウェアの DR だけをサポートする場合があります。NIC の DR をサポートするプラットフォームでは、IPMP を使用して透過的にネットワークアクセスの障害を迂回し、システムのネットワークアクセスは中断なしで継続させることができます。

IPMP がどのように DR をサポートするかについては、733 ページの「IPMP と動的再構成」を参照してください。

IPMP の基本要件

IPMP は Oracle Solaris に組み込まれており、特殊なハードウェアを必要としません。Oracle Solaris でサポートされているインタフェースはすべて、IPMP と使用できます。ただし、IPMP はネットワーク構成とトポロジに次の要件を課します。

- IPMP グループ内のすべてのインタフェースは、一意の MAC アドレスを持つ必要があります。

  デフォルトでは、SPARC ベースのシステムのネットワークインタフェースは、すべてで 1 つの MAC アドレスを共有します。よって、SPARC ベースのシステムで IPMP を使用するには、デフォルトを明示的に変更する必要があります。詳細については、739 ページの「IPMP グループの計画を立てる方法」を参照してください。

- 1 つの IPMP グループ内のすべてのインタフェースは、同じメディアタイプでなければなりません。詳細については、722 ページの「IPMP グループ」を参照してください。
- 1 つの IPMP グループ内のすべてのインタフェースは、同じ IP リンク上になければなりません。詳細については、722 ページの「IPMP グループ」を参照してください。

注 – 同じリンク層 (L2 またはレイヤー 2) ブロードキャストドメイン上に複数の IPMP グループを構成する操作はサポートされていません。通常、L2 ブロードキャストドメインは特定のサブネットに対応します。したがって、サブネットごとに IPMP グループを 1 つだけ構成する必要があります。

- 障害検出に関する要件によって、特定のタイプのネットワークインタフェースを使用するか、各ネットワークインタフェースに追加の IP アドレスを構成する必要がある場合もあります。729 ページの「リンクベースの障害検出」と730 ページの「検査信号ベースの障害検出」を参照してください。

IPMP アドレス指定

IPMP 障害検出は、IPv4 ネットワークと IPv4 および IPv6 のデュアルスタックネットワークで構成できます。IPMP で構成されたインタフェースは、 データアドレスと検査用アドレスという 2 種類のアドレスをサポートしています。

データアドレス

「データアドレス」は、ブート時または手動で `ifconfig` コマンドによって NIC のインタフェースに割り当てられる従来の IPv4 および IPv6 アドレスです。標準 IPv4 と、適用できる場合は、インタフェースを通した IPv6 パケットトラフィックは、「データトラフィック」とみなされます。

検査用アドレス

「検査用アドレス」は、`in.mpathd`デーモンが使用する IPMP 固有のアドレスです。検査信号ベースの障害と回復の検出を使用するインタフェースの場合、そのインタフェースは 1 つ以上の検査用アドレスで構成する必要があります。

注 – DR を使用してプローブベースの障害検出を使用する場合のみ、テストアドレスを構成する必要があります。

in.mpathd デーモンは、検査用アドレスを使用して、「検査信号トラフィック」とも呼ばれる ICMP 検査信号を IP リンク上のほかのターゲットと交換します。検査信号トラフィックは、インタフェースで障害が発生していないかどうかなど、インタフェースと NIC のステータスを判断するのに役立ちます。検査信号は、インタフェースとの送受信パスが正しく機能していることを確認します。

各インタフェースは、IP 検査用アドレスで構成できます。デュアルスタックネットワークのインタフェースの場合、IPv4 検査用アドレス、IPv6 検査用アドレス、または IPv4 と IPv6 検査用アドレスの両方を構成できます。

インタフェースで障害が発生すると、in.mpathd がそれ以降の回復をチェックするために検査信号を送信し続けることができるように、検査用アドレスは障害が発生したインタフェースに留まります。アプリケーションが間違って使用しないように、検査用アドレスは具体的に構成しなければなりません。詳細については、726 ページの「アプリケーションによる検査用アドレス使用の防止」を参照してください。

検査信号ベースの障害検出については、730 ページの「検査信号ベースの障害検出」を参照してください。

IPv4 検査用アドレス

一般的に、どの IPv4 アドレスもサブネット上で検査用アドレスとして使用できます。IPv4 検査用アドレスは、ルートが指定できなくても構いません。IPv4 アドレスは、多くのサイトでは限定リソースなので、ルート指定できない RFC 1918 プライベートアドレスを検査用アドレスとして指定したい場合もあります。 in.mpathd デーモンは、ICMP 検査信号を検査用アドレスと同じサブネットのホストとしか交換しません。RFC 1918 形式の検査用アドレスを使用していない場合は、IP リンク上のほかのシステム (ルーターが望ましい) を適切な RFC 1918 サブネットのアドレスで必ず構成してください。この構成により、in.mpathd デーモンは、ターゲットシステムと正常に検査信号を交換できます。

IPMP の例は、192.168.0/24 ネットワークの RFC 1918 アドレスを IPv4 検査用アドレスとして使用します。RFC 1918 プライベートアドレスの詳細については、RFC 1918, Address Allocation for Private Internets (http://www.ietf.org/rfc/rfc1918.txt?number=1918) を参照してください。

IPv4 検査用アドレスを構成するには、タスク741 ページの「複数のインタフェースを持つ IPMP グループを構成する方法」を参照してください。

IPv6 検査用アドレス

有効な IPv6 検査用 IP アドレスは、物理インタフェースのリンクローカルアドレスだけです。IPMP 検査用 IP アドレスとして機能する別の IPv6 アドレスは必要ありません。IPv6 リンクローカルアドレスは、インタフェースのメディアアクセスコントロール (MAC) アドレスに基づいています。リンクローカルアドレスは、インタフェースがブート時に IPv6 を使用できるようになったり、`ifconfig` によって手動で構成された場合に、自動的に構成されます。

インタフェースのリンクローカルアドレスを識別するには、IPv6 が有効なノードで `ifconfig` *interface* コマンドを実行します。リンクローカル接頭辞 `fe80` で始まるアドレスの出力をチェックします。次の `ifconfig` の `NOFAILOVER` フラグは、`hme0` インタフェースのリンクローカルアドレス `fe80::a00:20ff:feb9:17fa/10` が検査用アドレスとして使用されていることを示しています。

```
hme0: flags=a000841<UP,RUNNING,MULTICAST,IPv6,NOFAILOVER> mtu 1500 index 2
          inet6 fe80::a00:20ff:feb9:17fa/10
```

リンクローカルアドレスについては、81 ページの「リンクローカルユニキャストアドレス」を参照してください。

IPMP グループですべてのグループのインタフェースに IPv4 と IPv6 の両方が使用される場合には、別個の IPv4 検査用アドレスを構成する必要はない場合があります。`in.mpathd` デーモンは、IPv6 リンクローカルアドレスを検査用アドレスとして使用します。

IPv6 検査用アドレスを作成するには、タスク741 ページの「複数のインタフェースを持つ IPMP グループを構成する方法」を参照してください。

アプリケーションによる検査用アドレス使用の防止

検査用アドレスの構成後、アドレスがアプリケーションによって使用されないことを確認する必要があります。それ以外の場合、インタフェースで障害が発生しても、検査用アドレスはフェイルオーバー操作でフェイルオーバーできないので、アプリケーションを操作できなくなります。IP が一般的なアプリケーションに対して検査用アドレスを選択しないことを確認するために、検査用アドレスを `deprecated` とマークします。

deprecated (推奨されない) と指定したアドレスは、アプリケーションで明示的に指定されていない限り、通信のソースアドレスとしては選択されません。`in.mpathd` デーモンは、検査信号トラフィックを送受信するためにこのようなアドレスを明示的に指定します。ただし、アプリケーションがアドレスを明示的に指定していない場合、またインタフェース上で `UP` とマークされた唯一のアドレスが deprecated (非推奨) のマークも付けられている場合は、最後の手段として、そのアドレスをソースアドレスとして使用します。

注 – フェイルオーバーおよびフェイルバックの場合、重複アドレス検出が継続して実行中である間は、ソースアドレスとして推奨されないアドレスを使用するパケットをアプリケーションが受信する可能性があります。これは予想される動作です。一般的に、DAD が終了した後は、推奨されないアドレスがアプリケーションによって処理されることはありません。ただし、TCP パケットでまれな例外が観測される場合があります。TCP 接続が特定のソースアドレスを選択した後、その接続が継続している間は、そのアドレスを使用を変更できません。この期間は長時間に及ぶことがあります。このようなエッジケースでは、DAD が完了した後でも、アプリケーションが推奨されないアドレスを引き続き使用する場合がある可能性が存在します。

IPv6 リンクローカルアドレスは通常ネームサービス内にないので、DNS と NIS アプリケーションは通信のリンクローカルアドレスを使用しません。結果として、IPv6 リンクローカルアドレスを `deprecated` とマークする必要はなくなります。

IPv4 検査用アドレス を DNS および NIS ネームサービステーブルに配置しないでください。通常はネームサービステーブルには追加されません。

IPMP インタフェースの構成

IPMP 構成は、通常同じ IP リンクに接続された同じシステムの複数の物理インタフェースで構成されます。これらの物理インタフェースは、同じ NIC 上にある場合とない場合があります。これらのインタフェースは、同じ IPMP グループのメンバーとして構成されます。システムは、2 番目の IP リンクに追加インタフェースを持つので、これらのインタフェースを別の IPMP グループとして構成する必要があります。

単独インタフェースは、それ自体の IPMP グループ内で構成できます。単独インタフェースIPMP グループは、複数のインタフェースを持つ IPMP グループと同じように動作します。ただし、インタフェースが 1 つだけの IPMP グループでは、フェイルオーバーと回復した経路への復帰は実行できません。

IP インタフェースからグループを構成するのと同じ手順を使用して、VLAN を IPMP グループに構成することもできます。手順については、741 ページの「IPMP グループの構成」を参照してください。VLAN を IPMP グループに構成する際には、723 ページの「IPMP の基本要件」に記載されているのと同じ要件が適用されます。

注意 – VLAN を IPMP グループとして構成するときに、VLAN の命名に使用される規則によってエラーが生じることがあります。VLAN 名の詳細については、160 ページの「VLAN タグと物理接続点」 in 『System Administration Guide: IP Services』を参照してください。`bge1000`、`bge1001`、`bge2000`、`bge2001` という 4 つの VLAN の例を考えてみます。IPMP の実装では、これらの VLAN を次のようにグループ化する必要があります。つまり、`bge1000` と `bge1001` は同じ VLAN 1 の 1 つのグループに属し、`bge2000` と `bge2001` は同じ VLAN 2 の別のグループに属すことが必要です。VLAN 名が原因で、たとえば `bge1000` と `bge2000` など、異なるリンクに属している VLAN を 1 つの IPMP グループに混在させるといった誤りが発生しやすくなっています。

IPMP グループ内の予備インタフェース

IPMP グループ内の「予備インタフェース」は、グループ内のほかのインタフェースで障害が発生しない限り、データトラフィックには使用されません。障害が発生すると、障害が発生したインタフェースのデータアドレスが予備インタフェースに移行されます。移行後、障害が発生したインタフェースが回復されるまで、予備インタフェースはほかのアクティブなインタフェースと同じように扱われます。一部の障害では、待機インタフェースが選択されないことがあります。そのかわり、フェイルオーバーで、待機インタフェースより少ないデータアドレスを持ち、UP として構成されたアクティブなインタフェースが選択されます。

待機インタフェースには検査用アドレスだけを構成します。IPMP では、`ifconfig` コマンドによって `standby` として構成したインタフェースにデータアドレスを追加することはできません。このような構成を作成しようとしても失敗します。同様に、すでにデータアドレスを持っているインタフェースを `standby` として構成しても、それらのアドレスは、自動的に IPMP グループ内の別のインタフェースにフェイルオーバーされます。これらの制限により、インタフェースを `standby` として設定する前に、`ifconfig` コマンドを使用して検査用アドレスを `-deprecated` および `failover` としてマークする必要があります。待機インタフェースを構成するには、748 ページの「IPMP グループの待機インタフェースを構成する方法」を参照してください。

一般的な IPMP インタフェースの構成

724 ページの「IPMP アドレス指定」のとおり、IPMP グループ内のインタフェースは、インタフェースの構成によって、通常のデータトラフィックと検査信号トラフィックを処理します。`ifconfig` コマンドの IPMP オプションを使用して、構成を行います。

「アクティブなインタフェース」とは、データトラフィックと検査信号トラフィックの両方を転送する物理インタフェースです。741 ページの「複数のインタフェースを持つ IPMP グループを構成する方法」または750 ページの「単一インタフェースの IPMP グループを構成する方法」のどちらかのタスクを実行すると、インタフェースが「アクティブ」として構成されます。

次に IPMP 構成の一般的な種類を 2 つ示します。

アクティブ-アクティブ構成	両方のインタフェースが「アクティブ」である 2 つのインタフェースを持つ IPMP グループです。つまり、検査信号とデータの両方のトラフィックが送信されている可能性があります。
アクティブ-待機構成	一方のインタフェースが「待機」として構成されている、2 つのインタフェースを持つ IPMP グループです。

インタフェースのステータスチェック

インタフェースのステータスは、`ifconfig` インタフェース コマンドを発行してチェックできます。`ifconfig` ステータスレポートの一般的な情報については、211 ページの「特定のインタフェースに関する情報を入手する方法」を参照してください。

たとえば、`ifconfig` コマンドを使用して、待機インタフェースのステータスを取得できます。待機インタフェースがデータアドレスのホストになっていない場合は、そのインタフェースのステータスには `INACTIVE` フラグが付いています。このフラグは、`ifconfig` の出力の インタフェースのステータス行で見ることができます。

IPMP 障害検出とリカバリ機能

`in.mpathd`デーモンは次の種類の障害検出を処理します。

- リンクベースの障害検出 (NIC ドライバがサポートしている場合)
- 検査信号ベースの障害検出 (検査用アドレスが構成されている場合)
- ブート時に不足しているインタフェースの検出

`in.mpathd` デーモンがどのようにインタフェース障害の検出を処理するかについては、`in.mpathd(1M)` のマニュアルページに詳細に説明されています。

リンクベースの障害検出

リンクベースの障害検出は、インタフェースでサポートされている場合は常に有効です。次の Sun ネットワークドライバは Oracle Solaris の現在のリリースでサポートされています。

- `hme`
- `eri`
- `ce`
- `ge`

- bge
- qfe
- dmfe
- e1000g
- igb
- ixgb
- nge
- nxge
- rge
- xge

サン以外のインタフェースがリンクベースの障害検出をサポートしているかどうかを判断するには、メーカーのマニュアルを参照してください。

これらのネットワークインタフェースドライバは、インタフェースのリンク状態を監視し、リンク状態が変わったときにネットワークサブシステムに通知します。変更を通知されると、ネットワークサブシステムは、インタフェースの RUNNING フラグを適宜設定または解除します。インタフェースの RUNNING フラグが解除されたことを検出すると、デーモンは即座にインタフェースに障害があるものとみなします。

検査信号ベースの障害検出

in.mpathd デーモンは、検査用アドレスを持つ IPMP グループの各インタフェースで検査信号ベースの障害検出を実行します。検査信号ベースの障害検出では、検査用アドレスを使用して ICMP 検査信号メッセージを送受信します。これらのメッセージは、インタフェースを経由して同じ IP リンクの 1 つ以上のターゲットシステムに届きます。検査用アドレスの紹介については、724 ページの「検査用アドレス」を参照してください。検査用アドレスの構成については、741 ページの「複数のインタフェースを持つ IPMP グループを構成する方法」を参照してください。

in.mpathd デーモンは、動的に検査信号を送信するターゲットシステムを検出します。IP リンクに接続されているルーターは、自動的に検査信号のターゲットとして選択されます。リンクにルーターがない場合、in.mpathd は検査信号をリンクの隣接ホストに送信します。ターゲットシステムとして使用するホストの選択にあたっては、すべてのホストを意味するマルチキャストアドレス (IPv4 では 224.0.0.1、IPv6 では ff02::1) にマルチキャストパケットが送信されます。検査信号は、ICMP エコーパケットに応答する最初のいくつかのホストに送信されます。in.mpathd が ICMP エコーパケットに応答したルーターまたはホストを検出できなかった場合、in.mpathd は検査信号ベースの障害を検出できません。

ホストルートを使用して、in.mpathd が使用するターゲットシステムのリストを明示的に構成できます。手順については、746 ページの「ターゲットシステムの構成」を参照してください。

IPMP グループの各インタフェースが正常に機能するかどうかを確認するために、in.mpathd は、IPMP グループのすべてのインタフェースを通してすべてのターゲットに個別に検査信号を送信します。連続する 5 つの検査信号に対して応答がない場合、in.mpathd はそのインタフェースに障害があるものとみなします。検査信号を発信する頻度は、「障害検出時間」に依存します。障害検出時間のデフォルト値は 10 秒です。ただし、障害検出時間は /etc/default/mpathd ファイルで調整できます。手順については、759 ページの「/etc/default/mpathd ファイルを構成する方法」を参照してください。

回復検出時間が 10 秒の場合、検査信号を発信する頻度はおよそ 2 秒に 1 度になります。最短の回復検出時間は障害検出時間の倍の時間となり、デフォルトは 20 秒です。これは、検査信号に対する応答を連続して 10 回受け取る必要があるためです。障害および回復検出時間は、検査信号ベースの障害検出だけに適用されます。

注 – VLAN から構成される IPMP グループでは、リンクベースの障害検出が物理リンクごとに実装されるため、そのリンク上のすべての VLAN に影響します。検査信号ベースの障害検出は、VLAN リンクごとに実行されます。たとえば、bge0/bge1 と bge1000/bge1001 は、1 つのグループに構成されます。bge0 のケーブルが取り外されると、リンクベースの障害検出では、即時の障害発生場所として bge0 と bge1000 の両方が報告されます。ただし、bge0 上のすべての検査ターゲットが到達不可能になった場合は、障害発生場所として bge0 のみが報告されます。これは、 bge1000 は自身の VLAN 上の検査ターゲットを持つためです。

グループ障害

「グループ障害」は、IPMP グループ内のすべてのインタフェースで同じ時間に障害が起こったと考えられる場合に、発生します。in.mpathd デーモンは、グループ障害に対するフェイルオーバーは実行しません。これは、すべてのターゲットシステムで同時に障害が発生した場合も同様です。この場合 in.mpathd は、現在のすべてのターゲットシステム選択を取り消し、新しくターゲットシステムを見つけます。

物理インタフェースの回復検出

in.mpathd デーモンが回復するインタフェースを考慮できるように、インタフェースに RUNNING フラグを設定する必要があります。検査信号ベースの障害検出を使用する場合、in.mpathdデーモンは、インタフェースが回復したとみなされる前に、10 の連続する検査信号パケットへの応答を受信する必要があります。インタフェースが回復したとみなされると、別のインタフェースにフェイルオーバーしていたアドレスが、回復したインタフェースに復帰されます。障害前に「アクティブ」として構成されていたインタフェースは、回復後、トラフィックの送受信を再開できます。

インタフェースのフェイルオーバー時の処理

次の 2 つの例は、一般的な構成とインタフェースの障害時に構成がどのように自動的に変更されるかを示します。hme0 インタフェースに障害が発生すると、すべてのデータアドレスが hme0 から hme1 に移されます。

例 27–1 インタフェースに障害が発生する前のインタフェース構成

```
hme0: flags=9000843<UP,BROADCAST,RUNNING,MULTICAST,IPv4>
     mtu 1500 index 2
     inet 192.168.85.19 netmask ffffff00 broadcast 192.168.85.255
     groupname test
hme0:1: flags=9000843<UP,BROADCAST,RUNNING,MULTICAST,DEPRECATED,IPv4,NOFAILOVER>
     mtu 1500
     index 2 inet 192.168.85.21 netmask ffffff00 broadcast 192.168.85.255
hme1: flags=9000843<UP,BROADCAST,RUNNING,MULTICAST,IPv4> mtu 1500 index 2
8     inet 192.168.85.20 netmask ffffff00 broadcast 192.168.85.255
     groupname test
hme1:1: flags=9000843<UP,BROADCAST,RUNNING,MULTICAST,DEPRECATED,IPv4,NOFAILOVER>
     mtu 1500
     index 2 inet 192.168.85.22 netmask ffffff00 broadcast 192.168.85.255
hme0: flags=a000841<UP,RUNNING,MULTICAST,IPv6,NOFAILOVER> mtu 1500 index 2
     inet6 fe80::a00:20ff:feb9:19fa/10
     groupname test
hme1: flags=a000841<UP,RUNNING,MULTICAST,IPv6,NOFAILOVER> mtu 1500 index 2
     inet6 fe80::a00:20ff:feb9:1bfc/10
     groupname test
```

例 27–2 インタフェースに障害が発生したあとのインタフェース構成

```
hme0: flags=19000842<BROADCAST,RUNNING,MULTICAST,IPv4,
      NOFAILOVER,FAILED> mtu 0 index 2
      inet 0.0.0.0 netmask 0
      groupname test
hme0:1: flags=19040843<UP,BROADCAST,RUNNING,MULTICAST,DEPRECATED,IPv4,
      NOFAILOVER,FAILED> mtu 1500 index 2
      inet 192.168.85.21 netmask ffffff00 broadcast 10.0.0.255
hme1: flags=9000843<UP,BROADCAST,RUNNING,MULTICAST,IPv4> mtu 1500 index 2
      inet 192.168.85.20 netmask ffffff00 broadcast 192.168.85.255
      groupname test
hme1:1: flags=9000843<UP,BROADCAST,RUNNING,MULTICAST,DEPRECATED,IPv4,
      NOFAILOVER> mtu 1500
      index 2 inet 192.168.85.22 netmask ffffff00 broadcast 10.0.0.255
hme1:2: flags=1000843<UP,BROADCAST,RUNNING,MULTICAST,IPv4> mtu 1500 index 6
      inet 192.168.85.19 netmask ffffff00 broadcast 192.168.18.255
hme0: flags=a000841<UP,RUNNING,MULTICAST,IPv6,NOFAILOVER,FAILED> mtu 1500 index 2
      inet6 fe80::a00:20ff:feb9:19fa/10
      groupname test
hme1: flags=a000841<UP,RUNNING,MULTICAST,IPv6,NOFAILOVER> mtu 1500 index 2
      inet6 fe80::a00:20ff:feb9:1bfc/10
      groupname test
```

上記の例では、このインタフェースに障害が発生したことを示す FAILED フラグが hme0 に設定されています。また、hme1:2 が新しく作成されているのがわかります。hme0 の構成は hme1:2 に引き継がれました。アドレス192.168.85.19 は、hme1 によってアクセス可能になります。

192.168.85.19 と関連するマルチキャストメンバーシップは、そのままパケットを受信できますが、パケットの受信は hme1 を通して行われます。アドレス 192.168.85.19 が hme0 から hme1 にフェイルオーバーされると、ダミーアドレス 0.0.0.0 が hme0 に作成されます。ダミーアドレスは、hme0 を引き続きアクセスできる状態に保つために作成されます。hme0 がなければ、hme0:1 は存在できません。ダミーアドレスは、回復した経路への復帰時に削除されます。

同様に、IPv6 アドレスが hme0 から hme1 へ移されています。IPv6 では、マルチキャストメンバーシップはインタフェースインデックスに関連付けられています。マルチキャストメンバーシップも、障害経路の迂回処理により、hme0 から hme1 へ移されます。 in.ndpd によって構成されたすべてのアドレスも移動します。この動作は、上記の例には示されていません。

in.mpathd デーモンは引き続き、障害が発生したインタフェースの hme0 を通して検査を行います。デーモンは、デフォルトの回復検出時間 20 秒の間に連続して 10 回の応答を受け取った時点で、インタフェースが回復されたものとみなします。RUNNING フラグも hme0 で設定されるので、デーモンは回復した経路への復帰を呼び出します。回復した経路への復帰が行われると、元の構成が再びリストアされます。

障害時および回復時にコンソールに記録されるすべてのエラーメッセージの説明については、in.mpathd(1M) のマニュアルページを参照してください。

IPMP と動的再構成

動的再構成 (DR) 機能によって、システムの実行中にインタフェースなどのシステムハードウェアを再構成できます。この節では、DR が IPMP とどのように相互運用できるかについて説明します。

NIC の DR をサポートするシステム上では、IPMP を使用して接続を保持したり、既存の接続の切断を防止できます。DR をサポートし、IPMP を使用するシステムの NIC は、安全に接続、切断、または再接続できます。これが可能なのは、IPMP が RCM (Reconfiguration Coordination Manager) フレームワークと統合されているからです。「*RCM*」は、システムコンポーネントの動的再構成を管理します。

一般的には、cfgadm コマンドを使用して、DR 操作を実行します。ただし、ほかの方法で動的再構成を行うプラットフォームもあります。詳細は、お使いのプラットフォームのマニュアルを参照してください。DR に関する具体的な文書は、次のリソースから得ることができます。

表 27–1　動的再構成の文書リソース

説明	参照先
cfgadm コマンドの詳細情報	cfgadm(1M) のマニュアルページ
Sun Cluster 環境での DR に関する具体的な情報	Sun Cluster 3.1 System Administration Guide
Sun Fire 環境での DR に関する具体的な情報	Sun Fire 880 の動的再構成に関するガイド
DR と cfgadm コマンドに関する紹介情報	『Oracle Solaris の管理: デバイスとファイルシステム』の第 4 章「デバイスの動的構成 (タスク)」
DR をサポートするシステムでの IPMP グループの管理タスク	755 ページの「動的再構成をサポートするシステムでの障害が発生した物理インタフェースの交換」

NIC の接続

How to Configure an IPMP Group With Multiple Interfacesの説明どおり、741 ページの「複数のインタフェースを持つ IPMP グループを構成する方法」 コマンドを使用して、IPMP グループにいつでもインタフェースを追加できます。よって、システムブート後に接続したシステムコンポーネント上のすべてのインタフェースは plumb され、既存の IPMP グループに追加されます。また、適当であれば、新たに追加したインタフェースを独自の IPMP グループで構成することも可能です。

これらのインタフェースとこれらに構成されたデータアドレスは、IPMP グループによって即座に使用できます。ただし、システムがリブート後、自動的にインタフェースを構成し、使用するようにするには、新しいインタフェースごとに /etc/hostname.*interface* ファイルを作成する必要があります。手順については、153 ページの「システムインストール後に物理インタフェースを構成する方法」を参照してください。

インタフェースの接続時に、/etc/hostname.*interface* ファイルがすでに存在する場合は、RCM は、このファイルの内容に従って、自動的にインタフェースを構成します。よって、インタフェースは、システムブート後に受け取るのと同じ構成を受け取ります。

NIC の切断

NIC を含むシステムコンポーネントを切断するすべての要求は、まず接続性を保持できるかどうかチェックされます。たとえば、デフォルトでは、IPMP グループ外の NIC を切断することはできません。IPMP グループ内の機能中のインタフェースだけを含む NIC も切断できません。ただし、システムコンポーネントを削除しなければならない場合は、cfgadm(1M) のマニュアルページに説明されている cfgadm の -f オプションを使用して、この動作を無効にできます。

チェックが成功すると、切断された NIC に関連するデータアドレスは、切断された NIC で障害が発生した場合のように、同じグループ内の機能中の NIC にフェイルオーバーされます。 NIC が切断されると、NIC のインタフェースのすべての検査用アドレスの構成が解除されます。次に、NIC はシステムを unplumb します。これらの手順のいずれかが失敗した場合、または同じシステムコンポーネントのその他のハードウェアの DR で障害が発生した場合は、前の構成が元の状態にリストアされます。ユーザーは、このイベントに関するステータスメッセージを受け取るはずです。それ以外の場合、切断要求は正常に完了しています。システムからコンポーネントを削除できます。既存の接続は切断されません。

NIC の再接続

RCM は、実行中のシステムから切断された NIC と関連する構成情報を記録します。結果として、RCM は、新しい NIC の接続と同様に、以前切断された NIC の再接続を扱います。つまり、RCM は plumb することだけを行います。

ただし、再接続された NIC は、通常既存の `/etc/hostname.`*interface* ファイルで指定された構成にフェイルオーバーされます。この場合、RCM は、既存の `/etc/hostname.`*interface* ファイルで指定された構成にフェイルオーバーされます。さらに、RCM は、再接続されたインタフェースに元々あった各データアドレスを `in.mpathd`デーモンに通知します。よって、再接続されたインタフェースが正しく機能するようになると、そのすべてのデータアドレスが、回復時のように再接続されたインタフェースに復帰されます。

再接続されている NIC に `/etc/hostname.`*interface*ファイルがない場合は、構成情報は使用できません。RCM は、インタフェースの構成方法に関する情報をまったく持ちません。このため、以前別のインタフェースにフェイルオーバーされたアドレスが回復した経路へ復帰されないことになります。

システムブート時にない NIC

システムブート時にない NIC は、特別な障害検出です。起動時、起動スクリプトは、plumb できない `/etc/hostname.`*interface* ファイルを持つインタフェースを追跡します。このようなインタフェースの`/etc/hostname.`*interface* ファイル内のデータアドレスは、IPMP グループ内の代替インタフェースに自動的に配置されます。

このような場合は、次のようなエラーメッセージを受け取ります。

```
moving addresses from failed IPv4 interfaces: hme0 (moved to hme1)
moving addresses from failed IPv6 interfaces: hme0 (moved to hme1)
```

代替インタフェースが存在しない場合は、次のようなエラーメッセージを受け取ります。

```
moving addresses from failed IPv4 interfaces: hme0 (couldn't move;
   no alternative interface)
 moving addresses from failed IPv6 interfaces: hme0 (couldn't move;
   no alternative interface)
```

注 – このような障害検出では、不足インタフェースの `/etc/hostname.`*interface*ファイルで明示的に指定されているデータアドレスだけが、代替インタフェースに移されます。通常、RARP または DHCP などのほかの手段で取得されるアドレスは、取得または移動されません。

DR を使用して、システムブート時に不足していた別のインタフェースと同じ名前のインタフェースが再接続される場合、RCM は、インタフェースを自動的に plumb します。次に、RCM は、インタフェースの `/etc/hostname.`*interface*ファイルの内容に従って、インタフェースを構成します。最後に、インタフェースが回復したときのように、RCM はデータアドレスを回復した経路へ復帰させます。よって、最終的なネットワーク構成は、システムが現在のインタフェースでブートされた場合と同一の構成になります。

◆ ◆ ◆ 第 28 章

IPMP の管理 (タスク)

この章では、IP ネットワークマルチパス (IPMP) でインタフェースグループを管理するためのタスクを紹介します。この章では、主に次の内容について説明します。

- 737 ページの「IPMP の構成 (タスクマップ)」
- 739 ページの「高可用性のための IPMP グループの使用」
- 751 ページの「IPMP グループの維持」
- 755 ページの「動的再構成をサポートするシステムでの障害が発生した物理インタフェースの交換」
- 757 ページの「システムのブート時に存在しない物理インタフェースの回復」
- 759 ページの「IPMP 構成の変更」

IPMP の概念の概要については、第 27 章「IPMP の紹介 (概要)」を参照してください。

IPMP の構成 (タスクマップ)

このセクションには、この章で説明するタスクへのリンクが含まれています。

IPMP グループの構成と管理 (タスクマップ)

タスク	説明	参照先
IPMP グループの計画	IPMP グループを構成する以前の補助的な情報と必要なタスクをすべて示す	739 ページの「IPMP グループの計画を立てる方法」
複数のインタフェースを持つ IPMP インタフェースグループの構成	複数のインタフェースを IPMP グループのメンバーとして構成する	741 ページの「複数のインタフェースを持つ IPMP グループを構成する方法」

タスク	説明	参照先
インタフェースの 1 つが待機インタフェースである IPMP グループの構成	マルチインタフェース IPMP グループのインタフェースの 1 つを待機インタフェースとして構成する	748 ページの「IPMP グループの待機インタフェースを構成する方法」
1 つのインタフェースで構成される IPMP グループの構成	単一インタフェースの IPMP グループを作成する	750 ページの「単一インタフェースの IPMP グループを構成する方法」
物理インタフェースが属する IPMP グループの表示	`ifconfig` コマンドの出力からインタフェースの IPMP グループの名前を取得する方法を説明する	752 ページの「インタフェースの IPMP グループメンバーシップを表示する方法」
IPMP グループへのインタフェースの追加	既存の IPMP グループのメンバーとして新しいインタフェースを追加する	752 ページの「IPMP グループにインタフェースを追加する方法」
IPMP グループからのインタフェースの削除	IPMP グループからインタフェースを削除する方法を説明する	753 ページの「IPMP グループからインタフェースを削除する方法」
既存の IPMP グループから別のグループへのインタフェースの移動	IPMP グループ間でインタフェースを移動する	754 ページの「インタフェースを 1 つの IPMP グループから別のグループに移動する方法」
`in.mpathd` デーモンの 3 つのデフォルト設定の変更	`in.mpathd` デーモンの障害検出時間などのパラメータをカスタマイズする	759 ページの「`/etc/default/mpathd` ファイルを構成する方法」

動的再構成をサポートするインタフェースでの IPMP の管理 (タスクマップ)

タスク	説明	参照先
障害が発生したインタフェースの削除	システム内の障害が発生したインタフェースを削除する	755 ページの「障害が発生した物理インタフェースを削除する方法 (DR-Detach)」
障害が発生したインタフェースの交換	障害が発生したインタフェースを交換する	756 ページの「障害が発生した物理インタフェースを交換する方法 (DR-Attach)」
ブート時に構成されなかったインタフェースの回復	障害が発生したインタフェースを回復する	757 ページの「システムのブート時に存在しない物理インタフェースを回復する方法」

高可用性のための IPMP グループの使用

この節では、IPMP グループの構成手順を紹介します。また、インタフェースを待機インタフェースとして構成する方法についても説明します。

IPMP グループの計画

IPMP グループの一部としてシステム上のインタフェースを構成する前に、構成前の計画を立てる必要があります。

▼ IPMP グループの計画を立てる方法

次の手順には、計画タスクと IPMP グループを構成する前に収集すべき情報が含まれています。これらのタスクは、順番どおり行う必要はありません。

1. **システムのどのインタフェースを IPMP グループの一部とするかを決定します。**

   IPMP グループは、通常、同じ IP リンクに接続されている 2 つ以上の物理インタフェースによって構成されています。ただし、必要に応じて、単一インタフェースの IPMP グループを構成することも可能です。IPMP グループの紹介については、727 ページの「IPMP インタフェースの構成」を参照してください。たとえば、同じ Ethernet スイッチまたは同じ IP サブネットを同じ IPMP グループに構成できます。同じ IPMP グループにはいくつでもインタフェースを構成できます。

   論理インタフェースでは、`ifconfig` コマンドの `group` パラメータを使用することはできません。たとえば、`group` パラメータを `hme0` と使用することはできますが、`hme0:1` とは使用できません。

2. **グループ内の各インタフェースが一意の MAC アドレスを持っていることを確認します。**

   手順については、156 ページの「SPARC: インタフェースの MAC アドレスが一意であることを確認する方法」を参照してください。

3. **IPMP グループの名前を選択します。**

   空文字以外であれば、どんなグループ名でも構いません。インタフェースが接続されている IP リンクを識別する名前を使用するのが良いでしょう。

4 **STREAMS** モジュールの同じセットが転送され、**IPMP** グループ内のすべてのインタフェースで構成されていることを確認します。

同じグループのすべてのインタフェースは、同じ順番で構成された STREAMS モジュールを持っていなければなりません。

a. 予想される **IPMP** グループのすべてのインタフェースの **STREAMS** モジュールの順番を確認します。

`ifconfig` *interface* `modlist` コマンドを使用して、STREAMS モジュールの一覧を印刷できます。たとえば、`hme0` インタフェースの `ifconfig` 出力は次のようになります。

```
# ifconfig hme0 modlist
    0 arp
    1 ip
    2 hme
```

この出力の `ifconfig hme0 modlist` で示されているように、インタフェースは、通常 IP モジュールの直下にネットワークドライバとして存在します。追加構成は必要ありません。

ただし、NCA や IP フィルタのような一部のテクノロジーは、IP モジュールとネットワークドライバ間に自分自身を STREAMS モジュールとして挿入します。これにより、同じ IPMP グループのインタフェースの動作方法に問題が生じる場合があります。

STREAMS モジュールの処理状態を把握可能な場合には、グループ内のすべてのインタフェースに同じモジュールを転送している場合でも、フェイルオーバーで予想外の動作が実行される可能性はあります。ただし、IPMP グループのすべてのインタフェースに同じ順番で転送している場合は、処理状態を把握できない STREAMS モジュールを使用できます。

b. インタフェースのモジュールを **IPMP** グループでの標準的な順番で転送します。

`ifconfig` *interface* **`modinsert`** *module-name*

```
ifconfig hme0 modinsert ip
```

5 **IPMP** グループのすべてのインタフェースで同じ **IP** アドレス指定書式を使用します。

1 つのインタフェースが IPv4 向けに構成されている場合は、そのグループのすべてのインタフェースを IPv4 向けに構成する必要があります。たとえば、複数の NIC のインタフェースで構成される IPMP グループがあるとします。1 つの NIC のインタフェースに IPv6 アドレス指定を追加すると、IPMP グループ内のすべてのインタフェースを IPv6 によってサポートする構成にする必要があります。

6 **IPMP** グループ内のすべてのインタフェースが同じ **IP** リンクに接続されていることをチェックします。

7 **IPMP グループに、別のネットワークメディアタイプのインタフェースが含まれていないことを確認します。**

グループ化するインタフェースは、`/usr/include/net/if_types.h` で定義されている同じインタフェースタイプになるべきです。たとえば、1 つの IPMP グループに Ethernet インタフェースとトークンリングインタフェースを組み合わせることはできません。別の例としては、同じ IPMP グループに、トークンバスインタフェースと非同期転送モード (ATM) インタフェースを組み合わせることはできません。

8 **ATM インタフェースを持つ IPMP の場合は、LAN エミュレーションモードで ATM インタフェースを構成します。**

IPMP は、従来型の IP を ATM で使用するインタフェースではサポートされていません。

IPMP グループの構成

このセクションには、2 つ以上の物理インタフェースを持つ一般的な IPMP グループの構成タスクを記載します。

- マルチインタフェースの IPMP グループの紹介については、722 ページの「IPMP グループ」を参照してください。
- 計画を立てるタスクについては、739 ページの「IPMP グループの計画」を参照してください。
- 1 つの物理インタフェースだけで IPMP グループを構成するには、750 ページの「1 つの物理インタフェースを持つ IPMP グループの構成」を参照してください。

▼ 複数のインタフェースを持つ IPMP グループを構成する方法

次の IPMP グループの構成手順は、VLAN を IPMP グループに構成する際にも適用されます。

始める前に　IPv4 アドレス、および該当する場合は、予想される IPMP グループ内のすべてのインタフェースの IPv6 アドレスを構成しておく必要があります。

注意 – 各サブネットまたは L2 ブロードキャストドメインに対して、IPMP グループを 1 つだけ構成する必要があります。詳細については、723 ページの「IPMP の基本要件」を参照してください。

1 **構成するインタフェースのシステムで、Primary Administrator の役割を引き受けるか、スーパーユーザーになります。**

Primary Administrator 役割には、Primary Administrator プロファイルが含まれます。役割を作成してユーザーに役割を割り当てるには、『Oracle Solaris の管理: 基本管理』の第 2 章「Solaris 管理コンソールの操作 (タスク)」を参照してください。

2 各物理インタフェースを IPMP グループを配置します。

```
# ifconfig interface group group-name
```

たとえば、hme0 と hme1 をグループ testgroup1 の下に配置するには、次のコマンドを入力します。

```
# ifconfig hme0 group testgroup1
# ifconfig hme1 group testgroup1
```

グループ名には空白文字を使用しないでください。ifconfig ステータスディスプレイは、スペースを表示しません。したがって、違いは一方の名前がスペースを含んでいるだけという 2 つの類似するグループ名は作成しないでください。グループ名の 1 つにスペースが含まれる場合、これらのグループ名はステータスディスプレイでは同じに見えます。

デュアルスタック環境では、特定のグループにインタフェースの IPv4 インスタンスを配置すると、IPv6 インスタンスが自動的に同じグループに配置されます。

3 (任意) 1 つ以上の物理インタフェース上で IPv4 検査用アドレスを構成します。

検査用アドレスを構成する必要があるのは、特定のインタフェースで検査信号ベースの 障害検出を使用する場合だけです。検査用アドレスは、ifconfig コマンドに指定した物理インタフェースの論理インタフェースとして構成されます。

グループ内の 1 つのインタフェースを待機インタフェースにする場合は、この時点ではそのインタフェースの検査用アドレスを構成しないでください。待機インタフェースの検査用アドレスは、748 ページの「IPMP グループの待機インタフェースを構成する方法」のタスクの一環として構成します。

検査用アドレスを構成するには、次の構文の ifconfig コマンドを使用します。

```
# ifconfig interface addif ip-address parameters -failover deprecated up
```

たとえば、プライマリネットワークインタフェース hme0 には次の検査用アドレスを作成します。

```
# ifconfig hme0 addif 192.168.85.21 netmask + broadcast + -failover deprecated up
```

このコマンドは、プライマリネットワークインタフェース hme0 に対して次のパラメータを設定します。

- アドレスを 192.168.85.21 に設定する。
- ネットマスクおよびブロードキャストアドレスをデフォルト値に設定する。
- -failover と deprecated を指定

注 – この検査用アドレスをアプリケーションから使用されないようにするため IPv4 検査用アドレスを deprecated と指定する必要があります。

---

4 特定のインタフェースの **IPv4** 構成をチェックします。

`ifconfig` *interface* を入力することによって、インタフェースの現在のステータスを常に参照できます。インタフェースのステータス参照の詳細については、211 ページの「特定のインタフェースに関する情報を入手する方法」を参照してください。

検査用アドレスに割り当てられている論理インタフェースを指定することで、物理インタフェースの検査用アドレス構成に関する情報を取得できます。

```
# ifconfig hme0:1
    hme0:1: flags=9000843<UP,BROADCAST,RUNNING,MULTICAST,DEPRECATED,IPv4,NOFAILOVER>
    mtu 1500 index 2
    inet 192.168.85.21 netmask ffffff00 broadcast 192.168.85.255
```

5 (任意) 適用できる場合、**IPv6** 検査用アドレスを構成します。

```
# ifconfig interface inet6 -failover
```

IPv6 アドレスを持つ物理インタフェースは、インタフェースの IPv4 アドレスと同じ IPMP グループに配置されます。この状況は、IPv4 アドレスを持つ物理インタフェースを IPMP グループに構成する場合に発生します。まず IPv6 アドレスを持つ物理インタフェースを IPMP グループに配置する場合は、IPv4 アドレスを持つ物理インタフェースも暗示的に同じ IPMP グループに配置されます。

たとえば、IPv6 検査用アドレスを持つ `hme0` を構成するには、次のように入力します。

```
# ifconfig hme0 inet6 -failover
```

検査用アドレスをアプリケーションから使用されないようにするために IPv6 検査用アドレスを deprecated と指定する必要はありません。

6 **IPv6** 構成を確認します。

```
# ifconfig hme0 inet6
    hme0: flags=a000841<UP,RUNNING,MULTICAST,IPv6,NOFAILOVER> mtu 1500 index 2
            inet6 fe80::a00:20ff:feb9:17fa/10
            groupname test
```

IPv6 検査用アドレスは、インタフェースのリンクローカルアドレスです。

7 (任意) リブートしても **IPMP** グループ構成を保持させます。

- IPv4 の場合は、次の行を `/etc/hostname.`*interface* ファイルに追加します。

  ```
  interface-address <parameters> group group-name up \
      addif logical-interface -failover deprecated <parameters> up
  ```

  この例では、検査用 IPv4 アドレスは、次回のリブートで有効になります。構成をその場で有効にするには、手順 1、2、および任意で 3 を実行する必要があります。

- IPv6 の場合は、次の行を `/etc/hostname6.`*interface* ファイルに追加します。

  ```
  -failover group group-name up
  ```

この検査用 IPv6 アドレスは、次回のリブートで有効になります。構成をその場で有効にするには、手順 1、2、および任意で 5 を実行する必要があります。

8 **(任意) 手順 1 - 6 を繰り返して、IPMP グループにさらにインタフェースを追加します。**

実行中のシステムの既存のグループに新しいインタフェースを追加できます。ただし、リブート後は変更の内容は失われます。

例 28–1 2 つのインタフェースを持つ IPMP グループの構成

次の操作を実行したいとします。

- ネットマスクおよびブロードキャストアドレスをデフォルト値に設定する。
- 検査用アドレス `192.168.85.21` を持つインタフェースを構成する。

次のコマンドを入力します。

```
# ifconfig hme0 addif 192.168.85.21 netmask + broadcast + -failover deprecated up
```

この検査用アドレスをアプリケーションから使用されないようにするため IPv4 検査用アドレスを `deprecated` と指定する必要があります。741 ページの「複数のインタフェースを持つ IPMP グループを構成する方法」を参照してください。

アドレスのフェイルオーバー属性を有効にするには、ダッシュなしで `failover` オプションを使用します。

IPMP グループのすべての検査用 IP アドレスには、同じネットワークアドレスを使用してください。すべての検査用 IP アドレスは 1 つの IP サブネットに属していなければなりません。

例 28–2 リブート後の IPv4 IPMP グループ構成の保持

次の構成で `testgroup1` という名の IPMP グループを作成したいとします。

- データアドレス `192.168.85.19` を持つ物理インタフェース `hme0`
- 検査用アドレス `192.168.85.21` を持つ論理インタフェース

注 – この例では、物理インタフェースとデータアドレスが互いにペアになっています。論理インタフェースと検査用アドレスについても同様です。ただし、インタフェースのタイプとアドレスタイプの間に固有の関係が存在するわけではありません。

- `deprecated` と `-failover` オプション設定
- ネットマスクおよびブロードキャストアドレスをデフォルト値に設定する。

この場合、/etc/hostname.hme0 ファイルに次の行を追加します。

```
192.168.85.19 netmask + broadcast + group testgroup1 up \
    addif 192.168.85.21 deprecated -failover netmask + broadcast + up
```

同様に、2 番目のインタフェース hme1 を同じグループ (testgroup1) に入れ、検査用アドレスを指定するには、次の行を追加します。

```
192.168.85.20 netmask + broadcast + group testgroup1 up \
    addif 192.168.85.22 deprecated -failover netmask + broadcast + up
```

例 28–3 リブート後の IPv6 IPMP グループ構成の保持

IPv6 アドレスを持つインタフェース hme0 のテストグループを作成するには、次の行を /etc/hostname6.hme0 ファイルに追加します。

```
-failover group testgroup1 up
```

同様に、2 番目のインタフェース hme1 をグループ (testgroup1) に入れ、検査用アドレスを指定するには、次の行を /etc/hostname6.hme1 ファイルに追加します。

```
-failover group testgroup1 up
```

注意事項 IPMP グループの構成時、in.mpathd は、システムコンソールまたは syslog ファイルに多数のメッセージを出力します。これらのメッセージは、本質的に参考情報で、IPMP 構成が正しく機能していることを示します。

- このメッセージは、インタフェース hme0 が IPMP グループ testgroup1 に追加されたことを示します。ただし、hme0 では検査用アドレスは構成されていません。検査信号ベースの障害検出を有効にするには、検査用アドレスをインタフェースに割り当てる必要があります。

  ```
  May 24 14:09:57 host1 in.mpathd[101180]:
  No test address configured on interface hme0;
  disabling probe-based failure detection on it.
  testgroup1
  ```

- このメッセージは、IPMP グループに追加される IPv4 アドレスだけを持つすべてのインタフェースに対して表示されます。

  ```
  May 24 14:10:42 host4 in.mpathd[101180]:
  NIC qfe0 of group testgroup1 is not
  plumbed for IPv6 and may affect failover capability
  ```

- インタフェースの検査用アドレスを構成した場合には、このメッセージが表示されるはずです。

  ```
  Created new logical interface hme0:1
  May 24 14:16:53 host1 in.mpathd[101180]:
  Test address now configured on interface hme0;
   enabling probe-based failure detection on it
  ```

参照 IPMP グループをアクティブ-待機構成にする場合は、748 ページの「IPMP グループの待機インタフェースを構成する方法」を参照してください。

ターゲットシステムの構成

検査信号ベースの障害検出では、730 ページの「検査信号ベースの障害検出」で説明されているようにターゲットシステムを使用します。一部の IPMP グループでは、`in.mpathd` が使用するデフォルトのターゲットで十分です。ただし、一部の IPMP グループでは、検査信号ベースの障害検出用に特定のターゲットを構成したほうが良いこともあります。ルートテーブルのホストルートを検査信号のターゲットとして設定して、検査信号ベースの障害検出を実行します。経路制御テーブルに構成されているすべてのホストルートは、デフォルトルーターの前に一覧化されます。したがって、IPMP はターゲットを選択するために、明示的に定義されたホストルートを使用します。直接ターゲットを指定するには、 ホストルートを手動で設定するか、起動スクリプトになることができるシェルスクリプトを作成します。

ネットワーク上のどのホストが適切なターゲットになるのかの評価では、 次の基準を検討します。

- 予想されるターゲットが使用可能で、実行されていることを確認します。IP アドレスの一覧を作成します。
- ターゲットインタフェースが、構成中の IPMP グループと同じネットワークにあることを確認します。
- ターゲットシステムのネットマスクとブロードキャストアドレスは、IPMP グループ内のアドレスと同じでなければなりません。
- ターゲットホストは、検査信号ベースの障害検出を使用しているインタフェースからの ICMP 要求に応答できなければなりません。

▼ 検査信号ベースの障害検出のターゲットシステムを手動で指定する方法

1 検査信号ベースの障害検出を構成しているシステムにユーザーアカウントでログインします。

2 検査信号ベースの障害検出のターゲットとして使用される特定のホストにルートを追加します。

```
$ route add -host destination-IP gateway-IP -static
```

destination-IP と *gateway-IP* の値を、ターゲットとして使用されるホストの IPv4 アドレスと置き換えます。たとえば、IPMP グループ `testgroup1` のインタフェースと同じサブネット上のターゲットシステム `192.168.85.137` を指定するには、次のように入力します。

```
$ route add -host 192.168.85.137 192.168.85.137 -static
```

3 ターゲットシステムとして使用されるネットワーク上の追加ホストにルートを追加します。

▼ シェルスクリプトでターゲットシステムを指定する方法

1 **IPMP** グループを構成したシステムで、**Primary Administrator** の役割を引き受けるか、スーパーユーザーになります。

Primary Administrator 役割には、Primary Administrator プロファイルが含まれます。役割を作成してユーザーに役割を割り当てるには、『Oracle Solaris の管理: 基本管理』の第 2 章「Solaris 管理コンソールの操作 (タスク)」を参照してください。

2 提案したターゲットへの静的なルートを設定するシェルスクリプトを作成します。

たとえば、次のような内容の `ipmp.targets` というシェルスクリプトを作成します。

```
TARGETS="192.168.85.117 192.168.85.127 192.168.85.137"

case "$1" in
        'start')
            /usr/bin/echo "Adding static routes for use as IPMP targets"
        for target in $TARGETS; do
      /usr/sbin/route add -host $target $target
        done
                ;;
        'stop')
              /usr/bin/echo "Removing static routes for use as IPMP targets"
         for target in $TARGETS; do
        /usr/sbin/route delete -host $target $target
         done
                ;;
  esac
```

3 シェルスクリプトを起動スクリプトディレクトリにコピーします。

```
# cp ipmp.targets /etc/init.d
```

4 新しい起動スクリプトのアクセス権を変更します。

```
# chmod 744 /etc/init.d/ipmp.targets
```

5 新しい起動スクリプトの所有権を変更します。

```
# chown root:sys /etc/init.d/ipmp.targets
```

6 **/etc/init.d** ディレクトリ内に起動スクリプトのリンクを作成します。

```
# ln /etc/init.d/ipmp.targets /etc/rc2.d/S70ipmp.targets
```

ファイル名 `S70ipmp.targets` の接頭辞 S70 によって、ほかの起動スクリプトを尊重しながら新しいスクリプトが命令されます。

待機インタフェースの構成

IPMP グループをアクティブ-待機構成にする場合は、この手順を行ってください。この種類の構成についての詳細は、727 ページの「IPMP インタフェースの構成」を参照してください。

▼ IPMP グループの待機インタフェースを構成する方法

始める前に
- すべてのインタフェースを IPMP グループのメンバーとして構成しておく必要があります。
- 待機インタフェースとなるインタフェースには、検査用アドレスを構成しないでください。

IPMP グループの構成と検査用アドレスの割り当てについては、741 ページの「複数のインタフェースを持つ IPMP グループを構成する方法」を参照してください。

1 構成する待機インタフェースのシステムで、**Primary Administrator** の役割を引き受けるか、スーパーユーザーになります。

Primary Administrator 役割には、Primary Administrator プロファイルが含まれます。役割を作成してユーザーに役割を割り当てるには、『Oracle Solaris の管理: 基本管理』の第 2 章「Solaris 管理コンソールの操作 (タスク)」を参照してください。

2 インタフェースを待機用として構成し、検査用アドレスを割り当てます。

```
# ifconfig interface plumb \
ip-address other-parameters deprecated -failover standby up
```

待機インタフェースは、1 つの IP アドレス (検査用アドレス) しか持つことができません。`standby up` オプションを設定する前に、`-failover` オプションを設定してください。`<other-parameters>` の場合は、`ifconfig(1M)` のマニュアルページに説明されているように、構成で必要なパラメータを使用します。

- たとえば、IPv4 検査用アドレスを作成するには、次のコマンドを入力します。

```
# ifconfig hme1 plumb 192.168.85.22 netmask + broadcast + deprecated -failover standby up
```

`hme1`	待機インタフェースとして構成する物理インタフェースとして `hme1` を定義します。
`192.168.85.22`	この検査用アドレスを待機インタフェースに割り当てます。
`deprecated`	その検査用アドレスが出力パケットには使用されないことを示します。
`-failover`	インタフェースで障害が発生しても、検査用アドレスのフェイルオーバーは行われないことを示します。
`standby`	インタフェースを待機インタフェースに指定します。

- たとえば、IPv6 検査用アドレスを作成するには、次のコマンドを入力します。

```
# ifconfig hme1 plumb -failover standby up
```

3 待機インタフェースの構成結果をチェックします。

```
# ifconfig hme1
hme1: flags=69040843<UP,BROADCAST,RUNNING,MULTICAST,DEPRECATED,IPv4,NOFAILOVER,
      STANDBY,INACTIVE mtu 1500
         index 4 inet 192.168.85.22 netmask ffffff00 broadcast 19.16.85.255
         groupname test
```

INACTIVE は、このインタフェースが送信パケットには使用されないことを示します。この待機インタフェースに障害経路の迂回が行われると、INACTIVE 状態は取り消されます。

注 – ifconfig *interface* コマンドを入力することによって、インタフェースの現在のステータスを常に参照できます。インタフェースのステータス参照については、211 ページの「特定のインタフェースに関する情報を入手する方法」を参照してください。

---

4 (オプション) リブート後も **IPv4** 待機インタフェースを保持します。

待機インタフェースを同じ IPMP グループに割り当て、待機インタフェースの検査用アドレスを構成します。

たとえば、hme1 を待機インタフェースとして構成するには、/etc/hostname.hme1 ファイルに次の行を追加します。

```
192.168.85.22 netmask + broadcast + deprecated group test -failover standby up
```

5 (任意) リブート後も **IPv6** 待機インタフェースを保持します。

待機インタフェースを同じ IPMP グループに割り当て、待機インタフェースの検査用アドレスを構成します。

たとえば、hme1 を待機インタフェースとして構成するには、/etc/hostname6.hme1 ファイルに次の行を追加します。

```
-failover group test standby up
```

例 28–4 IPMP グループの待機インタフェースの構成

次の構成に基づいて検査用アドレスを作成するとします。

- 物理インタフェース hme2 を待機インタフェースにする。
- 検査用アドレスは 192.168.85.22 とする。
- deprecated と -failover オプション設定
- ネットマスクおよびブロードキャストアドレスをデフォルト値に設定する。

その場合、次のように入力します。

```
# ifconfig hme2 plumb 192.168.85.22 netmask + broadcast + \
deprecated -failover standby up
```

インタフェースは、アドレスが NOFAILOVER として設定されたあとにだけ、待機インタフェースとして設定されます。

次のように入力して、インタフェースの待機ステータスを解除します。

```
# ifconfig interface -standby
```

1 つの物理インタフェースを持つ IPMP グループの構成

IPMP グループにインタフェースが 1 つしかない場合は、フェイルオーバーを実行できません。ただし、インタフェースを IPMP グループに割り当てることで、インタフェースの障害検出を有効にすることはできます。単一インタフェースの IPMP グループの障害検出を確立するために、専用テスト IP アドレスを構成する必要はありません。単一の IP アドレスを、データの送信と障害検出の両方に使用できます。

▼ 単一インタフェースの IPMP グループを構成する方法

1. **予想される単一インタフェース IPMP グループのあるシステムで、Primary Administrator の役割を引き受けるか、スーパーユーザーになります。**

   Primary Administrator 役割には、Primary Administrator プロファイルが含まれます。役割を作成してユーザーに役割を割り当てるには、『Oracle Solaris の管理: 基本管理』の第 2 章「Solaris 管理コンソールの操作 (タスク)」を参照してください。

2. **IPv4 の場合は、単一インタフェースの IPMP グループを作成します。**

   次の構文で単一インタフェースを IPMP グループに割り当てます。

   ```
   # ifconfig interface group group-name
   ```

   次の例では、インタフェース hme0 が IPMP グループ v4test に割り当てられます。

   ```
   # ifconfig hme0 group v4test
   ```

   この手順の実行後、IPMP によりリンクベースの障害検出がインタフェース上で有効になります。

   また、ifconfig コマンドの -failover サブコマンドを使用して、検査信号ベースの障害検出を有効にすることもできます。次の例では、現在 hme0 に割り当てられている IP アドレスを使用して、hme0 で検査信号ベースの障害検出を有効にします。

   ```
   # ifconfig hme0 -failover
   ```

複数インタフェースのグループとは異なり、同じ IP アドレスをデータアドレスと検査用 IP アドレスの両方に使用できます。アプリケーションで検査用アドレスをデータアドレスとして使用できるようにするには、単一インタフェースの IPMP グループで検査用アドレスに `deprecated` のマークを付けないでください。

3 **IPv6 の場合は、単一インタフェースの IPMP グループを作成します。**

次の構文で単一インタフェースを IPMP グループに割り当てます。

`# ifconfig` *interface* **`inet6 group`** *group-name*

たとえば、単一インタフェース `hme0` を IPMP グループ `v6test` に追加するには、次のように入力します。

```
# ifconfig hme0 inet6 group v6test
```

この手順の実行後、IPMP によりリンクベースの障害検出がインタフェース上で有効になります。

また、`ifconfig` コマンドの `-failover` サブコマンドを使用して、検査信号ベースの障害検出を有効にすることもできます。次の例では、現在 `hme0` に割り当てられている IP アドレスを使用して、`hme0` で検査信号ベースの障害検出を有効にします。

```
# ifconfig hme0 inet6 -failover
```

複数インタフェースのグループとは異なり、同じ IP アドレスをデータアドレスと検査用 IP アドレスの両方に使用できます。アプリケーションで検査用アドレスをデータアドレスとして使用できるようにするには、単一インタフェースの IPMP グループで検査用アドレスに `deprecated` のマークを付けないでください。

単一物理インタフェース構成では、検査信号が送信されるターゲットシステムで障害が発生しているのか、インタフェースで障害が発生しているのかを確認できません。検査信号の受信システムは、単一の物理インタフェースを介して検査されます。唯一のデフォルトルーターがサブネット上にある場合、単一物理インタフェースがグループ内にあるときは、IPMP をオフにします。IPv4 と IPv6 のデフォルトルーターが別個に存在する場合 (または複数のデフォルトルーターが存在する場合) は、検査信号のターゲットシステムは 2 つ以上あります。よって、IPMP を安全にオンに設定できます。

IPMP グループの維持

このセクションには、既存の IPMP グループとこれらのグループを構成するインタフェースを維持するタスクを記載します。739 ページの「高可用性のための IPMP グループの使用」の説明に従って、すでに IPMP グループを構成していることが前提です。

▼ インタフェースの IPMP グループメンバーシップを表示する方法

1 **IPMP** グループ構成を持つシステムで、スーパーユーザーまたは同等の役割になります。

役割には、認証と特権コマンドが含まれます。役割の詳細については、『Solaris のシステム管理: セキュリティーサービス』の「RBAC の構成 (タスクマップ)」を参照してください。

2 インタフェースが属するグループなど、インタフェースに関する情報を表示します。

```
# ifconfig interface
```

3 適切な場合は、インタフェースの **IPv6** 情報を表示します。

```
# ifconfig interface inet6
```

例 28–5 物理インタフェースグループの表示

`hme0` のグループ名を表示するには、次のように入力します。

```
# ifconfig hme0
    hme0: flags=9000843<UP,BROADCAST,RUNNING,MULTICAST,IPv4> mtu 1500
      index 2 inet 192.168.85.19 netmask ffffff00 broadcast 192.168.85.255
      groupname testgroup1
```

IPv6 情報だけのグループ名を表示するには、次のように入力します。

```
# ifconfig hme0 inet6
    hme0: flags=a000841<UP,RUNNING,MULTICAST,IPv6> mtu 1500 index 2
            inet6 fe80::a00:20ff:feb9:19fa/10
            groupname testgroup1
```

▼ IPMP グループにインタフェースを追加する方法

1 **IPMP** グループ構成を持つシステムで、**Primary Administrator** の役割を引き受けるか、スーパーユーザーになります。

Primary Administrator 役割には、Primary Administrator プロファイルが含まれます。役割を作成してユーザーに役割を割り当てるには、『Oracle Solaris の管理: 基本管理』の第 2 章「Solaris 管理コンソールの操作 (タスク)」を参照してください。

2 **IPMP** グループへインタフェースを追加します。

```
# ifconfig interface group group-name
```

interface で指定したインタフェースが、IPMP グループ *group-name* のメンバーになります。

例 28–6 IPMP グループへのインタフェースの追加

hme0 を IPMP グループ testgroup2 に追加するには、次のコマンドを入力します。

```
# ifconfig hme0 group testgroup2
  hme0: flags=9000843<UP ,BROADCAST,RUNNING,MULTICAST,IPv4,NOFAILOVER> mtu 1500 index 2
  inet 192.168.85.19 netmask ff000000 broadcast 10.255.255.255
  groupname testgroup2
  ether 8:0:20:c1:8b:c3
```

▼ IPMP グループからインタフェースを削除する方法

ifconfig コマンドの group パラメータに空文字列を指定すると、インタフェースが現在の IPMP グループから削除されます。グループからインタフェースを削除する場合は、慎重に行う必要があります。IPMP グループのほかのインタフェースに障害が発生している場合、障害経路の迂回が行われていることがあります。たとえば、hme0 に障害が発生し、すべてのアドレスが、同じグループに属する hme1 に移されたとします。このグループから hme1 を削除すると in.mpathd デーモンはこれらの障害経路の迂回が行われたアドレスをグループ内のほかのインタフェースに戻します。正常に動作しているインタフェースがグループ内になければ障害経路の迂回が行われず、すべてのネットワークアクセスは維持できません。

同様に、グループ内のインタフェースを unplumb する必要がある場合は、まずインタフェースをグループから削除する必要があります。そのあと、構成されたすべての IP アドレスを確実に維持します。これは、グループから削除されるインタフェースの構成を in.mpathd デーモンが再現しようとするからです。インタフェースを unplumb する場合は、その前に構成が再現されていなければなりません。フェイルオーバーの前後のインタフェースの見た目については、732 ページの「インタフェースのフェイルオーバー時の処理」を参照してください。

1. **IPMP グループ構成を持つシステムで、Primary Administrator の役割を引き受けるか、スーパーユーザーになります。**

   Primary Administrator 役割には、Primary Administrator プロファイルが含まれます。役割を作成してユーザーに役割を割り当てるには、『Oracle Solaris の管理: 基本管理』の第 2 章「Solaris 管理コンソールの操作 (タスク)」を参照してください。

2. **IPMP グループからインタフェースを削除します。**

   ```
   # ifconfig interface group ""
   ```

引用符 ("") は空文字列を表します。

例 28–7 グループからのインタフェースの削除

hme0 を IPMP グループ test から削除するには、次のコマンドを入力します。

```
# ifconfig hme0 group ""
    # ifconfig hme0
    hme0: flags=9000843<UP,BROADCAST,RUNNING,MULTICAST,IPv4> mtu 1500
    index 2 inet 192.168.85.19 netmask ffffff00 broadcast 192.168.85.255
    # ifconfig hme0 inet6
    hme0: flags=a000841<UP,RUNNING,MULTICAST,IPv6> mtu 1500 index 2
    inet6 fe80::a00:20ff:feb9:19fa/10
```

▼ インタフェースを 1 つの IPMP グループから別のグループに移動する方法

インタフェースが既存の IPMP グループに属している場合は、新しい IPMP グループにインタフェースを配置できます。この場合、現在の IPMP グループからインタフェースを削除する必要はありません。新しいグループに追加されたインタフェースは、既存の IPMP グループから自動的に削除されます。

1. **IPMP グループ構成を持つシステムで、Primary Administrator の役割を引き受けるか、スーパーユーザーになります。**

   Primary Administrator 役割には、Primary Administrator プロファイルが含まれます。役割を作成してユーザーに役割を割り当てるには、『Oracle Solaris の管理: 基本管理』の第 2 章「Solaris 管理コンソールの操作 (タスク)」を参照してください。

2. **新しい IPMP グループへインタフェースを移動します。**

   `# ifconfig` *interface* `group` *group-name*

   インタフェースを新しいグループに追加すると、そのインタフェースは現在のグループから自動的に削除されます。

例 28–8 別の IPMP グループへのインタフェースの移動

インタフェース hme0 の IPMP グループを変更するには、次のように入力します。

```
# ifconfig hme0 group cs-link
```

このコマンドは、hme0 インタフェースを IPMP グループ test から削除し、グループ cs-link に追加します。

動的再構成をサポートするシステムでの障害が発生した物理インタフェースの交換

この節には、動的再構成 (DR) をサポートするシステムを管理する手順を記載します。

注 – このタスクは、`ifconfig` コマンドを使用して構成される IP 層にしか関係ありません。ATM またはほかのサービスなど、IP 層よりも上位または下位の層が自動化されていない場合には、手動による特別な手順が必要です。次の手順は、切断の前処理でインタフェースの構成を解除し、接続の後処理でインタフェースを構成するために使用します。

▼ 障害が発生した物理インタフェースを削除する方法 (DR-Detach)

この手順では、DR をサポートするシステムの物理インタフェースを削除する方法を示します。ここでは、すでに次の状態が存在していることを想定しています。

- サンプルインタフェースは、物理インタフェース `hme0` と `hme1` です。
- 両方のインタフェースが同じ IPMP グループ内にあります。
- `hme0` で障害が発生しています。
- 論理インタフェース `hme0:1` は、検査用アドレスを持っています。
- 障害のあるインタフェースを同じ名前の別の物理インタフェースで置き換えます。`hme0` の置き換えを例にとります。

注 – 検査用アドレスが `/etc/hostname.hme0` ファイルを使用して plumb されている場合は、手順 2 は省略できます。

---

1 **IPMP グループ構成を持つシステムで、Primary Administrator の役割を引き受けるか、スーパーユーザーになります。**

Primary Administrator 役割には、Primary Administrator プロファイルが含まれます。役割を作成してユーザーに役割を割り当てるには、『Oracle Solaris の管理: 基本管理』の第 2 章「Solaris 管理コンソールの操作 (タスク)」を参照してください。

2 検査用アドレスの構成を表示します。

```
# ifconfig hme0:1

hme0:1:
flags=9040842<BROADCAST,RUNNING,MULTICAST,DEPRECATED,IPv4,NOFAILOVER>
```

```
mtu 1500 index 3
inet 192.168.233.250 netmask ffffff00 broadcast 192.168.233.255
```

この情報は、物理インタフェースを交換する時に、検査用アドレスを再度 plumb するために必要です。

3 物理インタフェースを削除します。

物理インタフェースの削除に関する完全な説明については、次の参考文書を参照してください。

- cfgadm(1M) のマニュアルページ
- 『Sun Enterprise 6x00, 5x00, 4x00, and 3x00 Systems Dynamic Reconfiguration User's Guide』
- 『Sun Enterprise 10000 DR Configuration Guide』

▼ 障害が発生した物理インタフェースを交換する方法 (DR-Attach)

この手順では、DR をサポートするシステムの物理インタフェースを交換する方法を示します。

1 **IPMP** グループ構成を持つシステムで、**Primary Administrator** の役割を引き受けるか、スーパーユーザーになります。

Primary Administrator 役割には、Primary Administrator プロファイルが含まれます。役割を作成してユーザーに役割を割り当てるには、『Oracle Solaris の管理: 基本管理』の第 2 章「Solaris 管理コンソールの操作 (タスク)」を参照してください。

2 物理インタフェースを交換します。

手順については、次の参考文書を参照してください。

- cfgadm(1M) のマニュアルページ
- 『Sun Enterprise 6x00, 5x00, 4x00, and 3x00 Systems Dynamic Reconfiguration User's Guide』
- 『Sun Enterprise 10000 DR Configuration Guide』または『Sun Fire 880 Dynamic Reconfiguration User's Guide』

システムのブート時に存在しない物理インタフェースの回復

注 – この手順は、`ifconfig` コマンドを使用して構成される IP 層にしか関係ありません。ATM またはほかのサービスなど、IP 層よりも上位または下位の層が自動化されていない場合には、手動による特別な手順が必要です。次の手順は、切断の前処理でインタフェースの構成を解除し、接続の後処理でインタフェースを構成するために使用します。

動的再構成後の回復は、Sun Fire™ プラットフォームの I/O ボードの一部であるインタフェースでは自動的に行われます。NIC が Sun Crypto Accelerator I - cPCI board の場合も、回復は自動的に行われます。よって、インタフェースが DR 操作の一部として戻される場合には、次の手順を行う必要はありません。Sun Fire x800 および Sun Fire 15000 システムの詳細については、`cfgadm_sbd(1M)` のマニュアルページを参照してください。物理インタフェースは、`/etc/hostname.`*interface* ファイルで指定された構成にフェイルオーバーされます。リブートしても構成を保持できるようにインタフェースを構成する方法についての詳細は、739 ページの「高可用性のための IPMP グループの使用」を参照してください。

注 – 以前の Sun Fire システム (Exx00) の場合には、DR 切り離しは手動で行う必要があります。ただし、DR 接続は自動的に行われます。

▼ システムのブート時に存在しない物理インタフェースを回復する方法

システムのブート時に存在しない物理インタフェースを回復するには、次の手順を行なってください。この手順の例は、次のような構成です。

- 物理インタフェース `hme0` と `hme1` は、インタフェースです。
- 両方のインタフェースが同じ IPMP グループ内にあります。
- `hme0` は、システムブート時にインストールされていません。

注 – 障害が発生した物理インタフェースの回復時における、IP アドレスの障害回路の迂回には、3 分かかります。この時間は、ネットワークトラフィックによって異なります。また、所要時間は `in.mpathd` によって障害経路を迂回し回復した着信インタフェースの安定性によっても異なります。

---

1 **IPMP** グループ構成を持つシステムで、**Primary Administrator** の役割を引き受けるか、スーパーユーザーになります。

Primary Administrator 役割には、Primary Administrator プロファイルが含まれます。役割を作成してユーザーに役割を割り当てるには、『Oracle Solaris の管理: 基本管理』の第 2 章「Solaris 管理コンソールの操作 (タスク)」を参照してください。

2 コンソールログの障害エラーメッセージから、障害が発生したネットワークの情報を取得します。

syslog(3C) のマニュアルページを参照してください。エラーメッセージは次のように表示されます。

```
moving addresses from failed IPv4 interfaces:
hme1 (moved to hme0)
```

このメッセージは、障害が発生したインタフェース hme1 の IPv4 アドレスの障害が hme0 インタフェースに迂回されたことを示しています。

また、次のようなメッセージを受け取ることもあります。

```
moving addresses from failed IPv4 interfaces:
hme1 (couldn't move, no alternative interface)
```

このメッセージは、 障害が発生したインタフェース hme1 と同じグループにアクティブなインタフェースを発見できなかったことを示しています。したがって、hme1 の IPv4 アドレスの障害を迂回することはできません。

3 システムに物理インタフェースを接続します。

物理インタフェースの交換方法については、次の参考文書を参照してください。

- cfgadm(1M) のマニュアルページ
- 『Sun Enterprise 10000 DR Configuration Guide』
- 『Sun Enterprise 6x00, 5x00, 4x00, and 3x00 Systems Dynamic Reconfiguration User's Guide』

4 手順 **2** のエラーメッセージの内容を参照し、アドレスを移動できなかった場合は手順**6** へ、アドレスが移動された場合は手順**5** へ進んでください。

5 フェイルオーバープロセスの一環として構成された論理インタフェースを **unplumb** します。

a. **/etc/hostname.** *moved-from-interface* ファイルの内容から、フェイルオーバー処理の一部として構成された論理インタフェースを確認してください。

b. 各フェイルオーバー **IP** アドレスを **unplumb** します。

```
# ifconfig moved-to-interface removeif moved-ip-address
```

注 – フェイルオーバーアドレスは、failover パラメータが指定されたアドレス、または -failover パラメータが指定されていないアドレスです。-failover が指定された IP アドレスは、unplumb する必要がありません。

たとえば、/etc/hostname.hme0 ファイルの中に次の行が含まれている場合

```
inet 10.0.0.4 -failover up group one
addif 10.0.0.5 failover up
addif 10.0.0.6 failover up
```

各フェイルオーバー IP アドレスを unplumb するためには、次のコマンドを入力します。

```
# ifconfig hme0 removeif 10.0.0.5
# ifconfig hme0 removeif 10.0.0.6
```

6 問題となっている各インタフェース用に次のコマンドを入力して、交換した物理インタフェースの **IPv4** 情報を再構成します。

```
# ifconfig removed-from-NIC <parameters>
```

たとえば、次のコマンドを入力します。

```
# ifconfig hme1 inet plumb
# ifconfig hme1 inet 10.0.0.4 -failover up group one
# ifconfig hme1 addif 10.0.0.5 failover up
# ifconfig hme1 addif 10.0.0.6 failover up
```

IPMP 構成の変更

IPMP グループに関連する次のシステム共通パラメータを設定するには、IPMP 構成ファイル /etc/default/mpathd を使用します。

- FAILURE_DETECTION_TIME
- TRACK_INTERFACES_ONLY_WITH_GROUPS
- FAILBACK

▼ /etc/default/mpathd ファイルを構成する方法

1 **IPMP** グループ構成を持つシステムで、**Primary Administrator** の役割を引き受けるか、スーパーユーザーになります。

Primary Administrator 役割には、Primary Administrator プロファイルが含まれます。役割を作成してユーザーに役割を割り当てるには、『Oracle Solaris の管理: 基本管理』の第 2 章「Solaris 管理コンソールの操作 (タスク)」を参照してください。

2 **/etc/default/mpathd** ファイルを編集します。

3 つのパラメータの 1 つ以上のデフォルト値を変更します。

a. **FAILURE_DETECTION_TIME** パラメータの新しい値を入力します。

```
FAILURE_DETECTION_TIME=n
```

ここで、*n* は ICMP 検証がインタフェースの障害が発生していないかどうかを検出する時間 (秒単位) です。デフォルトは 10 秒です。

b. **FAILBACK** パラメータの新しい値を入力します。

```
FAILBACK=[yes | no]
```

- *yes* - 値 *yes* を指定した場合は、IPMP のデフォルトのフェイルバック動作になります。障害が発生したインタフェースの回復が検出されると、ネットワークアクセスはこの回復したインタフェースに復帰されます。詳細は、729 ページの「IPMP 障害検出とリカバリ機能」を参照してください。
- *no* - *no* を指定した場合、回復したインタフェースにデータトラフィックが戻ることはありません。障害が発生したインタフェースの回復が検出されると、そのインタフェースに *INACTIVE* フラグが設定されます。このフラグは、現時点でそのインタフェースをデータトラフィックに使用すべきでないことを示します。ただし、そのインタフェースを検査信号トラフィックに使用することはできます。

  たとえば、IPMP グループが 2 つのインタフェース ce0 と ce1 で構成されているとします。また、*/etc/default/mpathd* に値 FAILBACK=no が設定されているとします。ce0 で障害が発生した場合、IPMP の期待される動作に従って、そのトラフィックの処理は ce1 に継続されます。ただし、/etc/default/mpathd に FAILBACK=no パラメータが設定されているため、IPMP が*ce0* の回復を検出してもトラフィックが ce1 から復帰することはありません。ce1 インタフェースに障害が発生しないかぎり、ce0 インタフェースはその INACTIVE ステータスを維持し、トラフィックには使用されません。ce1 インタフェースに障害が発生すると、ce1 上のアドレスは ce0 に戻され、ce0 の INACTIVE フラグは消去されます。この移行が発生するのは、ce0 がグループ内で唯一の INACTIVE インタフェースである場合です。グループ内にほかの INACTIVE インタフェースが存在している場合は、ce0 以外の INACTIVE インタフェースにアドレスが移行されることもあります。

c. **TRACK_INTERFACES_ONLY_WITH_GROUPS** パラメータの新しい値を入力します。

```
TRACK_INTERFACES_ONLY_WITH_GROUPS=[yes | no]
```

- *yes* - 値 *yes* を指定した場合は、IPMP のデフォルトの動作になります。このパラメータを指定した場合、IPMP は、IPMP グループ内に構成されていないネットワークインタフェースを無視します。

- *no* - 値 *no* を指定すると、IPMP グループ内に構成されているネットワークインタフェースかどうかにかかわらず、すべてのネットワークインタフェースの障害と回復を検出するように設定されます。ただし、IPMP グループ内に構成されていないインタフェースで障害や回復が検出されたときは、フェイルオーバーやフェイルバックは発生しません。したがって、値 *no* を指定することは、障害の報告には役立ちますが、ネットワークの可用性を直接向上させることはありません。

3 **in.mpathd** デーモンを再起動します。

```
# pkill -HUP in.mpathd
```

パート VI

IP サービス品質 (IPQoS)

このパートには、Oracle Solaris での差別化サービス実装である IP サービス品質 (IPQoS) に関するタスクと関連情報を記載します。

◆ ◆ ◆ 第 29 章

IPQoS の紹介 (概要)

IP サービス品質 (IPQoS) を使用すると、優先順位付け、管理、およびアカウンティング統計情報の収集を行うことができます。IPQoS によって、ネットワークのユーザーに一貫したサービスレベルを提供できます。また、ネットワークの輻輳を防ぐために、トラフィックを管理することもできます。

この章では、次の内容について説明します。

IPQoS の基本

IPQoS は、Internet Engineering Task Force (IETF) の Differentiated Services Working Group によって定義されている差別化サービス (Diffserv) アーキテクチャーに対応しています。Oracle Solaris では、TCP/IP プロトコルスタックの IP レベルで IPQoS が実装されます。

差別化サービスとは

IPQoS を有効にすると、選択した顧客や選択したアプリケーションにさまざまなレベルのネットワークサービスを提供できます。この異なるレベルのサービスは、まとめて「差別化サービス」と呼ばれます。顧客に提供する差別化サービスは、ユーザーの企業が顧客に提供するサービスレベルの構造を基に決定できます。ネットワーク上のアプリケーションやユーザーに設定した優先順位に基づく場合もあります。

サービス品質を提供するためには、次のタスクを行います。

- 顧客や企業内の部署などのグループごとに異なるサービスレベルを提供する
- 特定のグループやアプリケーションにネットワークサービスを優先的に提供する
- ネットワーク上の障害やその他の輻輳の発生箇所を特定し、問題を取り除く
- ネットワークパフォーマンスを監視し、パフォーマンスの統計情報を提供する
- ネットワークリソースに対する帯域幅を調整する

IPQoS の機能

IPQoS には次の機能があります。

- `ipqosconf` QoS ポリシーを構成するためのコマンド行ツール
- アクションを選択するクラシファイア。アクションは、組織の QoS ポリシーを構成するフィルタに基づいている
- Diffserv モデルに従ってネットワークトラフィックを測定するメータリングモジュール
- パケットの IP ヘッダーに転送情報を付ける機能に基づく、サービスの差別化
- トラフィックフローの統計情報を収集するフローアカウンティングモジュール
- UNIX® の `kstat` コマンドによる、トラフィッククラスごとの統計情報の収集
- SPARC® および x86 アーキテクチャーのサポート
- IPv4 および IPv6 アドレス指定のサポート
- IP セキュリティーアーキテクチャーの相互運用性 (IPsec)
- 仮想ローカルエリアネットワーク (VLAN) の 802.1D ユーザー優先順位マークのサポート

サービス品質の理論と実践に関する情報をもっと得るには

差別化サービスやサービス品質の詳細情報は、印刷物やオンラインで入手できます。

サービス品質に関する書籍

サービス品質の理論や実践については、次の関連書籍を参照してください。

- Ferguson, Paul および Geoff Huston 著『*Quality of Service*』。John Wiley & Sons, Inc. 発行、1998 年
- Kilkki, Kalevi 著『*Differentiated Services for the Internet* 』Macmillan Technical Publishing 発行、1999 年

サービス品質に関する RFC (Request for Comments)

IPQoS は、次の RFC および Internet Draft の仕様に準拠しています。

- RFC 2474, Definition of the Differentiated Services Field (DS Field) in the IPv4 and IPv6 Headers (`http://www.ietf.org/rfc/rfc2474.txt?number=2474`) – 差別化サービスをサポートするための、IPv4 や IPv6 パケットヘッダーのサービスタイプ (ToS) フィールドまたは DS フィールドの拡張を説明している
- RFC 2475, An Architecture for Differentiated Services (`http://www.ietf.org/rfc/rfc2475.txt?number=2475`) – Diffserv アーキテクチャーの構成とモジュールについて詳細に解説している
- RFC 2597, Assured Forwarding PHB Group (`http://www.ietf.org/rfc/rfc2597.txt?number=2597`) – 相対的優先転送 (AF) ホップ単位動作について解説している
- RFC 2598, An Expedited Forwarding PHB (`http://www.ietf.org/rfc/rfc2598.txt?number=2598`) – 完全優先転送 (EF) ホップ単位動作について解説している
- インターネットドラフト「*An Informal Management Model for Diffserv Routers*」 – ルーター上に Diffserv アーキテクチャーを実装するためのモデルを紹介している

サービス品質に関する情報が掲載されている Web サイト

IETF の Differentiated Services Working Group は、Diffserv Internet Draft へのリンクを含む Web サイト (`http://www.ietf.org/html.charters/diffserv-charter.html`) を管理しています。

Cisco Systems 社や Juniper Networks 社などのルーター製造元は、それぞれ自社の Web サイトで、差別化サービスの製品への実装状況について解説しています。

IPQoS のマニュアルページ

IPQoS のドキュメントには、次のマニュアルページが含まれます。

- `ipqosconf(1M)` - IPQoS 構成ファイルを設定するコマンドについて説明する
- `ipqos(7ipp)` – Diffserv アーキテクチャーモデルの IPQoS 実装について説明する
- `ipgpc(7ipp)` – Diffserv クラシファイアの IPQoS 実装について説明する
- `tokenmt(7ipp)` – IPQoS の `tokenmt` メーターについて説明する
- `tswtclmt(7ipp)` – IPQoS の `tswtclmt` メーターについて説明する
- `dscpmk(7ipp)` – DSCP マーカーモジュールについて説明する
- `dlcosmk(7ipp)` – IPQoS 802.1D ユーザー優先順位マーカーモジュールについて説明する
- `flowacct(7ipp)` – IPQoS フローアカウンティングモジュールについて説明する

- acctadm(1M) – Oracle Solaris 拡張アカウンティング機能を構成するコマンドについて説明する。acctadm コマンドには、IPQoS の拡張が含まれます。

IPQoS によるサービス品質の提供

IPQoS 機能を使用すると、インターネットサービスプロバイダ (ISP) やアプリケーションサービスプロバイダ (ASP) は、顧客ごとに異なるレベルのネットワークサービスを提供できます。同様に、一般企業や教育機関では、この機能を使って内部組織向けのサービスや主要なアプリケーションのサービスを優先できます。

サービスレベル契約の実装

ISP や ASP では、顧客に提示する「サービスレベル契約 (Service-Level Agreement、SLA)」に基づいて IPQoS の構成を行うことができます。個々の SLA では、サービスプロバイダは、価格体系に基づいた一定レベルのネットワークサービスを顧客に保証します。たとえば、プレミアム価格の SLA では、顧客はすべての種類のネットワークトラフィックに対して毎日 24 時間もっとも高い優先順位が与えられます。逆に、中ぐらいの価格の SLA では、業務時間に電子メールだけに高い優先順位が保証されている場合もあります。ほかのトラフィックはすべて、一日 24 時間、中ぐらいの優先順位になります。

一般の組織にとってのサービス品質の保証

一般企業や法人である場合でも、ネットワークにサービス品質機能を提供できます。つまり、特定のグループまたは特定のアプリケーションのトラフィックに対して高レベルまたは低レベルのサービスを保証できます。

サービス品質ポリシーの紹介

サービス品質を実装するには、「サービス品質 *(Quality-of-Service、QoS)* ポリシー」を定義します。QoS ポリシーでは、顧客またはアプリケーションの優先順位、さまざまなカテゴリのトラフィックを処理するアクションなど、各種のネットワーク属性を定義します。組織の QoS ポリシーは IPQoS 構成ファイルに実装します。このファイルは、Oracle Solaris のカーネルに入っている IPQoS モジュールを構成します。IPQoS ポリシーが適用されているホストは、「*IPQoS* 対応システム」とみなされます。

QoS ポリシーは、一般に次のことを定義します。

- 「サービスクラス」と呼ばれるネットワークトラフィックの個別グループ
- クラスごとにネットワークトラフィックの量を調整するための測定基準。これらの測定基準によって、「メータリング」と呼ばれるトラフィック測定プロセスが管理される
- IPQoS システムおよび Diffserv ルーターがパケットフローに適用するアクション。このアクションは「ホップ単位動作 (PHB)」と呼ばれる
- サービスのクラスで必要な統計の収集。たとえば、顧客または特定のアプリケーションが生成したトラフィックなどがある

パケットがネットワークに渡されると、IPQoS 対応システムはパケットヘッダーを評価します。IPQoS システムが行うアクションは、作成した QoS ポリシーに応じて決まります。

QoS ポリシーの設計タスクについては、785 ページの「サービス品質ポリシーの計画」に説明があります。

IPQoS によるネットワーク効率の向上

IPQoS には、サービス品質の実装に伴ってネットワークパフォーマンスの効率を向上させるのに役立ついくつかの機能が含まれています。コンピュータネットワークが拡大すると、ユーザーや高機能なプロセッサの数が増加して、生成されるネットワークトラフィックを管理する必要性も増大します。ネットワークを過度に使用すると、データの消失やトラフィックの輻輳などの症状が現れます。どちらの症状も応答時間を遅らせます。

従来、システム管理者は帯域幅を拡張する方法でネットワークトラフィックの問題に対処してきました。しかし同時に、リンクごとのトラフィックのレベルには、大きなばらつきが見られがちでした。IPQoS を使用すると、既存のネットワーク上のトラフィックを管理しながら、ネットワークの拡大が必要かどうか、またどこに必要かを評価できます。

たとえば、企業や法人の場合は、効率的なネットワークを維持して、トラフィックに関する障害の発生を防ぐ必要があります。また、グループやアプリケーションが割り当てられた以上の帯域幅を消費しないようにする必要もあります。ISP や ASP は、顧客が料金分のレベルのネットワークサービスを確実に受けられるようにネットワークパフォーマンスを管理する必要があります。

ネットワークトラフィックへの帯域幅の影響

IPQoS を使用すると、ネットワークの「帯域幅」、つまりネットワークリンクまたはデバイスが完全に使用された場合に転送できるデータの最大量を調整できま

す。サービス品質を顧客またはユーザーに提供するには、QoS ポリシーで、帯域幅の使用に優先順位を付ける必要があります。IPQoS のメータリングモジュールを使用すると、IPQoS 対応ホスト上の各トラフィッククラスへの帯域幅の割り当て量を測定および管理できます。

ネットワーク上のトラフィックを効果的に管理するには、帯域幅の使用量に関して、次の点を明らかにしておく必要があります。

- ローカルネットワークのトラフィック問題の発生箇所はどこか
- 帯域幅を最適利用するために行うべきことは何か
- サイト内で最優先するべき、重要なアプリケーションはどれか
- 輻輳が発生しやすいアプリケーションはどれか
- 優先順位を落としてもよい、重要度の低いアプリケーションはどれか

サービスクラスを使ったトラフィックの優先順位付け

サービス品質を実現するには、ネットワークトラフィックを分析して、トラフィックを分類する大まかなグループ分けを決定します。次に、それぞれ特徴と優先順位を持つサービスクラスに各グループ分けを整理します。これらのクラスが基本的なカテゴリとなり、それに基づいて組織の QoS ポリシーを作成します。サービスクラスは、管理の対象となるトラフィックグループを代表します。

たとえば、プロバイダがプラチナ、ゴールド、シルバー、ブロンズの各レベルのサービスをそれぞれ異なる利用料金で提供するとします。プラチナレベルの SLA では、プロバイダが顧客用に運営している Web サイト宛ての着信トラフィックに対して、もっとも高い優先順位を保証します。このように、ある顧客の Web サイト宛ての着信トラフィックを 1 つのトラフィッククラスとしてまとめることができます。

企業向けには、部門の要求に基づくサービスクラスを作成できます。また、ネットワークトラフィック内の特定のアプリケーションの優位性に基づくクラスを作成することもできます。

次に、企業向けのトラフィッククラスの例をいくつか示します。

- 特定のサーバー宛ての電子メールや発信 FTP などの、よく使われるアプリケーション。アプリケーションごとに 1 つのクラスを構成できます。これらのアプリケーションは従業員によって絶えず使用されるため、QoS ポリシーで、電子メールと発信 FTP には少量の帯域幅と低い優先順位を割り当てます。
- 一日 24 時間実行の必要がある注文入力データベース。企業にとってのデータベースアプリケーションの重要度に応じて、大量の帯域幅と高い優先順位を割り当てます。
- 人事部門などの、極めて重要な業務または機密業務を行う部署。組織にとっての部署の重要度に応じて、割り当てる優先順位と帯域幅の大きさを決めます。

- 企業の外部向け Web サイトへの呼び出し。このクラスには、適度な大きさの帯域幅と低い優先順位を割り当てる

差別化サービスモデル

IPQoS には次のモジュールがあります。これらのモジュールは、RFC 2475 に定義されている差別化サービス *(Diffserv)* アーキテクチャーの一部です。

- クラシファイア
- メーター
- マーカー

IPQoS では、次の拡張機能が Diffserv モデルに追加されています。

- フローアカウンティングモジュール
- 802.1D データグラムマーカー

このセクションでは、IPQoS で使用する Diffserv モジュールについて簡単に説明します。QoS ポリシーを設定するには、これらのモジュール、その名前、およびその使用目的を認識しておく必要があります。各モジュールの詳細は、843 ページの「IPQoS アーキテクチャーと Diffserv モデル」を参照してください。

クラシファイア (ipgpc) の概要

Diffserv モデルでは、クラシファイアがネットワークトラフィックフローからパケットを選択します。「トラフィックフロー」は、次の IP ヘッダーフィールド内に同一の情報を持つパケットのグループで構成されます。

- 発信元アドレス
- 着信先アドレス
- 発信元ポート
- 着信先ポート
- プロトコル番号

IPQoS では、これらのフィールドを「*5* タプル」と呼びます。

IPQoS のクラシファイアモジュールの名前は `ipgpc` です。`ipgpc`クラシファイアは、トラフィックフローを、IPQoS 構成ファイルに構成されている特性に基づいたクラスに分類します。

`ipgpc` の詳細については、843 ページの「クラシファイアモジュール」を参照してください。

IPQoS クラス

「クラス」とは、似たような特性を共有するネットワークフローのグループのことです。たとえば、ISP は、顧客に提供するさまざまなサービスレベルを表すクラスを定義できます。一方、ASP は、アプリケーションごとに異なるサービスレベルを提供する SLA を定義できます。ASP の QoS ポリシーでは、特定の着信先 IP アドレス宛ての発信 FTP トラフィックを 1 つのクラスにまとめることができます。ある企業の外部 Web サイトからの発信トラフィックも、1 つのクラスとして定義できます。

トラフィックをいくつかのクラスに分類することは、QoS ポリシーを計画する際に欠かせない作業の 1 つです。`ipqosconf` ユーティリティーを使用してクラスを作成するときは、実際には`ipgpc` クラシファイアを構成しています。

クラスを定義するには、788 ページの「QoS ポリシーのクラスを定義する方法」を参照してください。

IPQoS フィルタ

「フィルタ」は、「セレクタ」と呼ばれるパラメータを含む規則のセットです。各フィルタは、必ず 1 つのクラスを指定する必要があります。IPQoS は、パケットを各フィルタのセレクタと突き合わせて、パケットがフィルタのクラスに属しているかどうかを調べます。さまざまなセレクタを使用してパケットにフィルタをかけることができます。セレクタの例として、IPQoS 5 タプルなどのよく使うパラメータを次に示します。

- 発信元および着信先のアドレス
- 発信先および着信先のポート
- プロトコル番号
- ユーザー ID
- プロジェクト ID
- 差別化サービスコードポイント (DSCP)
- インタフェースインデックス

たとえば、簡単なフィルタに値が 80 の宛先ポートが含まれているとします。`ipgpc` クラシファイアは、宛先ポート 80 (HTTP) 向けのパケットをすべて選択し、QoS ポリシーの指示どおりに選択したパケットを処理します。

フィルタの作成については、791 ページの「QoS ポリシーにフィルタを定義する方法」を参照してください。

メーター (tokenmt および tswtclmt) の概要

Diffserv モデルでは、「メーター」はトラフィックフローの転送速度をクラス単位で追跡します。メーターは、該当する結果 (outcome) を得るために、フローの実際の転送速度が構成された速度にどれだけ適合しているかを評価します。そして、トラ

フィックフローの結果に基づいて、次のアクションを選択します。次のアクションでは、パケットを別のアクションに送信したり、それ以上処理しないでネットワークに戻したりできます。

IPQoS のメーターは、ネットワークフローが、QoS ポリシーでそのクラスに定義されている転送速度に適合しているかどうかを調べます。IPQoS には、次の 2 つのメータリングモジュールがあります。

- `tokenmt` - 2 トークンバケットメータリング方式を使用
- `tswtclmt` - タイムスライディングウィンドウメータリング方式を使用

どちらのメータリングモジュールも、 赤、黄、緑という 3 つの結果を識別します。結果ごとに実行させたいアクションは、`red_action_name`、`yellow_action_name`、および `green_action_name` のパラメータで定義します。

また、`tokenmt` をカラーアウェアとして構成することもできます。カラーアウェアとして構成されているメータリングインスタンスでは、パケットのサイズ、DSCP、トラフィックの転送速度、および構成されたパラメータを使って結果を求めます。メーターは、DSCP を使用してパケットの結果を緑、黄、赤にマッピングします。

IPQoS メーターのパラーメータの定義については、792 ページの「フロー制御を計画する方法」を参照してください。

マーカー (dscpmk および dlcosmk) の概要

Diffserv モデルでは、「マーカー」は転送動作を表す値をパケットに付けます。「マーキング」とは、パケットをネットワークに転送する方法を示す値を、そのパケットのヘッダーに付加するプロセスのことです。

IPQoS には、次の 2 つのマーカーモジュールが含まれています。

- `dscpmk` – IP パケットヘッダーの DS フィールドに「差別化サービスコードポイント (*DSCP*)」と呼ばれる数値を付けます。Diffserv 対応ルーターは、この DS コードポイントを使って、適切な転送動作をパケットに適用できます。
- `dlcosmk` – Ethernet フレームヘッダーの仮想ローカルエリアネットワーク (VLAN) タグに「ユーザー優先順位」と呼ばれる数値を付けます。ユーザー優先順位は、データグラムに適用される適切な転送動作を定義する「サービスクラス *(CoS)*」のことです。

  `dlcosmk` は、IPQoS の追加機能として IETF によって設計されたものであり、Diffserv モデルの一部ではありません。

QoS ポリシーのマーカー戦略の実装については、795 ページの「転送動作を計画する方法」を参照してください。

フローアカウンティング (flowacct) の概要

IPQoS では、flowacct アカウンティングモジュールが Diffserv モデルに追加されます。flowacct を使用すると、トラフィックフローに関する統計情報を取得し、SLA に合わせて顧客に課金できます。フローアカウンティングは、容量計画やシステムの監視にも役立ちます。

flowacct モジュールを acctadm コマンドと組み合わせて、アカウンティングログファイルを作成できます。基本的なログには、次に示すように、IPQoS 5 タプルのほかに 2 つの属性が記録されます。

- 発信元アドレス
- 発信元ポート
- 着信先アドレス
- 着信先ポート
- プロトコル番号
- パケット数
- バイト数

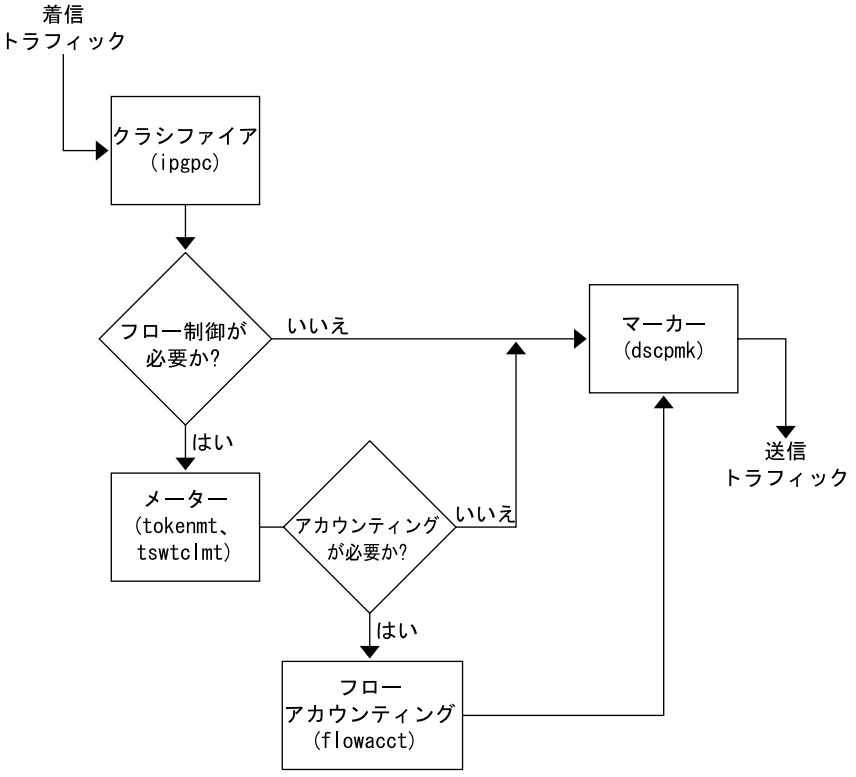
838 ページの「トラフィックフローに関する情報の記録」および flowacct(7ipp) や acctadm(1M) のマニュアルページに説明されているとおり、その他の属性の統計を集めることもできます。

フローアカウンティング戦略の計画については、797 ページの「フローアカウンティングを計画する方法」を参照してください。

トラフィックが IPQoS モジュールをどのように通過するか

次の図は、着信トラフィックが IPQoS モジュールのいくつかを通過するときに取りうる経路を示しています。

図 29–1　diffserv モデルの IPQoS 実装を通過するトラフィックフロー



この図は、IPQoS 対応マシンにおける一般的なトラフィックフローシーケンスを示しています。

1. クラシファイアが、システムの QoS ポリシーのフィルタリング条件に適合するすべてのパケットをパケットストリームから選択します。
2. 選択したパケットが評価されて、次に実行されるアクションが決められます。
3. クラシファイアが、フロー制御を必要としないトラフィックをマーカーに送信します。
4. フロー制御が必要なトラフィックは、メーターに送信されます。
5. メーターは、構成速度を実施します。次に、トラフィックの適合値をフロー制御されているパケットに割り当てます。
6. フロー制御されるパケットが評価されて、アカウンティングが必要であるかどうかが判断されます。
7. メーターが、フローアカウンティングを必要としないトラフィックをマーカーに送信します。

8. フローアカウンティングモジュールが、受信したパケットに関する統計情報を収集します。次に、それらのパケットをマーカーに送信します。
9. マーカーが DS コードポイントをパケットヘッダーに割り当てます。この DSCP は、Diffserv 対応システムがパケットに適用するべきホップ単位動作 (PHB) を示します。

IPQoS 対応ネットワークでのトラフィック転送

このセクションでは、IPQoS 対応ネットワークでのパケット転送に関係するいくつかの要素について簡単に説明します。IPQoS 対応システムは、着信先としてそのシステムの IP アドレスを持つ、ネットワークストリーム上のパケットを処理します。そして、IPQoS システムの QoS ポリシーをパケットに適用して、差別化サービスを確立します。

DS コードポイント

DS コードポイント (DSCP) は、マークされたパケットに対して Diffserv 対応システムが実行するアクションをパケットヘッダーに定義します。diffserv アーキテクチャーは、使用する IPQoS 対応システムと diffserv ルーターに対して一連の DS コードポイントを定義します。また、DSCP に対応する「転送動作」と呼ばれる一連の処理も定義します。IPQoS 対応システムは、パケットヘッダーにある DS フィールドの優先度ビットに DSCP を付けます。DSCP 値を持つパケットを受信すると、ルーターは、その DSCP と関連付けられた転送動作を実行します。次にパケットはネットワーク上に送出されます。

注 – `dlcosmk` マーカーは、DSCP を使用しません。代わりに、Ethernet フレームヘッダーに CoS 値を付加します。VLAN デバイスを使用するネットワークで IPQoS を構成する予定の場合は、849 ページの「マーカーモジュール」を参照してください。

ホップ単位動作

Diffserv 用語では、DSCP に割り当てられる転送動作を「ホップ単位動作 *(Per-Hop Behavior、PHB)*」と呼びます。PHB は、Diffserv 対応システム上で、マークされたパケットの転送がほかのトラフィックに比べて優先される度合いを定義します。この優先度によって、IPQoS 対応システムまたは Diffserv ルーターが、マークされたパケットを転送するかドロップするかが最終的に決まります。パケットが転送された場合、パケットがその着信先への途中で通過する各 Diffserv ルーターは、同じ PHB をそのパケットに適用します。ただし、別の Diffserv システムによって DSCP が変更された場合は例外です。PHB の詳細については、849 ページの「パケット転送での `dscpmk` マーカーの使用」を参照してください。

PHB の目的は、指定された量のネットワークリソースを連続したネットワーク上のトラフィッククラスに提供することです。この目的は、QoS ポリシーで達成できます。トラフィックフローが IPQoS 対応システムを離れたときのトラフィッククラスの優先順位を示す DSCP を定義します。優先度は、高い優先度 (ドロップ率が低い) から低い優先度 (ドロップ率が高い) の範囲になります。

たとえば、QoS ポリシーによってあるトラフィッククラスにドロップ率が低い PHB を保証する DSCP 割り当てることができます。このトラフィッククラスは、ドロップ率が低い優先度の PHB をすべての Diffserv 対応ルーターから与えられ、このクラスのパケットの帯域幅が保証されます。QoS ポリシーに別の DSCP を追加して、ほかのトラフィッククラスにさまざまなレベルの優先度を割り当てることもできます。優先度の低いパケットには、パケットの DSCP に示された優先順位に応じた帯域幅を、Diffserv システムが割り当てます。

IPQoS は、Diffserv アーキテクチャーに定義されている 2 種類の転送動作、完全優先転送 (EF) と相対的優先転送 (AF) をサポートしています。

完全優先転送

「完全優先転送 *(Expedited Forwarding、EF)*」PHB は、EF 関連の DSCP を持つトラフィッククラスが一番高い優先順位を割り当てられることを保証します。EF DSCP のトラフィックは、キューに格納されません。EF では、低損失、低遅延、低ジッターのサービスを提供します。EF の推奨 DSCP は、101110 です。101110 が付加されたパケットは、着信先への途中で Diffserv 対応ネットワークを通過するときに、ドロップ率の低い優先度を与えられます。EF DSCP は、プレミアム SLA を持つ顧客またはアプリケーションに優先順位を割り当てるときに使用してください。

相対的優先転送

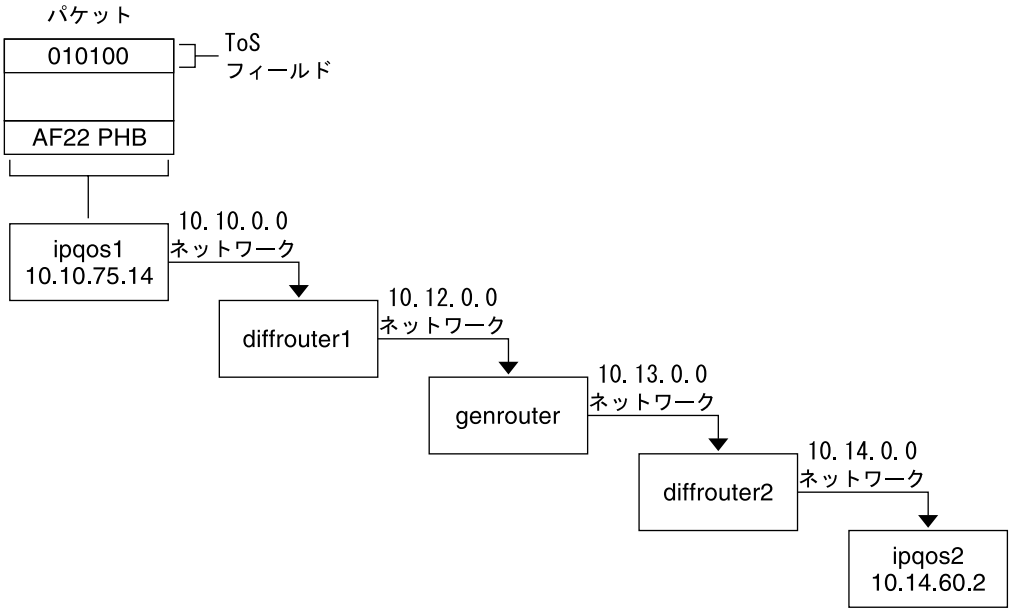
相対的優先転送 *(AF)* PHB には、パケットに割り当てられる転送クラスが 4 種類あります。表 34–2 に示すように、各転送クラスは 3 つのドロップ優先順位を提供します。

AF コードポイントには、顧客やアプリケーションにさまざまなレベルのサービスを割り当てる機能があります。QoS ポリシーでは、QoS ポリシーを計画するときに、ネットワーク上のトラフィックやサービスに優先順位を付けておきます。そうすれば、優先順位の付いたトラフィックにそれぞれ異なる AF レベルを割り当てることができます。

diffserv 環境でのパケット転送

次の図は、Diffserv 対応環境を部分的に備えた企業のイントラネット部分を示しています。このシナリオでは、ネットワーク `10.10.0.0` および `10.14.0.0` 上のホストはすべて IPQoS に対応しており、ローカルルーターは Diffserv に対応しています。しかし、間にある 2 つのネットワークは Diffserv 用に構成されていません。

図 29–2 diffserv 対応ネットワークのホップ間のパケット転送



次の手順は、前の図に示すパケットの流れをたどっています。この手順は、ホスト `ipqos1` で発生するパケットの前進で開始されます。手順は数回のホップでホスト `ipqos2` に続きます。

1. `ipqos1` のユーザーが `ftp` コマンドを実行して、3 ホップ離れたところにあるホスト `ipqos2` にアクセスしようとします。
2. `ipqos1` は、結果生じたパケットフローに QoS ポリシーを適用します。`ipqos1` は、`ftp` トラフィックを正常に分類します。

   システム管理者は、ローカルネットワーク 10.10.0.0 に起因するすべての発信 `ftp` トラフィックに対するクラスを作成済みです。`ftp` クラスのトラフィックには、AF22 ホップ単位動作 (クラス 2、中程度のドロップ優先度) が割り当てられます。`ftp` クラスには、2Mb/秒のトラフィックフロー速度が構成されます。
3. `ipqos-1` は、`ftp` フローを測定し、フローが 2 Mbit/秒を超過していないかどうかを判断します。
4. `ipqos1` 上のマーカーが、発信 `ftp` パケットの DS フィールドに 010100 DSCP (AF22 PHB に対応している) を付加します。
5. ルーター `diffrouter1` は、`ftp` パケットを受信します。次に、DSCP をチェックします。`diffrouter1` が輻輳している場合、AF22 でマークされているパケットは振り落とされます。
6. `diffrouter1` のファイルで AF22 に対して構成されている PHB に合わせて、`ftp` トラフィックが次のホップに転送されます。

7. ftp トラフィックがネットワーク 10.12.0.0 を通って genrouter に進みます。genrouter は Diffserv に対応していません。その結果、トラフィックはベストエフォートの転送動作を与えられます。
8. genrouter が ftp トラフィックをネットワーク 10.13.0.0 に渡し、diffrouter2 がそのトラフィックを受け取ります。
9. diffrouter2 は Diffserv に対応しています。したがって、ルーターのポリシーで AF22 パケットに対して定義されている PHB に合わせて、ftp パケットをネットワークに転送します。
10. ipqos2 は、ftp トラフィックを受信します。 ipqos2 は、ipqos1 のユーザーにユーザー名とパスワードの入力を促します。

◆ ◆ ◆ 第 30 章

IPQoS 対応ネットワークの計画 (タスク)

Oracle Solaris が動作する任意のシステムで IPQoS を構成できます。そして、IPQoS システムを Diffserv 対応ルーターと組み合わせると、イントラネット上で差別化サービスを提供したり、トラフィック管理を行うことができます。

この章では、IPQoS 対応システムを Diffserv 対応ネットワークに追加するための計画タスクについて説明します。この章の内容は次のとおりです。

一般的な IPQoS の構成計画 (タスクマップ)

IPQoS などの差別化サービスをネットワーク上に実装する場合は、綿密な計画が必要です。各 IPQoS 対応システムの位置と機能だけではなく、各システムとローカルネットワーク上のルーターとの関係についても考慮してください。次のタスクマップに、ネットワーク上で IPQoS を実装するための主な計画タスクの一覧と、各タスクを完了するための手順へのリンクを示します。

タスク	説明	手順
1. IPQoS 対応システムを取り入れた Diffserv ネットワークトポロジを計画する	さまざまな Diffserv ネットワークトポロジを考慮して、自分のサイトにもっとも適したソリューションを考える	782 ページの「diffserv ネットワークトポロジの計画」

タスク	説明	手順
2. IPQoS システムによって提供する各種サービスを計画する	ネットワークが提供するサービスの種類をいくつかの サービスレベル契約 (service-level agreement、SLA) に分類する	785 ページの「サービス品質ポリシーの計画」
3. IPQoS システムごとに QoS ポリシーを計画する	各 SLA の実装に必要なクラス、メータリング、およびアカウンティング機能を決める	785 ページの「サービス品質ポリシーの計画」
4. 必要であれば、Diffserv ルーターのポリシーを計画する	IPQoS システムで使用する Diffserv ルーターのスケジューリングポリシーおよびキューイングポリシーを決める	キューイングポリシーおよびスケジューリングポリシーについては、ルーターのドキュメントを参照

diffserv ネットワークトポロジの計画

ネットワークに差別化サービスを提供するには、IPQoS 対応システムが最低 1 つと Diffserv 対応ルーターが 1 台必要です。このセクションで説明するように、この基本構成はさまざまな方法で拡張できます。

diffserv ネットワークのハードウェア計画

一般に、IPQoS はサーバーやサーバー統合上 (Oracle の Sun Enterprise(tm) サーバーなど) で実行します。一方、ネットワークのニーズに応じて、UltraSPARC® システムなどのデスクトップシステムで IPQoS を実行することもできます。

次に、IPQoS 構成に使用できるシステムの例を示します。

- Oracle Solaris システム。Web サーバーやデータベースサーバーなどの各種サービスを提供する
- アプリケーションサーバー。電子メール、FTP などのよく使われるネットワークアプリケーションを提供する
- Web キャッシュサーバーまたはプロキシサーバー
- IPQoS 対応サーバーファームのネットワーク。Diffserv 対応ロードバランサによって管理される
- ファイアウォール。単一の異機種システム混在ネットワークのトラフィックを管理する
- 仮想ローカルエリアネットワーク (LAN) の一部である IPQoS システム

IPQoS システムを導入しようとするネットワークトポロジでは、Diffserv 対応ルーターがすでに機能していることもありえます。もし、現在使用しているルーターが Diffserv に対応していない場合は、Cisco Systems 社や Juniper Networks 社な

どのルーター製造元が提供する Diffserv のソリューションを検討してください。ローカルルーターが Diffserv を実装していないと、パケットのマークは評価されずに次のホップへ渡されてしまいます。

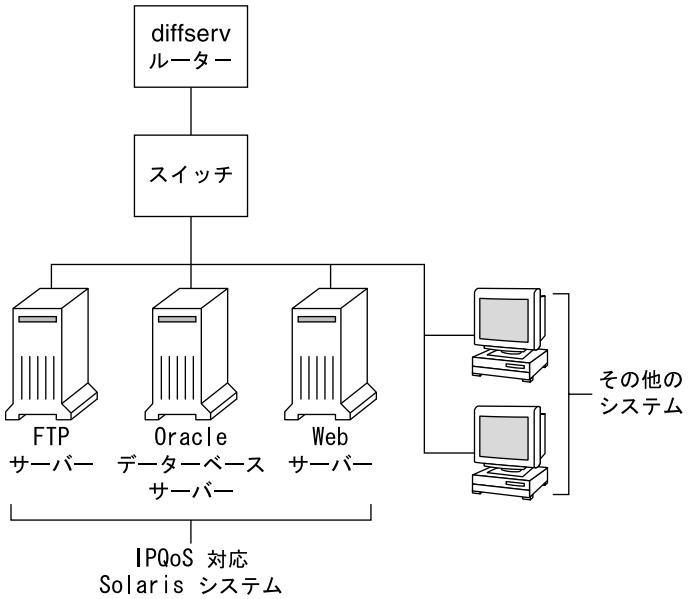
IPQoS ネットワークトポロジ

このセクションでは、ネットワーク上のさまざまなニーズを満たす IPQoS 計画を図で説明します。

個々のホストでの IPQoS

次の図は、IPQoS 対応システムの単一ネットワークを示しています。

図 30–1 ネットワークセグメント上の IPQoS システム



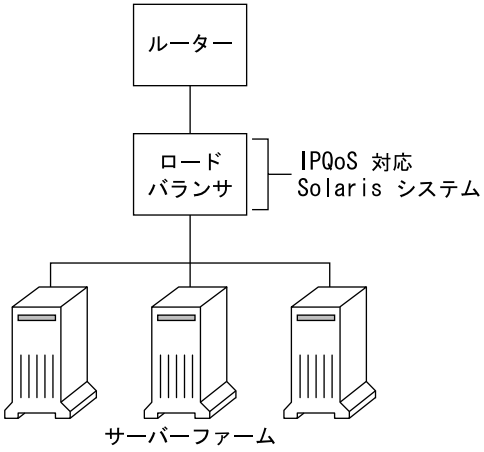
このネットワークは、ある企業のイントラネットの一部分です。アプリケーションサーバーや Web サーバーで IPQoS を有効にすると、各 IPQoS システムが発信トラフィックを送出する速度を制御できます。ルーターが Diffserv に対応していれば、着信トラフィックおよび発信トラフィックをさらに制御できます。

このガイドで取り上げる例では、「個々のホストでの IPQoS」シナリオを使用します。このガイドを通して使用されているサンプルトポロジについては、図 30–4 を参照してください。

サーバーファームのネットワークでの IPQoS

次の図には、複数の異種サーバーファームを備えるネットワークを示します。

図 30–2 IPQoS 対応サーバーファームのネットワーク



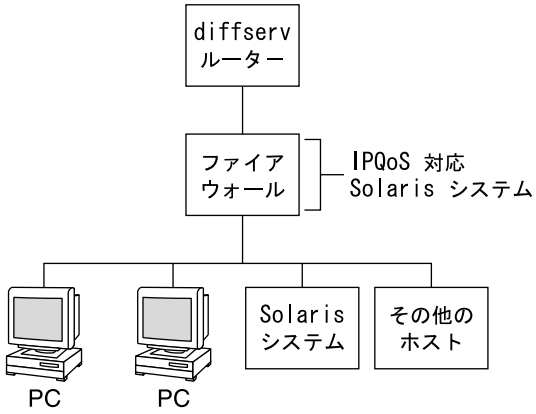
このようなトポロジでは、ルーターが Diffserv に対応しているため、着信トラフィックと発信トラフィックの両方をキューに入れたり評価したりできます。また、ロードバランサも Diffserv 対応システムであるため、サーバーファームは IPQoS 対応となります。ロードバランサは、ユーザー ID やプロジェクト ID などのセレクタを使用することによって、ルーター以外で追加フィルタリングを提供できます。これらのセレクタは、アプリケーションデータに含まれます。

このシナリオでは、フロー制御とトラフィック転送を行って、ローカルネットワーク上の輻輳を管理しています。また、このシナリオでは、サーバーファームからの発信トラフィックが原因でイントラネットのほかの部分が過負荷状態になるのを防いでいます。

ファイアウォールでの IPQoS

次の図は、ファイアウォールによってほかのセグメントから保護されている、企業ネットワークのセグメントの 1 つを示しています。

図 30–3 IPQoS 対応のファイアウォールによって保護されているネットワーク



このシナリオでは、トラフィックはまず Diffserv 対応ルーターに入り、そこでフィルタにかけられ、キューに入れられます。次に、ルーターによって転送された着信トラフィックはすべて、IPQoS 対応のファイアウォールに進みます。IPQoS を使用するには、ファイアウォールで IP 転送スタックをバイパスしないでください。

ファイアウォールのセキュリティーポリシーによって、着信トラフィックを内部ネットワークに入れて良いかどうかが決まります。QoS ポリシーは、ファイアウォールを通過した着信トラフィックのサービスレベルを制御します。QoS ポリシーによっては、発信トラフィックに転送動作を付けることもできます。

サービス品質ポリシーの計画

サービス品質 (QoS) ポリシーを計画するときは、ネットワークが提供するサービスの確認、分類、そして優先順位付けを行う必要があります。また、利用できる帯域幅の大きさを評価して、各トラフィッククラスがネットワークに送出される速度を決める必要もあります。

QoS ポリシー計画の手掛かり

IPQoS 構成ファイルで必要な情報を含む形式で QoS ポリシーの計画情報を収集します。たとえば、次のテンプレートを使って、IPQoS 構成ファイルで使用する主なカテゴリの情報を表にできます。

表 30–1 QoS 計画テンプレート

クラス	優先順位	フィルタ	セレクタ	レート	転送動作をどうするか	アカウンティングが必要か
クラス 1	1	フィルタ 1 フィルタ 3	セレクタ 1 セレクタ 2	メーターの速度 (メーターの種類による)	マーカーのドロップ優先度	フローアカウンティング統計情報を必要とする
クラス 1	1	フィルタ 2	セレクタ 1 セレクタ 2	なし	なし	なし
クラス 2	2	フィルタ 1	セレクタ 1 セレクタ 2	メーターの速度 (メーターの種類による)	マーカーのドロップ優先度	フローアカウンティング統計情報を必要とする
クラス 2	2	フィルタ 2	セレクタ 1 セレクタ 2	なし	なし	なし

各カテゴリをいくつかに分けて、QoS ポリシーをさらに細かく定義できます。以降のセクションでは、テンプレートに示されているカテゴリの情報を入手する方法について説明します。

QoS ポリシーの計画 (タスクマップ)

このタスクマップでは、QoS ポリシーを計画するための主なタスクの一覧と、各タスクの実行手順へのリンクを示します。

タスク	説明	手順
1. IPQoS をサポートするようネットワークトポロジを設計する	差別化サービスを提供するネットワーク上のホストとルーターを特定する	787 ページの「IPQoS 用のネットワークを準備する方法」
2. ネットワーク上のサービスを分類するためのクラスを定義する	自分のサイトで提供するサービスの種類と SLA を調べ、これらのサービスを分類するためのトラフィッククラスを決める	788 ページの「QoS ポリシーのクラスを定義する方法」
3. クラス用のフィルタを定義する	特定のトラフィッククラスをネットワークトラフィックフローから取り出すための最善の方法を決める	791 ページの「QoS ポリシーにフィルタを定義する方法」

タスク	説明	手順
4. IPQoS システムから送出されるトラフィックを測定するためのフロー制御速度を定義する	トラフィッククラスごとに容認できるフロー速度を決める	792 ページの「フロー制御を計画する方法」
5. QoS ポリシーで使用する DSCP すなわちユーザー優先順位の値を定義する	トラフィックフローがルーターまたはスイッチによって処理されるときに割り当てられる転送動作を決める方法を計画する	795 ページの「転送動作を計画する方法」
6. 必要であれば、ネットワーク上のトラフィックフローに対する統計監視計画を設定する	トラフィッククラスを評価して、アカウンティングまたは統計上の目的で監視するトラフィックフローを決める	797 ページの「フローアカウンティングを計画する方法」

注 – このセクションの残りの部分では、IPQoS 対応システムの QoS ポリシーを計画する方法について説明します。Diffserv ルーター向けの QoS ポリシーの計画を立てるには、ルーターのドキュメントおよびルーター製造元の Web サイトを参照してください。

▼ IPQoS 用のネットワークを準備する方法

次の手順では、QoS ポリシーを作成する前に行う一般的な計画タスクを示します。

1. ネットワークトポロジを見直します。次に、**IPQoS** システムと **Diffserv** ルーターを使用する戦略の計画を作成します。
   トポロジの例は、782 ページの「diffserv ネットワークトポロジの計画」を参照してください。

2. **IPQoS** を必要とするホスト、または **IPQoS** サービスの有力な候補となるホストをトポロジで特定します。

3. **IPQoS** 対応後に同じ **QoS** ポリシーを共用できそうなシステムを調べます。
   たとえば、ネットワーク上のすべてのホストで IPQoS を有効にする場合は、同じ QoS ポリシーを使用できるホストをすべて特定します。各 IPQoS 対応システムにはローカル QoS ポリシーが必要で、そのポリシーはそれぞれの IPQoS 構成ファイルに実装されます。ただし、IPQoS 構成ファイルを 1 つだけ作成し、これを同じ QoS ポリシー要件を持つすべてのシステムにコピーして使うこともできます。

4. ネットワーク上の **Diffserv** ルーターに対して必要な計画タスクをすべて確認し、実行します。
   詳細については、ルーターのドキュメントとルーター製造元の Web サイトを参照してください。

▼ QoS ポリシーのクラスを定義する方法

QoS ポリシーを定義するための最初の手順は、トラフィックフローをクラスに整理することです。Diffserv ネットワーク上のトラフィックの種類ごとにクラスを作成する必要はありません。また、計画したネットワークトポロジによっては、IPQoS 対応システムごとに異なる QoS ポリシーを作成する必要があります。

注 – クラスの概要は、772 ページの「IPQoS クラス」を参照してください。

次の手順では、787 ページの「IPQoS 用のネットワークを準備する方法」に従って、IPQoS 対応とするネットワーク上のシステムを決定済みであることを前提としています。

1. **QoS ポリシー情報を整理する QoS 計画テーブルを作成します。**
   提案については、表 30–1 を参照してください。

2. **ネットワーク上にあるすべての QoS ポリシーについて、残りの手順を実行します。**

3. **QoS ポリシーで使用するクラスを定義します。**
   次の質問は、ネットワークトラフィックを解析してクラス定義を行うためのガイドラインとなります。

   - 顧客にサービスレベル契約を提示しているか

     提示する場合は、顧客に提供する複数の SLA 間の相対的な優先レベルを評価します。保証されている優先レベルが異なる顧客に同じアプリケーションを提供する場合もあります。

     たとえば、企業が各顧客に対して Web サイトの運営サービスを提供するとします。この場合は、顧客の Web サイトごとに 1 つのクラスを定義する必要があります。SLA は、サービスレベルの 1 つとしてプレミアム Web サイトを提供するとします。別の SLA は、割引顧客に「ベストエフォート型」のパーソナル Web サイトを提供するとします。この場合は、Web サイトのクラスが異なるだけでなく、クラスに割り当てられる PHB も異なる可能性があります。

   - **IPQoS** システムでは、フロー制御を必要としそうなよく使われるアプリケーションを提供しているか

     よく使われるアプリケーションを提供しているために大量のトラフィックが生成されるサーバーの場合、IPQoS を有効にするとネットワークパフォーマンスが向上します。そのようなアプリケーションの例として、電子メール、ネットワークニュース、FTP などがあげられます。該当する場合は、サービスの種類ごとに着信トラフィックと発信トラフィックのクラスを別々に作成することを検討してください。たとえば、メールサーバーの QoS ポリシーに対して、`mail-in` クラスと `mail-out` クラスを作成します。

- ネットワークでもっとも高い優先順位の転送動作を必要とする特定のアプリケーションを実行しているか

  優先順位が最も高い転送動作を必要とする重要なアプリケーションには、ルーターのキューで最も高い優先順位を与える必要があります。一般的な例は、ストリーミングビデオやストリーミングオーディオです。

  まず、優先順位の高いこれらのアプリケーションに対して、それぞれ着信クラスと発信クラスを定義します。次に、定義したクラスを、それらのアプリケーションを提供する IPQoS 対応システムと Diffserv ルーターの両方の QoS ポリシーに追加します。

- 帯域幅を大量に消費するため、ネットワークで制御を必要とするトラフィックフローが発生したことがあるか

  `netstat`、`snoop` などのネットワーク監視ユーティリティーを使用して、ネットワーク上で問題のあるトラフィックの種類を検出します。これまでに作成したクラスを確認し、問題のトラフィックカテゴリが未定義である場合は、このカテゴリに対して新しいクラスを作成します。問題のトラフィックカテゴリのクラスが定義済みである場合は、このトラフィックを制御するメーターの速度を定義します。

  問題のあるトラフィックのクラスは、ネットワーク上の IPQoS 対応システムごとに作成します。これによって、各 IPQoS システムは、問題のあるトラフィックを受け取った場合に、トラフィックフローをネットワークに送出する速度を制限できるようになります。また、Diffserv ルーターの QoS ポリシーにもこれらの問題のあるクラスを必ず定義してください。これによって、diffserv ルーターは、QoS ポリシーの構成に従って、問題のあるフローをキューに入れたりスケジュールしたりできるようになります。

- 特定の種類のトラフィックに対して統計情報を取得する必要があるか

  SLA をざっと確認すると、どのタイプの顧客のトラフィックにアカウンティングが必要であるかがわかります。自分のサイトで SLA を提供している場合は、アカウンティングを必要とするトラフィックのクラスはおそらく作成済みです。また、クラスを定義して、監視しているトラフィックフローの統計収集を可能にすることもできます。さらに、セキュリティー上の理由でアクセスを制限するトラフィックのクラスも作成できます。

4 手順 **1** で作成した **QoS** 計画テーブルで定義したクラスを一覧にします。

5 優先レベルを各クラスに割り当てます。

たとえば、優先レベル 1 がもっとも高い優先順位のクラスを表すようにし、それ以降の優先レベルを残りのクラスに割り当てます。割り当てた優先レベルの目的は、クラスを整理することだけです。QoS ポリシーテンプレートで設定した優先レベルは、実際に IPQoS によって使用されるわけではありません。また、QoS ポリシーによっては、同じ優先順位を複数のクラスに割り当てることもできます。

6 クラスの定義の次は、791 ページの「QoS ポリシーにフィルタを定義する方法」の説明に従って、各クラスのフィルタを定義します。

参考 クラスの優先順位付け

クラスを作成するうちに、どのクラスの優先順位が高いか、中程度か、ベストエフォートでよいかをすぐに認識できます。795 ページの「転送動作を計画する方法」に説明されているとおり、クラスの優先順位を設定する良いスキームは、出力トラフィックにホップ単位の動作を割り当てる場合に、特に重要になります。

PHB をクラスに割り当てるほかに、そのクラスのフィルタに優先順位セレクタを定義することもできます。優先順位セレクタは、IPQoS 対応ホストでのみ有効です。たとえば、同じ速度と同じ DSCP を持ついくつかのクラスが、IPQoS システムから送出されるときに帯域幅をめぐって競合することがあるとします。このような場合、各クラスの優先順位セレクタによって、ほかの点ではまったく同じ評価のクラスに割り当てられるサービスレベルをさらに細かく順序付けることができます。

フィルタの定義

パケットフローを特定のクラスのメンバーとして識別するには、フィルタを作成します。各フィルタには、パケットフローの評価基準を定義するセレクタがいくつか含まれています。IPQoS 対応システムは、次にセレクタの基準を使用して、トラフィックフローからパケットを抽出します。そして、IPQoS システムは、パケットとクラスを関連付けます。 フィルタの紹介は、772 ページの「IPQoS フィルタ」を参照してください。

次の表に、もっとも一般的に使用されるセレクタを示します。最初の 5 つのセレクタは、IPQoS 5 タプルを表し、IPQoS システムがパケットをフローのメンバーとして識別するときに使用します。セレクタの完全なリストについては、表 34–1 を参照してください。

表 30–2 一般的な IPQoS セレクタ

名前	定義
saddr	発信元アドレス
daddr	着信先アドレス
sport	発信元ポート番号。/etc/services に定義されている既知のポート番号、またはユーザー定義のポート番号を使用できる
dport	着信先ポート番号
プロトコル	IP プロトコル番号またはプロトコル名。/etc/protocols のトラフィックフロータイプに割り当てられる

表 30–2 一般的な IPQoS セレクタ (続き)

名前	定義
ip_version	使用するアドレス指定方式。IPv4 または IPv6 を使用。デフォルトは IPv4。
dsfield	DS フィールドの内容、つまり DSCP。このセレクタは、特定の DSCP が付いている着信パケットを取り出すために使用する
priority	クラスに割り当てられている優先レベル。詳細は、788 ページの「QoS ポリシーのクラスを定義する方法」を参照
user	UNIX のユーザー ID またはユーザー名。上位アプリケーションの実行時に使用される
projid	プロジェクト ID。上位アプリケーションの実行時に使用される
direction	トラフィックフローの方向。LOCAL_IN、 LOCAL_OUT、FWD_IN、または FWD_OUT のいずれかの値

注 – セレクタは正しい判断のもとで選択してください。また、クラスのパケットを取り出すのに必要なものだけを使用してください。定義するセレクタが多いほど、IPQoS パフォーマンスに与える影響も大きくなります。

▼ QoS ポリシーにフィルタを定義する方法

始める前に 次の手順を実行する前に、788 ページの「QoS ポリシーのクラスを定義する方法」の手順を完了しておく必要があります。

1 **788 ページの「QoS ポリシーのクラスを定義する方法」で作成した QoS 計画テーブルの各クラスに 1 つ以上のフィルタを作成します。**

必要であれば、1 つのクラスの着信トラフィックと発信トラフィックに対して、フィルタを別々に作成することを検討してください。たとえば、ftp-in フィルタと ftp-out フィルタを IPQoS 対応の FTP サーバーの QoS ポリシーに追加します。そうすれば、基本セレクタに加えて、該当する方向セレクタも定義できます。

2 **クラスのフィルタごとにセレクタを最低 1 つ定義します。**

表 30–1 で紹介されている QoS 計画テーブルを使用して、定義したクラスのフィルタを埋めます。

例 30–1 FTP トラフィック向けのフィルタの定義

次の表は、出力 FTP トラフィック向けにフィルタを定義する方法を示す例です。

クラス	優先順位	フィルタ	セレクタ
ftp-traffic	4	ftp-out	saddr 10.190.17.44 daddr 10.100.10.53 sport 21 direction LOCAL_OUT

参照
- フロー制御スキームを定義するには、792 ページの「フロー制御を計画する方法」を参照してください。
- フローがネットワークストリームに戻るときのフローの転送動作を定義するには、795 ページの「転送動作を計画する方法」を参照してください。
- 特定の種類のトラフィックのフローアカウンティングを計画するには、797 ページの「フローアカウンティングを計画する方法」を参照してください。
- QoS ポリシーにさらにクラスを追加するには、788 ページの「QoS ポリシーのクラスを定義する方法」を参照してください。
- QoS ポリシーにさらにフィルタを追加するには、791 ページの「QoS ポリシーにフィルタを定義する方法」を参照してください。

▼ フロー制御を計画する方法

フロー制御では、クラスのトラフィックフローを測定し、定義された速度でネットワークにパケットを送出します。フロー制御を計画するときは、IPQoS メータリングモジュールが使用するパラメータを定義します。メーターは、トラフィックがネットワークに送出される速度を決定します。メータリングモジュールの紹介は、772 ページの「メーター (tokenmt および tswtclmt) の概要」を参照してください。

次の手順は、791 ページの「QoS ポリシーにフィルタを定義する方法」の説明に従って、フィルタとセレクタを定義していることが前提になります。

1. ネットワークの最大帯域幅を調べます。
2. ネットワークでサポートされている **SLA** を見直します。顧客と各顧客に保証されているサービスの種類を特定します。

   一定レベルのサービスを保証するには、顧客によって生成される特定のトラフィッククラスを計測する必要があります。
3. **788 ページの「QoS ポリシーのクラスを定義する方法」**で作成したクラスの一覧を見直します。

   SLA に対応付けられているクラス以外に計測する必要があるクラスがあるかどうか確認します。

たとえば、IPQoS システムが大量のトラフィックを生成するアプリケーションを実行するとします。この場合は、そのアプリケーションのトラフィックを分類したあと、フローのパケットがネットワークに戻される速度を制御して、フローを計測します。

注 – すべてのクラスを計測する必要があるとは限りません。クラスのリストを確認するときは、このガイドラインに留意してください。

---

4 各クラスのどのフィルタがフロー制御に必要なトラフィックを選択するのかを判断します。次に、メータリングを必要とするクラスのリストを精緻化します。

複数のフィルタを持つクラスでも 1 つのフィルタに対してだけ計測を必要とする場合もあります。たとえば、あるクラスの着信トラフィックと発信トラフィックに対してフィルタを定義したとします。しかし、結果的にどちらかの方向のトラフィックしかフロー制御を必要としない場合があります。

5 フロー制御を行うクラスごとにメーターモジュールを選択します。

作成した QoS 計画表のメーター欄にモジュール名を追加します。

6 計測するクラスごとの速度を構成表に追加します。

`tokenmt` モジュールを使用する場合は、次の速度を bps で定義する必要があります。

- 認定速度
- 最大速度

特定のクラスの計測に十分な場合は、認定速度と認定バーストを `tokenmt` に定義するだけでも構いません。

必要に応じて、次の速度も定義できます。

- 認定バースト
- 最大バースト

`tokenmt` 速度の詳しい説明については、847 ページの「tokenmt をツーレートメーターとして構成する」を参照してください。`tokenmt(7ipp)` のマニュアルページでも詳しく説明しています。

`tswtclmt` モジュールを使用する場合は、次の速度を bps で定義する必要があります。

- 認定速度
- 最大速度

また、ウィンドウサイズをミリ秒単位で定義することもできます。これらの速度は、848 ページの「tswtclmt メータリングモジュール」および`tswtclmt(7ipp)` のマニュアルページで定義されています。

7 計測するトラフィックのトラフィック適合結果 (outcome) を追加します。

どちらのメータリングモジュールでも、結果は緑、赤、黄の 3 つです。定義した速度に当てはまるトラフィック適合結果を QoS 構成表に追加します。メーターの結果については、846 ページの「メーターモジュール」で詳しく説明されています。

認定速度に適合したトラフィックまたは適合しなかったトラフィックに対して取るべきアクションを決める必要があります。常にではありませんが多くの場合、このアクションは、ホップ単位の動作でパケットヘッダーをマークすることです。緑レベルのトラフィックに対して取りうるアクションの 1 つとして、トラフィックフローが認定速度を超えないかぎり処理を続行することもあります。あるいは、フローが最大速度を超えた場合にそのクラスのパケットをドロップすることもできます。

例 30–2 メーターの定義

次の表は、電子メールトラフィックのクラスに対するメーターの記入例を示す例です。IPQoS システムを持つネットワークの総帯域幅は 100 Mbits/秒、つまり毎秒 10000000 ビットです。QoS ポリシーは、電子メールクラスに低い優先順位を割り当てます。このクラスには、ベストエフォート型の転送動作も割り当てられます。

クラス	優先順位	フィルタ	セレクタ	レート
email	8	mail_in	daddr10.50.50.5 dport imap direction LOCAL_IN	
email	8	mail_out	saddr10.50.50.5 sport imap direction LOCAL_OUT	メーター =tokenmt 認定速度 =5000000 認定バースト =5000000 最大速度 =10000000 最大バースト =1000000 緑の優先度 = 処理続行 黄の優先度 = 黄の PHB を付加 赤の優先度 = ドロップ

参照
- パケットがネットワークストリームに戻るときのフローの転送動作を定義するには、795 ページの「転送動作を計画する方法」を参照してください。
- 特定の種類のトラフィックのフローアカウンティングを計画するには、797 ページの「フローアカウンティングを計画する方法」を参照してください。

- QoS ポリシーにさらにクラスを追加するには、788 ページの「QoS ポリシーのクラスを定義する方法」を参照してください。
- QoS ポリシーにさらにフィルタを追加するには、791 ページの「QoS ポリシーにフィルタを定義する方法」を参照してください。
- 別のフロー制御スキームを定義するには、792 ページの「フロー制御を計画する方法」を参照してください。
- IPQoS 構成ファイルを作成するには、806 ページの「IPQoS 構成ファイルを作成し、トラフィッククラスを定義する方法」を参照してください。

▼ 転送動作を計画する方法

転送動作によって、ネットワークに転送されるトラフィックフローの優先度とドロップ優先順位が決まります。選択できる主な転送動作は 2 つあります。1 つはほかのトラフィッククラスとの関連でクラスのフローに優先順位を付けることで、もう 1 つはフローを完全にドロップすることです。

Diffserv モデルでは、マーカーを使用して選択した転送動作をトラフィックフローに割り当てます。IPQoS には、次のマーカーモジュールが用意されています。

- `dscpmk` – IP パケットの DS フィールドに DSCP を付けるために使用する
- `dlcosmk` – データグラムの VLAN タグにサービスクラス (CoS) 値を付けるために使用する

注 – このセクションでは、IP パケットに限定して説明します。IPQoS システムに VLAN デバイスが含まれている場合は、`dlcosmk` マーカーを使って、データグラムの転送動作をマークできます。詳細は、851 ページの「VLAN デバイスでの `dlcosmk` マーカーの使用」を参照してください。

IP トラフィックに優先順位を付けるには、各パケットに DSCP を割り当てる必要があります。`dscpmk` マーカーは、パケットの DS フィールドに DSCP のマークを設定します。クラスの DSCP は、転送動作の種類に対応付けられている既知のコードポイントのグループから選択します。既知のコードポイントには、EF PHB 用の 46 (101110) や AF PHB 用のいくつかのコードポイントがあります。DSCP と転送の概要情報については、776 ページの「IPQoS 対応ネットワークでのトラフィック転送」を参照してください。

始める前に　次の手順の前に、QoS ポリシーのクラスとフィルタを定義してあるものとします。トラフィックを制御する場合は、メーターをマーカーと組み合わせて使用しますが、転送動作を定義するだけであれば、マーカーを単独で使用できます。

1. これまでに作成したクラスとそれらに割り当てた優先順位を確認します。
   すべてのトラフィッククラスをマークする必要があるとは限りません。

2. もっとも高い優先順位のクラスに **EF PHB** を割り当てます。

   EF PHB は、EF DSCP 46 (101110) を持つパケットが、AF PHB を割り当てたパケットよりも先にネットワーク上に送出されることを保証します。このため、もっとも高い優先順位のトラフィックには EF PHB を使用します。EF については、850 ページの「完全優先転送 (Expedited Forwarding、EF) PHB」を参照してください。

3. トラフィックを計測するクラスに転送動作を割り当てます。

4. 残りのクラスに、すでに割り当てた優先順位に応じた **DS** コードポイントを割り当てます。

例 30–3 ゲームアプリケーション向け QoS ポリシー

トラフィックの計測は、一般に次の理由で行います。

- SLA は、ネットワークの使用率が高いときでも、このクラスのパケットにある程度のサービスを保証する
- 低い優先順位のクラスがネットワークをあふれさせる傾向がある

マーカーをメーターと組み合わせて使用すると、これらのクラスに対して差別化サービスを提供したり帯域幅の管理を行ったりできます。たとえば、次の表は QoS ポリシーの一部を示しています。このポリシーは、高いトラフィックレベルを生成する人気のあるゲームアプリケーション向けのクラスを定義します。

クラス	優先順位	フィルタ	セレクタ	レート	転送動作をどうするか
`games_app`	9	`games_in`	`sport 6080`	なし	なし
`games_app`	9	`games_out`	`dport 6081`	メーター =`tokenmt`	緑 =AF31
				認定速度 =5000000	黄 =AF42
				認定バースト =5000000	赤 = ドロップ
				最大速度 =10000000	
				最大バースト =15000000	
				緑の優先度 = 処理続行	
				黄の優先度 = 黄の PHB を付加	
				赤の優先度 = ドロップ	

これらの転送動作では、認定速度に適合しているか、最大速度を下回っている games_app トラフィックには、低い優先順位の DSCP を割り当てます。games_app トラフィックが最大速度を上回ると、QoS ポリシーは games_app のパケットをドロップするよう指示します。すべての AF コードポイントについては、表 34–2 を参照してください。

参照
- 特定の種類のトラフィックのフローアカウンティングを計画するには、797 ページの「フローアカウンティングを計画する方法」を参照してください。
- QoS ポリシーにさらにクラスを追加するには、788 ページの「QoS ポリシーのクラスを定義する方法」を参照してください。
- QoS ポリシーにさらにフィルタを追加するには、791 ページの「QoS ポリシーにフィルタを定義する方法」を参照してください。
- フロー制御スキームを定義するには、792 ページの「フロー制御を計画する方法」を参照してください。
- パケットがネットワークストリームに戻るときのフローの転送動作をさらに定義するには、795 ページの「転送動作を計画する方法」を参照してください。
- IPQoS 構成ファイルを作成するには、806 ページの「IPQoS 構成ファイルを作成し、トラフィッククラスを定義する方法」を参照してください。

▼ フローアカウンティングを計画する方法

IPQoS flowacct モジュールを使用して、請求またはネットワーク管理のためにトラフィックフローを追跡します。次の手順に従って、QoS ポリシーにフローアカウンティングを含める必要があるかどうか判断してください。

1 **顧客に SLA を提示していますか**

提示している場合は、フローアカウンティングを使用する必要があります。まず、SLA を確認して、企業が顧客に使用料を請求するネットワークトラフィックの種類を調べます。次に、QoS ポリシーを確認して、課金の対象となるトラフィックが含まれるクラスを調べます。

2 **ネットワークの問題を防ぐために監視またはテストを必要とするアプリケーションはありますか**

ある場合は、フローアカウンティングを使ってそれらのアプリケーションの動作を監視することを検討します。QoS ポリシーを確認して、監視を必要とするトラフィックに割り当てたクラスを調べます。

3 **QoS 計画表で、フローアカウンティングを必要とする各クラスのフローアカウンティング欄に Y と記入します。**

参照
- QoS ポリシーにさらにクラスを追加するには、788 ページの「QoS ポリシーのクラスを定義する方法」を参照してください。
- QoS ポリシーにさらにフィルタを追加するには、791 ページの「QoS ポリシーにフィルタを定義する方法」を参照してください。
- フロー制御スキームを定義するには、792 ページの「フロー制御を計画する方法」を参照してください。
- パケットがネットワークストリームに戻るときのフローの転送動作を定義するには、795 ページの「転送動作を計画する方法」を参照してください。
- 特定の種類のトラフィックの追加フローのアカウンティングを計画するには、797 ページの「フローアカウンティングを計画する方法」を参照してください。
- IPQoS 構成ファイルを作成するには、806 ページの「IPQoS 構成ファイルを作成し、トラフィッククラスを定義する方法」を参照してください。

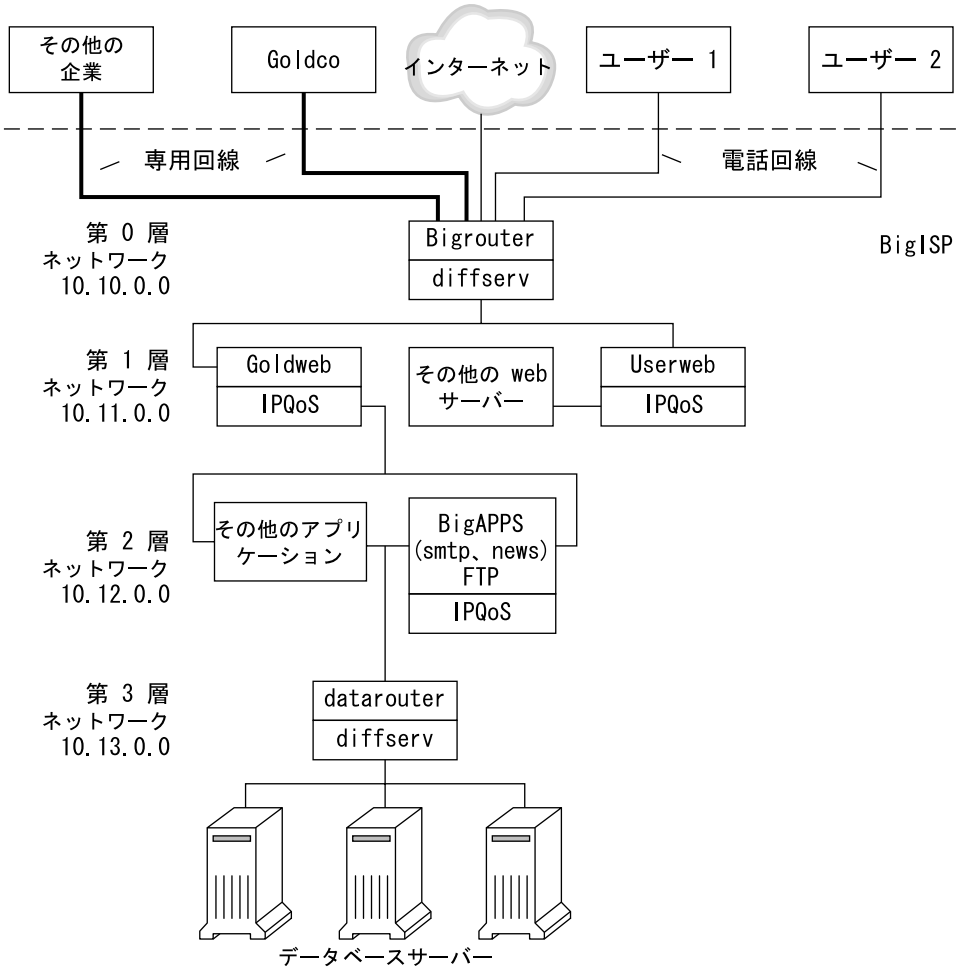
IPQoS の構成例の紹介

このドキュメントの残りの章で説明するタスクでは、このセクションで紹介する IPQoS の構成例を使用します。この例は、架空のサービスプロバイダである BigISP の公共イントラネットでの差別化サービスソリューションを示しています。BigISP は、専用回線によって BigISP にアクセスする大企業向けにサービスを提供しています。モデムによるダイアルインを行う個人顧客も BigISP からサービスを購入しています。

IPQoS トポロジ

次の図は、BigISP の公共イントラネットで使用するネットワークトポロジを示しています。

図 30–4 IPQoS のトポロジの例



BigISP は、公共イントラネットにこれらの 4 つの層を実装しています。

- 第 **0** 層 – ネットワーク `10.10.0.0` には、`Bigrouter` という大規模な Diffserv ルーターがあり、外部インタフェースと内部インタフェースの両方を備えています。Goldco 社という大企業をはじめとする数社の企業が `Bigrouter` で終端する専用回線サービスを借りています。第 0 層では、電話回線または ISDN を介して接続する個人客も管理しています。
- 第 **1** 層 – ネットワーク `10.11.0.0` では、Web サービスを提供しています。`Goldweb` サーバーは、Goldco 社が BigISP から購入したプレミアムサービスの一部である Web サイトのホストとして動作します。`Userweb` サーバーは、個人客が購入した小規模の Web サイトのホストして動作します。`Goldweb` と `Userweb` のどちらのサーバーも IPQoS に対応しています。
- 第 **2** 層 – ネットワーク `10.12.0.0` では、すべての顧客が使用するアプリケーションを提供します。アプリケーションサーバーの 1 つである `BigAPPS` は IPQoS 対応サーバーです。`BigAPPS` は、SMTP、ニュース、および FTP サービスを提供します。
- 第 **3** 層 – ネットワーク `10.13.0.0` には、大規模データベースサーバーがいくつか格納されています。第 3 層へのアクセスは `datarouter` という Diffserv ルーターによって制御されます。

◆ ◆ ◆ 第 31 章

IPQoS 構成ファイルの作成 (手順)

この章では、IPQoS 構成ファイルの作成方法について説明します。この章では、次の内容について説明します。

- 801 ページの「IPQoS 構成ファイル内での QoS ポリシーの定義 (タスクマップ)」
- 803 ページの「QoS ポリシー作成用のツール」
- 804 ページの「Web サーバー用 IPQoS 構成ファイルの作成」
- 817 ページの「アプリケーションサーバー用 IPQoS 構成ファイルの作成」
- 827 ページの「ルーター上での差別化サービスの提供」

この章の説明は、完全な QoS ポリシーを定義していること、このポリシーを IPQoS 構成ファイルのベースとしてすぐに使用できることを前提としています。QoS ポリシーを計画するには、785 ページの「サービス品質ポリシーの計画」を参照してください。

IPQoS 構成ファイル内での QoS ポリシーの定義 (タスクマップ)

このタスクマップでは、IPQoS 構成ファイルを作成するための一般的なタスクの一覧と、各タスクの実行手順を説明したセクションへのリンクを示します。

タスク	説明	手順
1. IPQoS 対応のネットワーク構成を計画する	ローカルネットワーク上でどのシステムを IPQoS 対応にするかを決定する	787 ページの「IPQoS 用のネットワークを準備する方法」
2. ネットワーク上の IPQoS システム用 QoS ポリシーを計画する	トラフィックフローを区別できるサービスクラスとして識別する。次に、トラフィック管理が必要なフローを決定する	785 ページの「サービス品質ポリシーの計画」

タスク	説明	手順
3. IPQoS 構成ファイルを作成し、その最初のアクションを定義する	IPQoS ファイルを作成し、IP クラシファイアを呼び出し、処理を実行させるクラスを定義する	806 ページの「IPQoS 構成ファイルを作成し、トラフィッククラスを定義する方法」
4. クラス用のフィルタを作成する	トラフィックの選択とクラス分けとを規定するフィルタを追加する	808 ページの「IPQoS 構成ファイル内でフィルタを定義する方法」
5. IPQoS 構成ファイルにクラスおよびフィルタをさらに追加する	IP クラシファイアに処理させるクラスおよびフィルタをさらに作成する	814 ページの「ベストエフォート Web サーバー用の IPQoS 構成ファイルを作成する方法」
6. メータリングモジュールを構成するパラメータを含む `action` 文を追加する	QoS ポリシーがフロー制御を必要とする場合、フロー制御速度および適合レベルをメーターに割り当てる	823 ページの「IPQoS 構成ファイル内でフロー制御を構成する方法」
7. マーカーを構成するパラメータを含む `action` 文を追加する	QoS ポリシーが差別化転送動作を必要とする場合、トラフィッククラスの転送方法を定義する	809 ページの「IPQoS 構成ファイル内でトラフィック転送を定義する方法」
8. フローアカウンティングモジュールを構成するパラメータを含む `action` 文を追加する	QoS ポリシーがトラフィックフローに関する統計の取得を必要とする場合、これらのアカウンティング統計の収集方法を定義する	812 ページの「IPQoS 構成ファイル内でクラスのアカウンティングを有効にする方法」
9. IPQoS 構成ファイルを適用する	作成した IPQoS 構成ファイルの内容を、適切なカーネルモジュールに追加する	830 ページの「新規構成を IPQoS カーネルモジュールへ適用する方法」
10. 転送動作をルーターファイル内で構成する	ネットワーク上のいずれかの IPQoS 構成ファイルで転送動作が定義されている場合、結果として得られる DSCP を、ルーターの適切なスケジューリングファイルに追加する	827 ページの「IPQoS 対応ネットワーク上でルーターを構成する方法」

QoS ポリシー作成用のツール

ネットワーク用の QoS ポリシーは、IPQoS 構成ファイル内に格納します。テキストエディタでこの構成ファイルを作成します。次に、作成したファイルを IPQoS 構成ユーティリティー `ipqosconf` の引数とします。`ipqosconf` に対し、構成ファイル内で定義されたポリシーの適用を指示すると、ポリシーがカーネル IPQoS システムに書き込まれます。`ipqosconf` コマンドの詳細については、`ipqosconf(1M)` のマニュアルページを参照してください。`ipqosconf` の使用方法については、830 ページの「新規構成を IPQoS カーネルモジュールへ適用する方法」を参照してください。

基本 IPQoS 構成ファイル

IPQoS 構成ファイルは、`Planning the Quality-of-Service Policy`で定義した QoS ポリシーを実行する785 ページの「サービス品質ポリシーの計画」 文のツリーで構成されています。IPQoS 構成ファイルは、IPQoS モジュールの構成を行います。各アクション文には、アクション文の中で呼び出されるモジュールが処理する「クラス」、「フィルタ」、または「パラメータ」のセットが含まれます。

IPQoS 構成ファイルの完全な構文については、例 34–3 および `ipqosconf(1M)` のマニュアルページを参照してください。

IPQoS トポロジ例の構成

この章では、3 つの IPQoS 対応システム用の IPQoS 構成ファイルを作成する方法を示します。これらのシステムは、図 30–4 で紹介した BigISP 社のネットワークトポロジの一部です。

- `Goldweb` – プレミアムレベル SLA を購入した顧客用 Web サイトのホストとして機能する Web サーバー
- `Userweb` – ベストエフォート SLA を購入したホームユーザー向けの個人用 Web サイトのホストとして機能する、やや能力の劣る Web サーバー
- `BigAPPS` – ゴールドレベルとベストエフォートの両方の顧客に、メール、ネットワークニュース、および FTP を提供するアプリケーションサーバー

3 つの構成ファイルを通して、最も一般的な IPQoS 構成を示します。IPQoS を実装するために、次のセクションのサンプルファイルをテンプレートとして使用することもできます。

Web サーバー用 IPQoS 構成ファイルの作成

このセクションでは、まず、プレミアム Web サーバー用の構成を通して、IPQoS 構成ファイルを紹介します。次に、個人用 Web サイトのホストとして機能するサーバー用の構成ファイルで、まったく異なるサービスレベルを構成する方法を示します。両方のサーバーは、図 30–4 のネットワーク例の一部です。

次の構成ファイルは、Goldweb サーバーの IPQoS アクティビティーを定義します。このサーバーは、プレミアム SLA を購入した Goldco 社の Web サイトのホストです。

例 31–1 プレミアム Web サーバー用 IPQoS 構成ファイルの例

```
fmt_version 1.0

action {
    module ipgpc
    name ipgpc.classify
    params {
        global_stats TRUE
    }
    class {
        name goldweb
        next_action markAF11
        enable_stats FALSE
    }
    class {
        name video
        next_action markEF
        enable_stats FALSE
    }
    filter {
        name webout
        sport 80
        direction LOCAL_OUT
        class goldweb
    }
    filter {
        name videoout
        sport videosrv
        direction LOCAL_OUT
        class video
    }
}
action {
    module dscpmk
    name markAF11
    params {
        global_stats FALSE
        dscp_map{0-63:10}
        next_action continue
    }
}
action {
    module dscpmk
    name markEF
    params {
```

例 31–1 プレミアム Web サーバー用 IPQoS 構成ファイルの例 (続き)

```
        global_stats TRUE
        dscp_map{0-63:46}
        next_action acct
    }
}
action {
    module flowacct
    name acct
    params {
        enable_stats TRUE
        timer 10000
        timeout 10000
        max_limit 2048
    }
}
```

次の構成ファイルは、Userweb の IPQoS アクティビティーを定義します。このサーバーは、低価格または「ベストエフォート型」の SLA を購入した個人の Web サイトのホストです。このサービスレベルでは、IPQoS システムがより高額の SLA を利用する顧客からのトラフィックを処理したあとに、できるかぎり最良のサービスをベストエフォートの顧客に保証します。

例 31–2 ベストエフォート Web サーバー用の構成例

```
fmt_version 1.0

action {
    module ipgpc
    name ipgpc.classify
    params {
        global_stats TRUE
    }
    class {
        name Userweb
        next_action markAF12
        enable_stats FALSE
    }
    filter {
        name webout
        sport 80
        direction LOCAL_OUT
        class Userweb
   }
}
action {
    module dscpmk
    name markAF12
    params {
        global_stats FALSE
        dscp_map{0-63:12}
        next_action continue
    }
}
```

▼ IPQoS 構成ファイルを作成し、トラフィッククラスを定義する方法

最初の IPQoS 構成ファイルは、メンテナンスしやすい任意のディレクトリに作成できます。この章のタスクでは、IPQoS 構成ファイルの位置としてディレクトリ `/var/ipqos` を使用します。次の手順では、例 31–1 で紹介された IPQoS 構成ファイルの最初のセグメントを構築します。

注 – IPQoS 構成ファイルを作成する際、各 `action` 文および句を必ず中括弧 ({}) で囲んでください。中括弧の使用例については、例 31–1 を参照してください。

---

1 プレミアム **Web** サーバーにログインし、新規 **IPQoS** 構成ファイルを拡張子 **.qos** を付けて作成します。

すべての IPQoS 構成ファイルで、最初の非コメント行にバージョン番号 `fmt_version 1.0` を記述する必要があります。

2 冒頭のパラメータに続き、初期 **action** 文を記述して汎用の **IP** クラシファイア **ipgpc** を構成します。

IPQoS 構成ファイルを成す `action` 文のツリーは、この初期アクションから始まります。たとえば、`/var/ipqos/Goldweb.qos` ファイルは、 `ipgpc` クラシファイアを呼び出す初期 `action` 文で始まります。

```
fmt_version 1.0

action {
    module ipgpc
    name ipgpc.classify
```

`fmt_version 1.0`	IPQoS 構成ファイルを開始する
`action {`	action 文を開始する
`module ipgpc`	構成ファイル内の最初のアクションとして `ipgpc` クラシファイアを構成する
`name ipgpc.classify`	クラシファイアの `action` 文の名前を定義する。名前は常に `ipgpc.classify` でなければならない

`action` 文の詳しい構文については、857 ページの「`action` 文」および `ipqosconf(1M)` のマニュアルページを参照してください。

3 統計パラメータ **global_stats** を含む **params** 句を追加します。

```
params {
        global_stats TRUE
   }
```

ipgpc.classify 文のパラメータ global_stats TRUE は、そのアクションに関する統計の収集を可能にします。また、global_stats TRUE を使用し、かつクラス句定義に enable_stats TRUE を指定すれば、そのクラスの統計の収集が可能になります。

統計の取得を有効にすると、パフォーマンスが影響を受けます。新規 IPQoS 構成ファイルを作成したときには、IPQoS が適正に動作するか検証するために、統計収集を有効にしてもかまいません。あとで global_stats の引数を FALSE に変更すれば、統計取得を無効にできます。

グローバル統計は、params 句で定義可能なパラメータの 1 種類に過ぎません。params 句の構文などについては、859 ページの「params 句」および ipqosconf(1M) のマニュアルページを参照してください。

4 プレミアムサーバーに向かうトラフィックを特定するクラスを定義します。

```
class {
        name goldweb
        next_action markAF11
        enable_stats FALSE
    }
```

この文は、「*class* 句」と呼ばれます。class 句には次の内容が含まれます。

name goldweb	goldweb クラスを作成して、Goldweb サーバーに向かうトラフィックを特定する
next_action markAF11	ipgpc モジュールに対し、goldweb クラスのパケットをアクション文 markAF11 に渡すよう指示する。アクション文 markAF11 は、 dscpmk マーカーを呼び出す
enable_stats FALSE	goldweb クラスの統計取得を可能にする。ただし、enable_stats の値が FALSE であるため、このクラスの統計取得は有効にはならない

class 句の構文の詳細については、859 ページの「class 句」および ipqosconf(1M) のマニュアルページを参照してください。

5 もっとも高い優先順位の転送を必要とするアプリケーションを特定するクラスを定義します。

```
class {
        name video
        next_action markEF
        enable_stats FALSE
    }
```

name video	video クラスを作成して、Goldweb サーバーから発信されるストリーミングビデオのトラフィックを特定する
next_action markEF	ipgpc モジュールに対し、ipgpc による処理が完了した video クラスのパケットを、markEF 文に渡すよう指示する。markEF 文は、dscpmk マーカーを呼び出す

`enable_stats FALSE`	`video` クラスの統計収集を可能にする。ただし、`enable_stats` の値が `FALSE` であるため、このクラスの統計収集は有効にはならない

参照
- 作成したばかりのクラスにフィルタを定義するには、808 ページの「IPQoS 構成ファイル内でフィルタを定義する方法」を参照してください。
- 構成ファイルに対して別の class 句を作成するには、806 ページの「IPQoS 構成ファイルを作成し、トラフィッククラスを定義する方法」を参照してください。

▼ IPQoS 構成ファイル内でフィルタを定義する方法

次の手順では、IPQoS 構成ファイル内でクラスのフィルタを定義します。

始める前に　次の手順の前に、構成ファイルの作成を開始しており、クラスを定義してあるものとします。この手順は、806 ページの「IPQoS 構成ファイルを作成し、トラフィッククラスを定義する方法」で作成された `/var/ipqos/Goldweb.qos` ファイルを引き続き構築します。

注 – IPQoS 構成ファイルを作成する際、各 `class` 句および `filter` 句を必ず中括弧 ({}) で囲んでください。中括弧の使用例については、例 31–1 を参照してください。

---

1 **IPQoS 構成ファイルを開き、最後に定義したクラスの末尾を探します。**

たとえば、IPQoS 対応サーバー `Goldweb` 用の構成ファイル `/var/ipqos/Goldweb.qos` では、次の `class` 句のあとから作業を始めます。

```
class {
        name video
        next_action markEF
        enable_stats FALSE
    }
```

2 **`filter` 句を定義し、IPQoS システムからの出力トラフィックを選択します。**

```
    filter {
        name webout
        sport 80
        direction LOCAL_OUT
        class goldweb
    }
```

`name webout`	フィルタに `webout` という名前を付ける
`sport 80`	ソースポート 80 のトラフィックを選択する。これは、既知の HTTP (Web) トラフィック用ポート
`direction LOCAL_OUT`	ローカルシステムから発信されるトラフィックを選択する

`class goldweb`	フィルタが所属するクラス (このインスタンスでは `goldweb` クラス) を特定する

IPQoS 構成ファイル内の `filter` 句の構文などについては、859 ページの「`filter` 句」を参照してください。

3 **IPQoS システムのストリーミングビデオトラフィックを選択する `filter` 句を定義します。**

```
filter {
    name videoout
    sport videosrv
    direction LOCAL_OUT
    class video
}
```

name videoout	フィルタに `videoout` という名前を付ける
`sport videosrv`	ソースポート `videosrv` のトラフィックを選択する。これは、以前にこのシステムのストリーミングビデオアプリケーション用に定義したポート
`direction LOCAL_OUT`	ローカルシステムから発信されるトラフィックを選択する
`class video`	フィルタが所属するクラス (このインスタンスでは `video` クラス) を特定する

参照
- マーカーモジュールの転送動作を定義するには、809 ページの「IPQoS 構成ファイル内でトラフィック転送を定義する方法」を参照してください。
- メータリングモジュールのフロー制御パラメータを定義するには、823 ページの「IPQoS 構成ファイル内でフロー制御を構成する方法」を参照してください。
- IPQoS 構成ファイルをアクティブにするには、830 ページの「新規構成を IPQoS カーネルモジュールへ適用する方法」を参照してください。
- さらにフィルタを定義するには、808 ページの「IPQoS 構成ファイル内でフィルタを定義する方法」を参照してください。
- アプリケーションからのトラフィックフロー向けのクラスを作成するには、819 ページの「アプリケーションサーバー用 IPQoS 構成ファイルを作成する方法」を参照してください。

▼ IPQoS 構成ファイル内でトラフィック転送を定義する方法

次の手順では、IPQoS 構成ファイルにクラスのホップ単位の動作を追加して、トラフィック転送を定義します。

始める前に　次の手順の前に、既存の IPQoS 構成ファイルにクラスおよびフィルタを定義してあるものとします。この手順では、Example 31–1 の 例 31–1 ファイルを引き続き構築します。

注 – 次の手順では、dscpmk マーカーモジュールを使用してトラフィック転送を構成する方法を示します。dlclosmk マーカーを使用した VLAN システムのトラフィック転送については、851 ページの「VLAN デバイスでの dlcosmk マーカーの使用」を参照してください。

1　IPQoS 構成ファイルを開き、最後に定義したフィルタの末尾を探します。

たとえば、IPQoS 対応サーバー Goldweb 用の構成ファイル /var/ipqos/Goldweb.qos では、次の filter 句のあとから作業を始めます。

```
filter {
        name videoout
        sport videosrv
        direction LOCAL_OUT
        class video
    }
}
```

この filter 句は、ipgpc クラシファイアの action 文の最後に位置します。このため、フィルタを終了させる閉じ括弧のあとに、action 文を終了させる閉じ括弧が必要です。

2　次の action 文でマーカーを呼び出します。

```
action {
    module dscpmk
    name markAF11
```

module dscpmk　dscpmk マーカーモジュールを呼び出す

name markAF11　action 文に markAF11 という名前を付ける

以前に定義した goldweb クラスには next_action markAF11 という文が含まれています。この文は、クラシファイアによる処理が完了したトラフィックフローを、アクション文 markAF11 に送信します。

3　トラックフローに対してマーカーが取るアクションを定義します。

```
    params {
        global_stats FALSE
        dscp_map{0-63:10}
        next_action continue
    }
}
```

global_stats FALSE　マーカー action 文 markAF11 の統計収集を可能にする。ただし、global_stats の値が FALSE であるため、統計は収集されない

dscp_map{0–63:10}	DSCP 10 を、マーカーにより処理中の goldweb クラスのパケットヘッダーに割り当てる
next_action continue	userweb クラスのパケットに対しこれ以上処理を行う必要がないこと、およびこれらのパケットをネットワークストリームに戻してもよいことを示す

DSCP 10 は、マーカーに対し、dscp マップ内のすべてのエントリを 10 進数値の 10 (バイナリ値 001010) に設定するよう指示します。このコードポイントは、goldweb トラフィッククラスのパケットが AF11 ホップ単位動作 (PHB) に従うことを示します。AF11 は、DSCP 10 を持つすべてのパケットが、低ドロップ、および高い優先順位のサービスを受けることを保証します。このため、Goldweb 上のプレミアム顧客用の発信トラフィックには、AF (相対的優先転送) PHB で指定可能なもっとも高い優先順位が与えられます。AF に設定可能な DSCP の表については、表 34–2 を参照してください。

4 別のマーカー **action** 文を開始します。

```
action {
    module dscpmk
    name markEF
```

module dscpmk	dscpmk マーカーモジュールを呼び出す
name markEF	action 文に markEF という名前を付ける

5 トラフィックフローに対してマーカーが取るアクションを定義します。

```
    params {
        global_stats TRUE
        dscp_map{0-63:46}
        next_action acct
    }
}
```

global_stats TRUE	video クラスの統計収集を有効にする。このクラスはストリーミングビデオのパケットを選択する
dscp_map{0–63:46}	DSCP 46 を、マーカーにより処理中の video クラスのパケットヘッダーに割り当てる
next_action acct	dscpmk モジュールに対し、dscpmk による処理が完了した video クラスのパケットを、acct 文 action に渡すよう指示する。action 文 acct は flowacct モジュールを呼び出す

DSCP 46 は、dscpmk モジュールに対し、dscp マップのすべてのエントリを DS フィールドの 10 進数の 46 (バイナリ 101110) に設定するよう指示します。このコードポイントは、video トラフィッククラスのパケットが完全優先転送ホップ単位動作 (PHB) に従うことを示します。

注 – EF のコードポイントは 46 (バイナリ値 101110) にすることをお勧めします。その他の DSCP は、AF PHB をパケットに割り当てるときに使用します。

EF PHB は、DSCP 46 を持つパケットが IPQoS および Diffserv 対応システムにより もっとも高い優先度を与えられることを保証します。ストリーミングアプリ ケーションは、もっとも高い優先順位のサービスを必要とします。これが、QoS ポ リシーでこれらのアプリケーションに EF PHB を割り当てる理由です。PHB の完全優先転送の詳細については、850 ページの「完全優先転送 (Expedited Forwarding、EF) PHB」を参照してください。

6 作成したばかりの **DSCP** を **Diffserv** ルーターの適切なファイルに追加します。

詳細については、827 ページの「IPQoS 対応ネットワーク上でルーターを構成する方法」を参照してください。

参照
- トラフィックフローのフローアカウンティング統計の収集を開始するには、812 ページの「IPQoS 構成ファイル内でクラスのアカウンティングを有効にする方法」を参照してください。
- マーカーモジュールの転送動作を定義するには、809 ページの「IPQoS 構成ファイル内でトラフィック転送を定義する方法」を参照してください。
- メータリングモジュールのフロー制御パラメータを定義するには、823 ページの「IPQoS 構成ファイル内でフロー制御を構成する方法」を参照してください。
- IPQoS 構成ファイルをアクティブにするには、830 ページの「新規構成を IPQoS カーネルモジュールへ適用する方法」を参照してください。
- さらにフィルタを定義するには、808 ページの「IPQoS 構成ファイル内でフィルタを定義する方法」を参照してください。
- アプリケーションからのトラフィックフロー向けのクラスを作成するには、819 ページの「アプリケーションサーバー用 IPQoS 構成ファイルを作成する方法」を参照してください。

▼ IPQoS 構成ファイル内でクラスのアカウンティングを有効にする方法

次の手順では、IPQoS 構成ファイル内でトラフィッククラスのアカウンティングを有効にします。この手順では、806 ページの「IPQoS 構成ファイルを作成し、トラフィッククラスを定義する方法」で紹介した `video` クラスのフローアカウンティングを定義します。このクラスは、プレミアム SLA の一部として課金されるストリーミングビデオのトラフィックを選択します。

始める前に　次の手順の前に、既存の IPQoS 構成ファイルにクラス、フィルタ、メーターのアクション (必要な場合だけ)、およびマーカーのアクション (必要な場合だけ) を定義してあるものとします。この手順では、Example 31–1 の 例 31–1 ファイルを引き続き構築します。

1 **IPQoS 構成ファイルを開き、最後に定義した action 文の末尾を探します。**

たとえば、IPQoS 対応サーバー Goldweb 用の構成ファイル /var/ipqos/Goldweb.qos. では、次の action 文 markEF のあとから作業を始めます。

```
action {
    module dscpmk
    name markEF
    params {
        global_stats TRUE
        dscp_map{0-63:46}
        next_action acct
    }
}
```

2 フローアカウンティングを呼び出す **action** 文を開始します。

```
action {
    module flowacct
    name acct
```

module flowacct	flowacct フローアカウンティングモジュールを呼び出す
name acct	action 文に acct という名前を付ける

3 トラフィッククラスに関するアカウンティングを制御する **params** 句を定義します。

```
params {
        global_stats TRUE
        timer 10000
        timeout 10000
        max_limit 2048
        next_action continue
    }
}
```

global_stats TRUE	video クラスの統計収集を有効にする。このクラスはストリーミングビデオのパケットを選択する
timer 10000	フローテーブル内で、タイムアウトしたフローが走査される間隔を、ミリ秒単位で指定する。このパラメータでは、間隔は 10000 ミリ秒
timeout 10000	最小の間隔タイムアウト値を指定する。フローのパケットがタイムアウト値で指定された時間検出されないと、フローは「タイムアウト」する。このパラメータでは、パケットは 10000 ミリ秒後にタイムアウトする

`max_limit 2048`	このアクションインスタンスのフローテーブル内でアクティブなフローレコードの最大数を設定する
`next_action continue`	`video` クラスのパケットに対しこれ以上処理を行う必要がないこと、およびこれらのパケットをネットワークストリームに戻してもよいことを示す

`flowacct` モジュールは、指定されたタイムアウト値に達するまで、特定のクラスのパケットフローに関する統計情報を収集します。

参照
- ルーターでホップ単位の動作を構成するには、827 ページの「IPQoS 対応ネットワーク上でルーターを構成する方法」を参照してください。
- IPQoS 構成ファイルをアクティブにするには、830 ページの「新規構成を IPQoS カーネルモジュールへ適用する方法」を参照してください。
- アプリケーションからのトラフィックフロー向けのクラスを作成するには、819 ページの「アプリケーションサーバー用 IPQoS 構成ファイルを作成する方法」を参照してください。

▼ ベストエフォート Web サーバー用の IPQoS 構成ファイルを作成する方法

ベストエフォート Web サーバー用の IPQoS 構成ファイルは、プレミアム Web サーバー用の IPQoS 構成ファイルとは少し違います。この手順では、例として例 31–2 の構成ファイルを使用します。

1. ベストエフォート **Web** サーバーにログインします。
2. 新規 **IPQoS** 構成ファイルを拡張子 **.qos** を付けて作成します。

```
fmt_vesion 1.0
action {
    module ipgpc
    name ipgpc.classify
    params {
        global_stats TRUE
   }
```

`/var/ipqos/userweb.qos` ファイルは、`ipgpc` クラシファイアを呼び出す部分 `action` 文から始める必要があります。この `action` 文には、統計収集を有効にする params 句も含めています。この `action` 文については、806 ページの「IPQoS 構成ファイルを作成し、トラフィッククラスを定義する方法」を参照してください。

3 ベストエフォート **Web** サーバーに向かうトラフィックを特定するクラスを定義します。

```
class {
        name userweb
        next_action markAF12
        enable_stats FALSE
    }
```

`name userweb`	`userweb` クラスを作成して、ユーザーから Userweb サーバーに向かうトラフィックを特定する
`next_action markAF1`	`ipgpc` モジュールに対し、`ipgpc` による処理が完了した `userweb` クラスのパケットを、`action` 文 `markAF12` に渡すよう指示する。`action` 文 `markAF12` は、`dscpmk` マーカーを呼び出す
`enable_stats FALSE`	`userweb` クラスの統計収集を可能にする。ただし、`enable_stats` の値が `FALSE` であるため、このクラスの統計は収集されない

`class` 句の処理については、806 ページの「IPQoS 構成ファイルを作成し、トラフィッククラスを定義する方法」を参照してください。

4 **userweb** クラスのトラフィックフローを選択する **filter** 句を定義します。

```
    filter {
       name webout
       sport 80
       direction LOCAL_OUT
       class userweb
   }
}
```

`name webout`	フィルタに `webout` という名前を付ける
`sport 80`	ソースポート 80 のトラフィックを選択する。これは、既知の HTTP (Web) トラフィック用ポート
`direction LOCAL_OUT`	ローカルシステムから発信されるトラフィックを選択する
class userweb	フィルタが所属するクラス (このインスタンスでは `userweb` クラス) を特定する

`filter` 句の処理については、808 ページの「IPQoS 構成ファイル内でフィルタを定義する方法」を参照してください。

5 **dscpmk** マーカーを呼び出す **action** 文を開始します。

```
action {
    module dscpmk
    name markAF12
```

`module dscpmk`	`dscpmk` マーカーモジュールを呼び出す

`name markAF12` `action` 文に `markAF12` という名前を付ける

以前に定義した `userweb` クラスには `next_action markAF12` という文が含まれています。この文は、クラシファイアによる処理が完了したトラフィックフローを、`action` 文 `markAF12` に送信します。

6 トラフィックフローの処理に使用する、マーカーのパラメータを定義します。

```
    params {
        global_stats FALSE
        dscp_map{0-63:12}
        next_action continue
    }
}
```

`global_stats FALSE`	マーカー `action` 文 `markAF12` の統計収集を可能にする。ただし、`enable_stats` の値が `FALSE` であるため、統計は収集されない
`dscp_map{0–63:12}`	DSCP 12 を、マーカーにより処理中の `userweb` クラスのパケットヘッダーに割り当てる
`next_action continue`	`userweb` クラスのパケットに対しこれ以上処理を行う必要がないこと、およびこれらのパケットをネットワークストリームに戻してもよいことを示す

DSCP 12 は、マーカーに対し、`dscp` マップ内のすべてのエントリを 10 進数値の 12 (バイナリ値 001100) に設定するよう指示します。このコードポイントは、`userweb` トラフィッククラスのパケットが AF12 ホップ単位動作 (PHB) に従うことを示します。AF12 は、DS フィールド内に DSCP 12 を持つすべてのパケットが、中程度のドロップ、および高い優先順位のサービスを受けることを保証します。

7 **IPQoS 構成ファイルを完成したら、構成を適用します。**

参照
- アプリケーションからのトラフィックフローに対して、クラスやほかの構成を追加するには、819 ページの「アプリケーションサーバー用 IPQoS 構成ファイルを作成する方法」を参照してください。
- ルーターでホップ単位の動作を構成するには、827 ページの「IPQoS 対応ネットワーク上でルーターを構成する方法」を参照してください。
- IPQoS 構成ファイルをアクティブにするには、830 ページの「新規構成を IPQoS カーネルモジュールへ適用する方法」を参照してください。

アプリケーションサーバー用 IPQoS 構成ファイルの作成

このセクションでは、顧客に主要アプリケーションを提供するアプリケーションサーバー用の、構成ファイルを作成する方法について説明します。この手順では、例として図 30–4 の BigAPPS サーバーを使用します。

次の構成ファイルは、BigAPPS サーバーの IPQoS アクティビティーを定義します。このサーバーは、顧客向けの FTP、電子メール (SMTP)、およびネットワークニュース (NNTP) のホストです。

例 31–3　アプリケーションサーバー用サンプル IPQoS 構成ファイル

```
fmt_version 1.0

action {
    module ipgpc
    name ipgpc.classify
    params {
        global_stats TRUE
    }
    class {
        name smtp
        enable_stats FALSE
        next_action markAF13
    }
    class {
        name news
        next_action markAF21
    }
    class {
        name ftp
        next_action meterftp
    }
    filter {
        name smtpout
        sport smtp
        class smtp
    }
    filter {
        name newsout
        sport nntp
        class news
    }
    filter {
        name ftpout
        sport ftp
        class ftp
    }
   filter {
        name ftpdata
        sport ftp-data
        class ftp
    }
}
action {
```

例 31–3 アプリケーションサーバー用サンプル IPQoS 構成ファイル (続き)

```
    module dscpmk
    name markAF13
    params {
        global_stats FALSE
        dscp_map{0-63:14}
        next_action continue
    }
}
action {
    module dscpmk
    name markAF21
    params {
        global_stats FALSE
        dscp_map{0-63:18}
        next_action continue
    }
}
action {
    module tokenmt
    name meterftp
    params {
        committed_rate 50000000
        committed_burst 50000000
        red_action_name AF31
        green_action_name markAF22
        global_stats TRUE
    }
}
action {
    module dscpmk
    name markAF31
    params {
        global_stats TRUE
        dscp_map{0-63:26}
        next_action continue
    }
}
action {
    module dscpmk
    name markAF22
    params {
        global_stats TRUE
        dscp_map{0-63:20}
        next_action continue
    }
}
```

▼ アプリケーションサーバー用 IPQoS 構成ファイルを作成する方法

1 **IPQoS 対応アプリケーションサーバーにログインし、新規 IPQoS 構成ファイルを拡張子 .qos を付けて作成します。**

たとえば、アプリケーションサーバー用に /var/ipqos/BigAPPS.qos ファイルを作成します。action 文の最初に、ipgpc クラシファイアを呼び出す次の記述を配置します。これらは必ず記述する必要があります。

```
fmt_version 1.0

action {
    module ipgpc
    name ipgpc.classify
    params {
        global_stats TRUE
    }
```

冒頭の action 文については、806 ページの「IPQoS 構成ファイルを作成し、トラフィッククラスを定義する方法」を参照してください。

2 **BigAPPS サーバー上の 3 つのアプリケーションからのトラフィックをそれぞれ選択するクラスを作成します。**

冒頭の action 文のあとに、クラス定義を追加します。

```
    class {
        name smtp
        enable_stats FALSE
        next_action markAF13
    }
    class {
        name news
        next_action markAF21
    }
    class {
        name ftp
        enable_stats TRUE
        next_action meterftp
    }
```

name smtp	smtp という名前のクラスを作成する。 このクラスには、SMTP アプリケーションが扱う電子メールのトラフィックフローが含まれる
enable_stats FALSE	smtp クラスの統計収集を可能にする。ただし、enable_stats の値が FALSE であるため、このクラスの統計は取得されない
next_action markAF13	ipgpc モジュールに対し、ipgpc による処理が完了した smtp クラスのパケットを、action 文 markAF13 に渡すよう指示する

`name news`	`news` という名前のクラスを作成する。 このクラスには、NNTP アプリケーションが扱うネットワークニュースのトラフィックフローが含まれる
`next_action markAF21`	`ipgpc` モジュールに対し、`ipgpc` による処理が完了した `news` クラスのパケットを、アクション文 `markAF21` に渡すよう指示する
`name ftp`	`ftp` という名前のクラスを作成する。 このクラスには、FTP アプリケーションが扱う発信トラフィックが含まれる
`enable_stats TRUE`	`ftp` クラスの統計収集を可能にする
`next_action meterftp`	`ipgpc` モジュールに対し、`ipgpc` による処理が完了した `ftp` クラスのパケットを、`action` 文 `meterftp` に渡すよう指示する

クラスの定義の詳細については、806 ページの「IPQoS 構成ファイルを作成し、トラフィッククラスを定義する方法」を参照してください。

3 手順 **2** で定義したクラスのトラフィックを選択する **filter** 句を定義します。

```
    filter {
        name smtpout
        sport smtp
        class smtp
    }
    filter {
        name newsout
        sport nntp
        class news
    }
        filter {
        name ftpout
        sport ftp
        class ftp
    }
        filter {
        name ftpdata
        sport ftp-data
        class ftp
    }
}
```

`name smtpout`	フィルタに `smtpout` という名前を付ける
`sport smtp`	ソースポート 25 のトラフィックを選択する。これは、既知の `sendmail` (SMTP) アプリケーション用ポート
`class smtp`	フィルタが所属するクラス (このインスタンスでは `smtp` クラス) を特定する
`name newsout`	フィルタに `newsout` という名前を付ける

`sport nntp`	ソースポート名 `nntp` のトラフィックを選択する。これは、既知のネットワークニュース (NNTP) アプリケーション用ポート
`class news`	フィルタが所属するクラス (このインスタンスでは `news` クラス) を特定する
`name ftpout`	フィルタに `ftpout` という名前を付ける
`sport ftp`	ソースポート 21 の制御データを選択する。これは、既知の FTP トラフィック用ポート番号
`name ftpdata`	フィルタに `ftpdata` という名前を付ける
`sport ftp-data`	ソースポート 20 のトラフィックを選択する。これは、既知の FTP データトラフィック用ポート番号
`class ftp`	`ftpout` および `ftpdata` フィルタが所属するクラス (このインスタンスでは `ftp`) を特定する

参照
- フィルタを定義するには、808 ページの「IPQoS 構成ファイル内でフィルタを定義する方法」を参照してください。
- アプリケーショントラフィックの転送動作を定義するには、821 ページの「IPQoS 構成ファイル内でアプリケーショントラフィックの転送を構成する方法」を参照してください。
- メータリングモジュールを使用してフロー制御を構成するには、823 ページの「IPQoS 構成ファイル内でフロー制御を構成する方法」を参照してください。
- フローアカウンティングを構成するには、812 ページの「IPQoS 構成ファイル内でクラスのアカウンティングを有効にする方法」を参照してください。

▼ IPQoS 構成ファイル内でアプリケーショントラフィックの転送を構成する方法

次の手順では、アプリケーショントラフィックの転送を構成します。次の手順では、アプリケーショントラフィッククラスのホップ単位動作を定義します。これらのクラスは、ネットワーク上のほかのトラフィックよりも優先度を低くする場合があります。この手順では、`Example 31–3` の 例 31–3 ファイルを引き続き構築します。

始める前に　この手順では、マークしたアプリケーションに対してクラスとフィルタをすでに定義した既存の IPQoS 構成ファイルがあることを前提にしています。

1 アプリケーションサーバー用に作成した **IPQoS** 構成ファイルを開き、最後の **filter** 句の末尾を検索します。

`/var/ipqos/BigAPPS.qos` ファイルでは、最後のフィルタは次のとおりです。

```
 filter {
        name ftpdata
        sport ftp-data
        class ftp
    }
}
```

2 次の方法でマーカーを呼び出します。

```
action {
    module dscpmk
    name markAF13
```

`module dscpmk` `dscpmk` マーカーモジュールを呼び出す

`name markAF13` `action` 文に `markAF13` という名前を付ける

3 電子メールのトラフィックフローにマークされるホップ単位動作を定義します。

```
    params {
        global_stats FALSE
        dscp_map{0-63:14}
        next_action continue
    }
}
```

`global_stats FALSE` マーカー `action` 文 `markAF13` の統計収集を可能にする。ただし、`global_stats` の値が `FALSE` であるため、統計は収集されない

`dscp_map{0–63:14}` DSCP 14 を、マーカーにより処理中の `smtp` クラスのパケットヘッダーに割り当てる

`next_action continue` `smtp` クラスのパケットに対しこれ以上処理を行う必要がないことを示す。よって、これらのパケットはネットワークストリームに戻すことができる

DSCP 14 は、マーカーに対し、`dscp` マップ内のすべてのエントリを 10 進数値の 14 (バイナリ値 001110) に設定するよう指示します。DSCP 14 は、AF13 のホップ単位の動作を設定します。マーカーは、DS フィールドの DSCP 14 で `smtp` トラフィッククラスのパケットをマークします。

AF13 は、DSCP 14 を持つすべてのパケットに高いドロップ優先度を割り当てます。しかし Class 1 の優先順位も保証するため、ルーターは電子メールの発信トラフィックに対し、キューの中で高い優先順位を与えます。設定可能な AF コードポイントの表については、表 34–2 を参照してください。

4 マーカー **action** 文を追加して、ネットワークニュースのトラフィック用のホップ単位動作を定義します。

```
action {
    module dscpmk
    name markAF21
    params {
        global_stats FALSE
        dscp_map{0-63:18}
        next_action continue
    }
}
```

`name markAF21`	`action` 文に `markAF21` という名前を付ける
`dscp_map{0–63:18}`	DSCP 18 を、マーカーにより処理中の `nntp` クラスのパケットヘッダーに割り当てる

DSCP 18 は、マーカーに対し、`dscp` マップ内のすべてのエントリを 10 進数値の 18 (バイナリ値 010010) に設定するよう指示します。DSCP 18 は、AF21 のホップ単位の動作を設定します。マーカーは、DS フィールドの DSCP 18 で `news` トラフィッククラスのパケットをマークします。

AF21 は DSCP 18 を持つすべてのパケットに低いドロップ優先度を保証しますが、優先順位は Class 2 にとどまります。よって、ネットワークニューストラフィックが振り落とされる可能性は低くなります。

参照
- Web サーバーの構成情報を追加するには、806 ページの「IPQoS 構成ファイルを作成し、トラフィッククラスを定義する方法」を参照してください。
- メータリングモジュールを使用してフロー制御を構成するには、823 ページの「IPQoS 構成ファイル内でフロー制御を構成する方法」を参照してください。
- フローアカウンティングを構成するには、812 ページの「IPQoS 構成ファイル内でクラスのアカウンティングを有効にする方法」を参照してください。
- ルーターで転送動作を構成するには、827 ページの「IPQoS 対応ネットワーク上でルーターを構成する方法」を参照してください。
- IPQoS 構成ファイルをアクティブにするには、830 ページの「新規構成を IPQoS カーネルモジュールへ適用する方法」を参照してください。

▼ IPQoS 構成ファイル内でフロー制御を構成する方法

ネットワークに送出される特定のトラフィックフローの速度を制御するには、メーターのパラメータを定義しなければなりません。IPQoS 構成ファイル内で、2 つのメーター `tokenmt` と `tswtclmt` とのどちらかを使用できます。

次の手順では、例 31–3 のアプリケーションサーバーの IPQoS 構成ファイルを引き続き構築します。次の手順では、メーターを構成するだけではなく、メーター action 文の内部で呼び出される 2 つの マーカーアクションも構成します。

始める前に　この手順の前に、フローを制御するアプリケーション用のクラスおよびフィルタを定義してあるものとします。

1 アプリケーションサーバー用に作成した **IPQoS** 構成ファイルを開きます。

/var/ipqos/BigAPPS.qos ファイルで、次のマーカーアクションのあとから作業を開始します。

```
action {
    module dscpmk
    name markAF21
    params {
        global_stats FALSE
        dscp_map{0-63:18}
        next_action continue
    }
}
```

2 **ftp** クラスのトラフィックをフロー制御するメーター **action** 文を作成します。

```
action {
    module tokenmt
    name meterftp
```

`module tokenmt`　tokenmt メーターを呼び出す

`name meterftp`　action 文に meterftp という名前を付ける

3 メーターの速度を構成するパラメータを追加します。

```
params {
       committed_rate 50000000
       committed_burst 50000000
```

`committed_rate 50000000`　ftp クラスのトラフィックに 50,000,000 bps の転送速度を割り当てる

`committed_burst 50000000`　ftp クラスのトラフィックに 50,000,000 ビットのバーストサイズを割り当てる

tokenmt パラメータについては、847 ページの「tokenmt をツーレートメーターとして構成する」を参照してください。

4 次のようにパラメータを追加して、トラフィック適合優先順位を構成します。

```
    red_action markAF31
    green_action_name markAF22
    global_stats TRUE
    }
}
```

`red_action_name markAF31`	`ftp` クラスのトラフィックフローが認定速度を超過した場合、パケットは、`markAF31` マーカー `action` 文に送信されることを示す
`green_action_name markAF22`	`ftp` クラスのトラフィックフローが認定速度に適合する場合、パケットがアクション文 `markAF22` に送られることを示す
`global_stats TRUE`	`ftp` クラスのメータリング統計取得を有効にする

トラフィックの適合性については、846 ページの「メーターモジュール」を参照してください。

5 ホップ単位動作を **`ftp`** クラスの不適合トラフィックフローに割り当てるマーカー **`action`** 文を追加します。

```
action {
    module dscpmk
    name markAF31
    params {
        global_stats TRUE
        dscp_map{0-63:26}
        next_action continue
    }
}
```

`module dscpmk`	`dscpmk` マーカーモジュールを呼び出す
`name markAF31`	`action` 文に `markAF31` という名前を付ける
`global_stats TRUE`	`ftp` クラスの統計取得を有効にする
`dscp_map{0–63:26}`	`ftp` クラスのトラフィックが認定速度を超過した場合は常に、DSCP 26 を ftp クラスのパケットヘッダーに割り当てる
`next_action continue`	`ftp` クラスのパケットに対しこれ以上処理を行う必要がないことを示す。よって、これらのパケットはネットワークストリームに戻すことができる

DSCP 26 は、マーカーに対し、`dscp` マップ内のすべてのエントリを 10 進数値の 26 (バイナリ値 011010) に設定するよう指示します。DSCP 26 は、AF31 のホップ単位の動作を設定します。マーカーは、DS フィールドの DSCP 26 で `ftp` トラフィッククラスのパケットをマークします。

AF31 は DSCP 26 を持つすべてのパケットに低いドロップ優先度を保証しますが、優先順位は Class 3 にとどまります。このため、速度不適合の FTP トラフィックがドロップされる可能性は低くなります。設定可能な AF コードポイントの表については、表 34–2 を参照してください。

6 認定速度に適合する **ftp** トラフィックフローにホップ単位動作を割り当てるマーカー **action** 文を追加します。

```
action {
    module dscpmk
    name markAF22
    params {
        global_stats TRUE
        dscp_map{0-63:20}
        next_action continue
    }
}
```

`name markAF22` — `marker` アクションに `markAF22` という名前を付ける

`dscp_map{0–63:20}` — `ftp` クラスのトラフィックが認定速度に適合する場合は常に、DSCP **20** をパケットヘッダーに割り当てる

DSCP 20 は、マーカーに対し、`dscp` マップ内のすべてのエントリを 10 進数値の 20 (バイナリ値 010100) に設定するよう指示します。DSCP **20** は、AF22 のホップ単位の動作を設定します。マーカーは、DS フィールドの DSCP **20** で `ftp` トラフィッククラスのパケットをマークします。

AF22 は、DSCP **20** を持つすべてのパケットに中程度のドロップ優先度と Class 2 の優先順位を保証します。このため、速度適合の FTP トラフィックは、IPQoS システムから同時に送出されるフロー内で中程度のドロップ優先度を保証されます。ただし、ルーターは、Class 1 で中程度のドロップ優先度以上を持つトラフィッククラスの転送を優先します。設定可能な AF コードポイントの表については、表 34–2 を参照してください。

7 アプリケーションサーバー用に作成した **DSCP** を、**Diffserv** ルーターの適切なファイルに追加します。

参照
- IPQoS 構成ファイルをアクティブにするには、830 ページの「新規構成を IPQoS カーネルモジュールへ適用する方法」を参照してください。
- Web サーバーの構成情報を追加するには、806 ページの「IPQoS 構成ファイルを作成し、トラフィッククラスを定義する方法」を参照してください。
- フローアカウンティングを構成するには、812 ページの「IPQoS 構成ファイル内でクラスのアカウンティングを有効にする方法」を参照してください。
- ルーターで転送動作を構成するには、827 ページの「IPQoS 対応ネットワーク上でルーターを構成する方法」を参照してください。

ルーター上での差別化サービスの提供

差別化サービスを提供するには、782 ページの「diffserv ネットワークのハードウェア計画」の説明に従って、ネットワークトポロジに Diffserv 対応ルーターを含める必要があります。ルーター上で Diffserv を構成し、ルーターのファイルを更新する実際の手順は、このガイドの扱う範囲ではありません。

このセクションでは、ネットワーク上のさまざまな IPQoS 対応システムおよび Diffserv ルーター間で、転送情報を調整する一般的な手順を説明します。

▼ IPQoS 対応ネットワーク上でルーターを構成する方法

次の手順では、例として図 30–4 のトポロジを使用します。

始める前に　次の手順の前に、この章のこれまでのタスクを実行することにより、ネットワーク上で IPQoS システムを構成してあるものとします。

1. **ネットワーク上のすべての IPQoS 対応システムの構成ファイルを確認します。**

2. **さまざまな QoS ポリシーで使用される各コードポイントを特定します。**

   コードポイント、およびコードポイントを適用するシステムとクラスの表を作成します。次の表から、同じコードポイントを使用した領域を知ることができます。同じコードポイントを使用したままでもかまいませんが、同じマークが付けられたクラス間の優先度を決めるには、IPQoS 構成ファイル内に `precedence` セレクタなどほかの条件を指定する必要があります。

   たとえば、この章の手順で使用するネットワーク例の場合、次のコードポイント表を作成できます。

システム	クラス	PHB	DS コードポイント
`Goldweb`	`video`	EF	46 (101110)
`Goldweb`	`goldweb`	AF11	10 (001010)
`Userweb`	`webout`	AF12	12 (001100)
`BigAPPS`	`smtp`	AF13	14 (001110)
`BigAPPS`	`news`	AF18	18 (010010)
`BigAPPS`	`ftp` 適合トラフィック	AF22	20 (010100)

システム	クラス	PHB	DS コードポイント
BigAPPS	ftp 不適合トラフィック	AF31	26 (011010)

3 ネットワークの **IPQoS** 構成ファイルから得たコードポイントを、**Diffserv** ルーターの適切なファイルに追加します。

これらのコードポイントは、ルーターの Diffserv スケジューリングメカニズムの構成に役立ちます。詳しくは、ルーターの製造元の文書および Web サイトを参照してください。

◆ ◆ ◆ 第 32 章

IPQoS の起動と保守(手順)

この章では、IPQoS 構成ファイルを有効化する方法および IPQoS 関連のイベントを記録する方法について説明します。次の項目について説明します。

- 829 ページの「IPQoS の管理 (タスクマップ)」
- 830 ページの「IPQoS 構成の適用」
- 831 ページの「IPQoS メッセージの syslog によるログ記録の有効化」
- 833 ページの「IPQoS のエラーメッセージの障害追跡」

IPQoS の管理 (タスクマップ)

このセクションには、Oracle Solaris システム上で IPQoS を起動および保守するための一連のタスクを示します。タスクを行う前に、801 ページの「IPQoS 構成ファイル内での QoS ポリシーの定義 (タスクマップ)」に従って、IPQoS 構成ファイルを完成しておく必要があります。

次の表では、それらのタスクについて箇条書き形式で説明し、各タスクを完了する方法が詳しく説明されたセクションへのリンクを示しています。

タスク	説明	手順
1. システムで IPQoS を構成します。	`ipqosconf` コマンドを使用して、システムの IPQoS 構成ファイルを有効にします。	830 ページの「新規構成を IPQoS カーネルモジュールへ適用する方法」
2. Oracle Solaris 起動スクリプトを使用して、各システムのブート後にデバッグ済みの IPQoS 構成ファイルを適用します。	システムをリブートするたびに、IPQoS 構成ファイルが確実に適用されるようにします。	831 ページの「リブート後にも IPQoS 構成を適用する方法」
3. `syslog` を使用した IPQoS のログ記録を有効にします。	エントリを追加して、`syslog` による IPQoS メッセージのログ記録を有効にします。	832 ページの「ブート時に IPQoS メッセージを記録する方法」

タスク	説明	手順
4. 発生する IPQoS の問題を解決します。	エラーメッセージを利用して IPQoS の問題を解決します。	表 32–1 のエラーメッセージを参照してください。

IPQoS 構成の適用

IPQoS 構成の有効化およびそのほかの操作には、 `ipqosconf` コマンドを使用します。

▼ 新規構成を IPQoS カーネルモジュールへ適用する方法

`ipqosconf` コマンドで、IPQoS 構成ファイルを読み取り、UNIX カーネルで IPQoS モジュールを構成します。次の手順では、`Creating IPQoS Configuration Files for Web Servers`で作成した 804 ページの「Web サーバー用 IPQoS 構成ファイルの作成」ファイルを例として使用します。詳細については、`ipqosconf(1M)` を参照してください。

1. **IPQoS 対応システムで、Primary Administrator の役割を引き受けるか、スーパーユーザーになります。**

   Primary Administrator 役割には、Primary Administrator プロファイルが含まれます。役割を作成してユーザーに役割を割り当てるには、『Oracle Solaris の管理: 基本管理』の第 2 章「Solaris 管理コンソールの操作 (タスク)」を参照してください。

2. **新規構成を適用します。**

   ```
   # /usr/sbin/ipqosconf -a/var/ipqos/Goldweb.qos
   ```

   `ipqosconf` により、指定された IPQoS 構成ファイル内の情報が Oracle Solaris カーネル内の IPQoS モジュールに書き込まれます。この例では、`/var/ipqos/Goldweb.qos` の内容が現行の Oracle Solaris カーネルに適用されます。

   注 – `-a` オプションを指定して IPQoS 構成ファイルを適用すると、ファイル内のアクションが現行のセッションの間だけ有効になります。

   ---

3. **新規 IPQoS 構成のテストおよびデバッグを行います。**

   UNIX ユーティリティーを使用して、IPQoS の動作を追跡し、IPQoS 実装に関する統計を収集します。この情報は、構成が予想どおりに機能するかを判断するのに役立ちます。

参照

- IPQoS モジュールがどのように機能するかに関する統計は、840 ページの「統計情報の収集」を参照してください。

- ipqosconf メッセージをログ記録するには、831 ページの「IPQoS メッセージの syslog によるログ記録の有効化」を参照してください。
- ブートのたびに現在の IPQoS 構成を適用させるには、831 ページの「リブート後にも IPQoS 構成を適用する方法」を参照してください。

▼ リブート後にも IPQoS 構成を適用する方法

リブート後にも IPQoS 構成を持続させるには、明示的に指定する必要があります。それ以外の場合、システムのリブート後に現行の構成が適用されません。システムで IPQoS が適正に動作するときは、次の操作を実行してリブート後にも構成が持続するようにします。

1 **IPQoS 対応システムで、Primary Administrator の役割を引き受けるか、スーパーユーザーになります。**

Primary Administrator 役割には、Primary Administrator プロファイルが含まれます。役割を作成してユーザーに役割を割り当てるには、『Oracle Solaris の管理: 基本管理』の第 2 章「Solaris 管理コンソールの操作 (タスク)」を参照してください。

2 **カーネルモジュール内に IPQoS 構成が存在することを確認します。**

```
# ipqosconf -l
```

構成がすでに存在する場合は、ipqosconf によって画面に表示されます。出力が行われない場合は、830 ページの「新規構成を IPQoS カーネルモジュールへ適用する方法」の説明に従って、構成を適用します。

3 **IPQoS システムをリブートするたびに既存の IPQoS 構成が適用されるようにします。**

```
# /usr/sbin/ipqosconf -c
```

-c オプションを指定すると、現行の IPQoS 構成が、ブート時の構成ファイル /etc/inet/ipqosinit.conf に書き込まれます。

IPQoS メッセージの syslog によるログ記録の有効化

IPQoS ブート時のメッセージを記録するには、次に示す手順に従って /etc/syslog.conf ファイルを変更する必要があります。

▼ ブート時に IPQoS メッセージを記録する方法

1. **IPQoS 対応システムで、Primary Administrator の役割を引き受けるか、スーパーユーザーになります。**

   Primary Administrator 役割には、Primary Administrator プロファイルが含まれます。役割を作成してユーザーに役割を割り当てるには、『Oracle Solaris の管理: 基本管理』の第 2 章「Solaris 管理コンソールの操作 (タスク)」を参照してください。

2. **/etc/syslog.conf ファイルを開きます。**

3. **ファイルの最後に、次のエントリを追加します。**

   ```
   user.info                       /var/adm/messages
   ```

   列の区切りは、空白ではなくタブを使用してください。

   このエントリを指定すると、IPQoS により生成されたブート時のメッセージがすべて /var/adm/messages ファイルに記録されます。

4. **システムをリブートして設定を適用します。**

例 32–1 /var/adm/messages からの IPQoS 出力

システムのリブート後に /var/adm/messages を表示すると、出力に次のような IPQoS ログメッセージが含まれることがあります。

```
May 14 10:44:33 ipqos-14 ipqosconf: [ID 815575 user.info]
 New configuration applied.
May 14 10:44:46 ipqos-14 ipqosconf: [ID 469457 user.info]
Current configuration saved to init file.
May 14 10:44:55 ipqos-14 ipqosconf: [ID 435810 user.info]
Configuration flushed.
```

また、IPQoS システムの /var/adm/messages ファイル内に、次のような IPQoS エラーメッセージが表示される場合もあります。

```
May 14 10:56:47 ipqos-14 ipqosconf: [ID 123217 user.error]
 Missing/Invalid config file fmt_version.
May 14 10:58:19 ipqos-14 ipqosconf: [ID 671991 user.error]
No ipgpc action defined.
```

これらのエラーメッセージについては、表 32–1 を参照してください。

IPQoS のエラーメッセージの障害追跡

このセクションには、IPQoS によって生成されるエラーメッセージと考えられる解決方法を示します。

表 32–1 IPQoS のエラーメッセージ

エラーメッセージ	説明	解決方法
`Undefined action in parameter` *parameter-name's* `action` *action-name*	*parameter-name* に指定したアクション名が IPQoS 構成ファイル内に存在しません。	アクションを作成します。またはパラメータ内の別の既存のアクションを参照します。
`action` *action-name* `involved in cycle`	IPQoS 構成ファイル内の *action-name* はアクション循環の一部です。これは IPQoS では許可されません。	アクション循環を決定します。次に、IPQoS 構成ファイルから循環参照の 1 つを削除します。
`Action` *action-name* `isn't referenced by any other actions`	`ipgpc` アクション以外で、ほかの定義済みアクションにより参照されないアクション定義が IPQoS 構成内にあります。これは IPQoS では許可されません。	参照されていないアクションを削除します。または別のアクションに現在参照されていないアクションを参照させます。
`Missing/Invalid config file fmt_version`	構成ファイルの書式がファイルの最初のエントリに指定されていません。これは IPQoS では必須です。	806 ページの「IPQoS 構成ファイルを作成し、トラフィッククラスを定義する方法」の説明に従って、書式のバージョンを追加します。
`Unsupported config file format version`	IPQoS がサポートしない書式のバージョンが、構成ファイル内で指定されています。	書式のバージョンを、Solaris 9 9/02 以降のリリースの IPQoS で必要な `fmt_version 1.0` に変更します。
`No ipgpc action defined.`	構成ファイル内で、`ipgpc` クラシファイアのアクションが定義されていません。これは IPQoS では必須です。	806 ページの「IPQoS 構成ファイルを作成し、トラフィッククラスを定義する方法」のように `ipgpc` のアクションを定義します。
`Can't commit a null configuration`	`ipqosconf -c` を実行して空の構成をコミットしようとしました。IPQoS は空の構成を許可しません。	構成ファイルを確実に適用してから構成をコミットします。手順については、830 ページの「新規構成を IPQoS カーネルモジュールへ適用する方法」を参照してください。
`Invalid CIDR mask on line` *line-number*	構成ファイル内で、CIDR マスクの IP アドレスとして無効なアドレスを使用しました。	マスク値を 1–32 (IPv4 の場合) および 1–128 (IPv6 の場合) の範囲内の値に変更します。
`Address masks aren't allowed for host names line` *line-number*	構成ファイル内で、ホストの CIDR マスク値を定義しました。これは IPQoS では許可されません。	マスクを削除するか、あるいはホスト名を IP アドレスに変更します。

表 32–1 IPQoS のエラーメッセージ (続き)

エラーメッセージ	説明	解決方法
`Invalid module name line` *line-number*	構成ファイル内のアクション文に無効なモジュール名を指定しました。	モジュール名のスペルに入力ミスがないか確認します。IPQoS モジュールのリストについては、表 34–5 を参照してください。
`ipgpc action has incorrect name line` *line-number*	構成ファイル内で `ipgpc` アクションに付けた名前が、必須の `ipgpc.classify` ではありません。	アクション名を `ipgpc.classify` に変更します。
`Second parameter clause not supported line` *line-number*	構成ファイル内で、単一のアクションに対し 2 つのパラメータ句を指定しました。これは IPQoS では許可されません。	アクションのパラメータすべてを結合して、単一のパラメータ句にします。
`Duplicate named action`	構成ファイル内で、2 つのアクションに同じ名前を付けました。	どちらかのアクションの名前を変更するか、あるいは削除します。
`Duplicate named filter/class in action` *action-name*	1 つのアクション内の 2 つのフィルタまたは 2 つのクラスに同じ名前を付けました。これは IPQoS 構成ファイルでは許可されません。	どちらかのフィルタまたはクラスの名前を変更するか、あるいは削除します。
`Undefined class in filter` *filter-name* `in action` *action-name*	フィルタが、構成ファイル内のアクションで定義されていないクラスを参照します。	クラスを作成するか、あるいは既存のクラスへの参照に変更します。
`Undefined action in class` *class-name* `action` *action-name*	クラスが、構成ファイル内で定義されていないアクションを参照します。	アクションを作成するか、あるいは既存のアクションへの参照に変更します。
`Invalid parameters for action` *action-name*	構成ファイル内のパラメータのどれかが無効です。	指名されたアクションが呼び出すモジュールについては、843 ページの「IPQoS アーキテクチャーと Diffserv モデル」のモジュールエントリを参照してください。あるいは、`ipqosconf(1M)` を参照します。
`Mandatory parameter missing for action` *action-name*	アクションに必要なパラメータが構成ファイル内に定義されていません。	指名されたアクションが呼び出すモジュールについては、843 ページの「IPQoS アーキテクチャーと Diffserv モデル」のモジュールエントリを参照してください。あるいは、`ipqosconf(1M)` のマニュアルページを参照します。
`Max number of classes reached in ipgpc`	IPQoS 構成ファイルの `ipgpc` アクションに、許可される数を超えたクラスを指定しました。最大数は 10007 です。	構成ファイルを確認して、不要なクラスを削除します。または、`/etc/system` ファイルにエントリ `ipgpc_max_classes`*class-number* を追加して最大クラス数を増やすこともできます。

表 32–1 IPQoS のエラーメッセージ (続き)

エラーメッセージ	説明	解決方法
`Max number of filters reached in action ipgpc`	IPQoS 構成ファイルの `ipgpc` アクションに、許可される数を超えたフィルタを指定しました。最大数は 10007 です。	構成ファイルを確認して、不要なフィルタを削除します。または、`/etc/system` ファイルにエントリ `ipgpc_max_filters`*filter-number* を追加して最大フィルタ数を増やすこともできます。
`Invalid/missing parameters for filter` *filter-name* `in action ipgpc`	構成ファイル内で、フィルタ *filter-name* に無効なパラメータが指定されているか、あるいはパラメータが不足しています。	`ipqosconf(1M)` のマニュアルページで、有効なパラメータのリストを参照します。
`Name not allowed to start with '!', line` *line-number*	アクション、フィルタまたはクラス名の最初に感嘆符 (!) を記述しました。IPQoS ファイルでは感嘆符は許可されていません。	感嘆符を削除するか、あるいは、アクション、クラス、またはフィルタの名前を変更します。
`Name exceeds the maximum name length line` *line-number*	構成ファイル内のアクション、クラス、またはフィルタの名前が、最大長の 23 文字を超えています。	アクション、クラス、またはフィルタの名前を短くします。
`Array declaration line` *line-number* `is invalid`	構成ファイル内で、*line-number* 行のパラメータの配列宣言が無効です。	無効な配列を持つ `action` 文が呼び出す配列宣言の正しい構文については、843 ページの「IPQoS アーキテクチャーと Diffserv モデル」を参照してください。あるいは、`ipqosconf(1M)` のマニュアルページを参照します。
`Quoted string exceeds line,` *line-number*	文字列の最後の閉じ引用符が同一行に存在しません。これは構成ファイルでは必須です。	引用符で囲まれた文字列を、構成ファイルの同一行内に収めます。
`Invalid value, line` *line-number*	構成ファイルの *line-number* に、パラメータとしてサポートされない値が指定されています。	`action` 文が呼び出すモジュールの許容値については、843 ページの「IPQoS アーキテクチャーと Diffserv モデル」のモジュールの説明を参照してください。あるいは、`ipqosconf(1M)` のマニュアルページを参照します。
`Unrecognized value, line` *line-number*	構成ファイルの *line-number* に、パラメータとしてサポートされない列挙値が指定されています。	パラメータの列挙値が適正であるかどうかを確認します。認識されない行番号を持つ `action` 文の説明については、843 ページの「IPQoS アーキテクチャーと Diffserv モデル」を参照してください。あるいは、`ipqosconf(1M)` のマニュアルページを参照します。
`Malformed value list line` *line-number*	構成ファイルの *line-number* で指定された列挙が、仕様構文に適合しません。	間違った形式の値リストを持つ `action` 文が呼び出すモジュールの正しい構文については、843 ページの「IPQoS アーキテクチャーと Diffserv モデル」のモジュールの説明を参照してください。あるいは、`ipqosconf(1M)` のマニュアルページを参照します。

表 32–1 IPQoS のエラーメッセージ (続き)

エラーメッセージ	説明	解決方法
`Duplicate parameter line` *line-number*	重複したパラメータが *line-number* に指定されています。これは構成ファイルでは許可されません。	重複したパラメータのどちらかを削除します。
`Invalid action name line` *line-number*	構成ファイルの *line-number* のアクションに、定義済みの名前 continue または drop を含む名前を付けました。	定義済みの名前を使用しないよう、アクションの名前を変更します。
`Failed to resolve src/dst host name for filter at line` *line-number*`, ignoring filter`	構成ファイル内で、あるフィルタ用に定義された発信元または着信先アドレスを、`ipqosconf` が解釈処理できません。このため、このフィルタは無視されます。	フィルタが重要な場合、あとで構成の適用を試みます。
`Incompatible address version line` *line-number*	*line-number* 上の IP バージョンのアドレスが、すでに指定済みの IP アドレスのバージョンまたは `ip_version` パラメータと互換性がありません。	競合する 2 つのエントリを変更して、互換性を持たせます。
`Action at line` *line-number* `has the same name as currently installed action, but is for a different module`	システムの IPQoS 構成内にすでに存在するアクションのモジュールを変更しようとしました。これは許可されません。	現行の構成をフラッシュしてから、新しい構成を適用します。

◆ ◆ ◆ 第 33 章

フローアカウンティングの使用と統計情報の収集 (タスク)

この章では、IPQoS システムによって処理されるトラフィックに関して、アカウンティング情報と統計情報を取得する方法について説明します。次の内容について説明します。

フローアカウンティングの設定 (タスクマップ)

次のタスクマップは、flowacct モジュールを使用してトラフィックフローに関する情報を取得するための一般的なタスクを示しています。マップでは、これらのタスクの実施手順へのリンクも示しています。

タスク	説明	手順
1. トラフィックフローのアカウンティング情報を格納するためのファイルを作成する	acctadm コマンドを使用して、flowacct による処理結果を格納するファイルを作成する	838 ページの「フローアカウンティングデータ用のファイルを作成する方法」
2. flowacct のパラメータを IPQoS 構成ファイルに定義する	timer、timeout、および max_limit の各パラメータの値を定義する	812 ページの「IPQoS 構成ファイル内でクラスのアカウンティングを有効にする方法」

トラフィックフローに関する情報の記録

トラフィックフローに関する情報を収集するには、IPQoS `flowacct` モジュールを使用します。たとえば、発信元アドレスや宛先アドレス、フロー内のパケット数などのデータを収集することが可能です。フローに関する情報を蓄積して記録するプロセスのことを「フローアカウンティング」と呼びます。

特定のクラスのトラフィックに関するフローアカウンティングの結果は、フローレコードというテーブルに記録されます。各フローレコードは、一連の属性から構成されます。これらの属性には、特定のクラスの一定時間のトラフィックフローに関するデータが格納されます。`flowacct` 属性のリストについては、表 34–4 を参照してください。

フローアカウンティングは、サービスレベル契約 (SLA) に定義されているとおりに顧客に課金するために、非常に役立ちます。また、フローアカウンティングを使って、重要なアプリケーションのフロー統計情報を取得することもできます。このセクションでは、`flowacct` を Oracle Solaris 拡張アカウンティング機能と組み合わせて、トラフィックフローに関するデータを取得するためのタスクについて説明します。

この章以外の場所からも次の情報が入手できます。

- IPQoS 構成ファイル内の `flowacct` のアクション文の作成方法については、823 ページの「IPQoS 構成ファイル内でフロー制御を構成する方法」を参照してください。
- `flowacct` がどのように機能するかについては、843 ページの「クラシファイアモジュール」を参照してください。
- 技術的な情報については、`flowacct(7ipp)` のマニュアルページを参照してください。

▼ フローアカウンティングデータ用のファイルを作成する方法

`flowacct` アクションを IPQoS 構成ファイルに追加する前に、`flowacct` モジュールからフローレコードのファイルを作成します。このためには、`acctadm` コマンドを使用します。`acctadm` では、基本属性または拡張属性のどちらもファイルに記録できます。すべての `flowacct` 属性のリストについては、表 34–4 を参照してください。`acctadm` については、`acctadm(1M)` のマニュアルページを参照してください。

1 **IPQoS 対応システムで、Primary Administrator の役割を引き受けるか、スーパーユーザーになります。**

Primary Administrator 役割には、Primary Administrator プロファイルが含まれます。役割を作成してユーザーに役割を割り当てるには、『Oracle Solaris の管理: 基本管理』の第 2 章「Solaris 管理コンソールの操作 (タスク)」を参照してください。

2 基本フローアカウンティングファイルを作成します。

次の例で、例 31–1 で構成されるプレミアム Web サーバー用の基本的なフローアカウンティングファイルを作成する方法を示します。

```
# /usr/sbin/acctadm -e basic -f /var/ipqos/goldweb/account.info flow
```

`acctadm -e`	`acctadm` を `-e` オプションを指定して呼び出します。`-e` オプションによって、あとに続く引数が有効になる
`basic`	`flowacct` の 8 つの基本属性のデータだけがファイルに記録されることを示す
`/var/ipqos/goldweb/account.info`	`flowacct` から得られるフローレコードを格納するファイルの絶対パス名を示す
`flow`	`acctadm` にフローアカウンティングを有効にするよう指示する

3 引数を指定しないで **acctadm** と入力し、**IPQoS** システムのフローアカウンティングに関する情報を表示します。

`acctadm` によって次の出力が生成されます。

```
Task accounting: inactive
       Task accounting file: none
     Tracked task resources: none
   Untracked task resources: extended
         Process accounting: inactive
    Process accounting file: none
  Tracked process resources: none
Untracked process resources: extended,host,mstate
           Flow accounting: active
      Flow accounting file: /var/ipqos/goldweb/account.info
     Tracked flow resources: basic
   Untracked flow resources: dsfield,ctime,lseen,projid,uid
```

最後の 4 つのエントリ以外はすべて、Oracle Solaris のリソースマネージャー機能で使用されます。次の表では、IPQoS に固有のエントリについて説明します。

エントリ	説明
`Flow accounting: active`	フローアカウンティングが有効になっていることを示す
`Flow accounting file: /var/ipqos/goldweb/account.info`	現在のフローアカウンティングファイルの名前を示す
`Tracked flow resources: basic`	基本フロー属性だけが記録されることを示す
`Untracked flow resources: dsfield,ctime,lseen,projid,uid`	ファイルに記録されない `flowacct` の属性を示す

4 (オプション) 拡張属性をアカウンティングファイルに追加します。

```
# acctadm -e extended -f /var/ipqos/goldweb/account.info flow
```

5 (オプション) 基本属性だけがアカウンティングファイルに記録されるような設定に戻します。

```
# acctadm -d extended -e basic -f /var/ipqos/goldweb/account.info
```

`-d` オプションによって拡張アカウンティングが無効になります。

6 フローアカウンティングファイルの内容を参照します。

フローアカウンティングファイルの内容の参照方法については、『System Administration Guide: Oracle Solaris Containers-Resource Management and Oracle Solaris Zones』の「Perl Interface to libexacct」を参照してください。

参照
- 拡張アカウンティング機能の詳細については、『System Administration Guide: Oracle Solaris Containers-Resource Management and Oracle Solaris Zones』の第 4 章「Extended Accounting (Overview)」を参照してください。
- IPQoS 構成ファイル内に `flowacct` パラメータを定義するには、812 ページの「IPQoS 構成ファイル内でクラスのアカウンティングを有効にする方法」を参照してください。
- `acctadm` で作成されたファイル内のデータを出力するには、『System Administration Guide: Oracle Solaris Containers-Resource Management and Oracle Solaris Zones』の「Perl Interface to libexacct」を参照してください。

統計情報の収集

`kstat` コマンドを使用すると、IPQoS モジュールから統計情報を生成できます。構文は次のとおりです。

`/bin/kstat -m` *ipqos-module-name*

表 34–5 に示されている、有効な IPQoS モジュール名であればどれでも指定できます。たとえば、`dscpmk` マーカーによって生成される統計情報を表示するには、次の形の `kstat` を使用します。

```
/bin/kstat -m dscpmk
```

技術的な情報については、`kstat(1M)` のマニュアルページを参照してください。

例 33–1 IPQoS 用の `kstat` 統計

ここでは、`kstat` を実行して `flowacct` モジュールに関する統計情報を取得した場合に予想される結果の一例について説明します。

例 33–1 IPQoS 用の kstat 統計 (続き)

```
# kstat -m flowacct
module: flowacct                        instance: 3
name:   Flowacct statistics             class:    flacct
        bytes_in_tbl                    84
        crtime                          345728.504106363
        epackets                        0
        flows_in_tbl                    1
        nbytes                          84
        npackets                        1
        snaptime                        345774.031843301
        usedmem                         256
```

`class: flacct` トラフィックフローが属するクラスの名前 (この例では `flacct`) を示す

`bytes_in_tbl` フローテーブルの総バイト数。フローテーブルの総バイト数とは、フローテーブルに現在格納されているすべてのフローレコードの合計バイト数。このフローテーブルの総バイト数は 84 である。テーブルにフローがない場合、`bytes_in_tbl` の値は 0 になる

`crtime` この `kstat` 出力が作成された最も最近の時間

`epackets` 処理中にエラーが発生したパケットの数 (この例では 0)

`flows_in_tbl` フローテーブルのフローレコード数 (この例では 1)。テーブルにレコードがない場合、`flows_in_tbl` の値は 0 になる

`nbytes` この `flowacct` アクションのインスタンスで表示される合計バイト数 (この例では 84)。フローテーブルに現在格納されているバイトを含む値。この値には、タイムアウトになり、フローテーブルに現在は含まれていない値も含まれる

`npackets` この `flowacct` アクションのインスタンスで表示される合計パケット数 (この例では 1)。`npackets` には、フローテーブルに現在あるパケットが含まれる。`npackets` には、タイムアウトになり、フローテーブルに現在は含まれていないパケットも含まれる。

`usedmem` この `flowacct` インスタンスで保持されているフローテーブルが使用しているメモリーのバイト数。この例では、`usedmem` の値は 256。フローテーブルにフローレコードがまったく存在しない場合、`usedmem` の値は 0 になる

◆ ◆ ◆ 第 34 章

IPQoS の詳細 (リファレンス)

この章は、IPQoS の詳細を説明するリファレンスです。この章では、次の内容について説明します。

- 843 ページの「IPQoS アーキテクチャーと Diffserv モデル」
- 856 ページの「IPQoS 構成ファイル」
- 860 ページの「ipqosconf 構成ユーティリティー」

概要は、第 29 章「IPQoS の紹介 (概要)」を参照してください。計画については、第 30 章「IPQoS 対応ネットワークの計画 (タスク)」を参照してください。IPQoS の構成手順については、第 31 章「IPQoS 構成ファイルの作成 (手順)」を参照してください。

IPQoS アーキテクチャーと Diffserv モデル

このセクションでは、IPQoS アーキテクチャーとこのアーキテクチャーが RFC 2475, An Architecture for Differentiated Services (`http://www.ietf.org/rfc/rfc2475.txt?number=2475`) で定義された差別化サービス (Diffserv) モデルを実装する方法について説明します。次に示す Diffserv モデルの要素が、IPQoS に含まれます。

- クラシファイア
- メーター
- マーカー

さらに、IPQoS には、仮想ローカルエリアネットワーク (VLAN) デバイスで使用されるフローアカウンティングモジュールと `dlcosmk` マーカーが含まれています。

クラシファイアモジュール

Diffserv モデルでは、「クラシファイア」は、トラフィックフローを選択して、それぞれに異なるサービスレベルを適用するためのグループに分類する作業を担当します。RFC 2475 で定義されたクラシファイアは、当初、境界ルーター用に設計されま

した。それとは対照的に、IPQoS クラシファイア ipgpc は、内部ホストからローカルネットワークへのトラフィックフローを処理するために設計されています。このため、IPQoS システムと Diffserv ルーターの両方を備えたネットワークは、より広範囲な差別化サービスを提供できます。ipgpc の技術情報については、ipgpc(7ipp) のマニュアルページを参照してください。

ipgpc クラシファイアは、次の機能を実行します。

1. IPQoS 対応システムの IPQoS 構成ファイルに指定された条件を満たすトラフィックフローを選択します。

   QoS ポリシーは、パケットヘッダーに存在する必要のあるさまざまな条件を定義します。これらの条件は、「セレクタ」と呼ばれます。 ipgpc クラシファイアは、これらのセレクタを、IPQoS システムから受信したパケットのヘッダーと比較して、一致するパケットをすべて選択します。
2. パケットフローを、IPQoS 構成ファイルの定義に従い、同じ特性を持つネットワークトラフィックである「クラス」に分類します。
3. パケットの差別化サービス (DS) フィールドの値を調べ、差別化サービスコードポイント (DSCP) の存在を確認します

   DSCP は、受信したトラフィックに送信側によって転送動作のマークが付けられているかどうかを示します。
4. 特定クラスのパケットに関して、IPQoS 構成ファイル内で次に指定されているアクションを調べます。
5. パケットを、IPQoS 構成ファイルで指定された次の IPQoS モジュールに渡すか、あるいはネットワークストリームに戻します。

クラシファイアの概要は、771 ページの「クラシファイア (ipgpc) の概要」を参照してください。IPQoS 構成ファイルでクラシファイアを呼び出すには、856 ページの「IPQoS 構成ファイル」を参照してください。

IPQoS セレクタ

ipgpc クラシファイアは、IPQoS 構成ファイルの filter 句で使用可能なさまざまなセレクタをサポートします。フィルタを定義するときには、特定クラスのトラフィック取得に必要な最小限のセレクタを使用してください。定義するフィルタの数が、IPQoS のパフォーマンスに影響を与える可能性があります。

次の表に、 ipgpc で使用できるセレクタを示します。

表 34–1 IPQoS クラシファイアで利用可能なフィルタセレクタ

セレクタ	引数	選択される情報
saddr	IP アドレス番号	発信元アドレス
daddr	IP アドレス番号	着信先アドレス

表 34–1 IPQoS クラシファイアで利用可能なフィルタセレクタ *(続き)*

セレクタ	引数	選択される情報
sport	ポート番号またはサービス名。/etc/services の定義に従う	トラフィッククラスの発信元ポート
dport	ポート番号またはサービス名。/etc/services の定義に従う	トラフィッククラスの着信先ポート
プロトコル	プロトコル番号またはプロトコル名。/etc/protocols の定義に従う	このトラフィッククラスが使用するプロトコル
dsfield	0 - 63 の値を持つ DS コードポイント (DSCP)	DSCP。パケットに適用される転送動作を定義する。このパラメータを指定した場合は、dsfield_mask パラメータも指定すること
dsfield_mask	0 - 255 の値を持つビットマスク	dsfield セレクタと組み合わせて使用。dsfield_mask は、dsfield セレクタに適用して、ビットのどれが一致するかを決定する
if_name	インタフェース名	特定クラスの着信トラフィックまたは発信トラフィックで使用されるインタフェース
user	選択する UNIX ユーザー ID の番号またはユーザー名。パケットにユーザー ID またはユーザー名が存在しない場合、デフォルトの –1 が使用される	アプリケーションに指定されるユーザー ID
projid	選択するプロジェクト ID の番号	アプリケーションに付加されるプロジェクト ID
priority	優先順位の番号。もっとも低い優先順位は 0	このクラスのパケットに与えられる優先順位。優先順位は、同じクラスに複数存在するフィルタの重要度の順位付けに使用される
direction	次の引数のいずれかを指定できる。	IPQoS マシン上のパケットフローの方向
	LOCAL_IN	ローカルシステムから IPQoS システムへの入力トラフィック
	LOCAL_OUT	ローカルシステムから IPQoS システムへの出力トラフィック
	FWD_IN	転送される入力トラフィック
	FWD_OUT	転送される出力トラフィック
precedence	優先度の値。もっとも高い優先度は 0	優先度は、同一優先順位のフィルタの順序付けに使用される
ip_version	V4 または V6	パケットにより使用されるアドレス指定スキーム (IPv4 または IPv6)

メーターモジュール

「メーター」はフローの転送速度をパケット単位で追跡します。このメーターは、構成されているパラメータにパケットが一致するかどうかを決定します。メーターモジュールは、パケットサイズ、構成されたパラメータ、およびフロー速度に基づき、パケットの次のアクションをアクションセットの中から決定します。

メーターには 2 つのメータリングモジュール、すなわち `tokenmt` および `tswtclmt` があります。モジュールの構成は、IPQoS 構成ファイルで行います。モジュールのどちらか一方または両方をクラスに構成できます。

メータリングモジュールを構成する際、速度に関する 2 つのパラメータを定義できます。

- `committed-rate` – 特定クラスのパケットに容認可能な転送速度を bps で定義する
- `peak-rate` – 特定クラスのパケットに最大限容認可能な転送速度を bps で定義する

パケットに対するメータリングアクションの結果 (outcome) は、次の 3 つのどれかになります。

- `green` – パケットの生成するフローは認定速度内である
- `yellow` – パケットの生成するフローは認定速度を超過しているが、最大速度内である
- `red` – パケットの生成するフローは最大速度を超過している

IPQoS 構成ファイル内で、結果ごとに異なるアクションを構成できます。認定速度および最大速度については、次のセクションで説明します。

`tokenmt` メータリングモジュール

`tokenmt` モジュールは、「トークンバケット」を使用してフローの転送速度を測定します。`tokenmt` は、シングルレートメーターまたはツーレートメーターとして機能するように構成できます。`tokenmt` アクションインスタンスは、2 つのトークンバケットを管理します。これらのトークンバケットは、トラフィックフローが構成されたパラメータに適合するかどうかを調べます。

`tokenmt(7ipp)` のマニュアルページでは、IPQoS がどのようにトークンメーターパラダイムを実装するかが説明されています。トークンバケットに関する一般的な情報は、Kalevi Kilkki 著『*Differentiated Services for the Internet*』および多数の Web サイトで入手できます。

tokenmt の構成パラメータを次に示します。

- committed_rate – フローの認定速度を bps で指定する
- committed_burst – 認定バーストサイズをビット単位で指定する。committed_burst パラメータは、認定速度でネットワークに渡すことのできる、特定クラスの発信パケット数を定義する
- peak_rate – 最大速度を bps で指定する
- peak_burst – 最大バーストサイズまたは超過バーストサイズをビット単位で指定する。peak_burst パラメータは、トラフィッククラスに、認定速度を超過する最大バーストサイズを付与する
- color_aware – tokenmt のカラーアウェアモードを有効にする
- color_map – DSCP 値を緑、黄、または赤にマッピングする整数配列を定義する

tokenmt をシングルレートメーターとして構成する

tokenmt をシングルレートメーターとして構成するには、IPQoS 構成ファイル内で tokenmt に peak_rate パラメータを指定しないでください。赤、緑、または黄の結果 (outcome) を識別するようにシングルレートの tokenmt インスタンスを構成するには、peak_burst パラメータを指定する必要があります。peak_burst パラメータを使用しないことによって、tokenmt が赤または緑の結果だけを識別するように構成することもできます。2 つの出力を持つシングルレート tokenmt の例については、例 31–3 を参照してください。

tokenmt がシングルレートメーターとして機能する場合、 peak_burst パラメータは実質的にバーストサイズを超過します。committed_burst と peak_burst のどちらかと committed_rate は、ゼロ以外の正の整数にする必要があります。

tokenmt をツーレートメーターとして構成する

tokenmt をツーレートメーターとして構成するには、IPQoS 構成ファイル内で tokenmt アクション用の peak_rate パラメータを指定します。ツーレートの tokenmt は、必ず赤、黄、および緑の 3 つの結果 (outcome) を識別します。committed_rate、committed_burst、および peak_burst パラメータは、ゼロ以外の正の整数にする必要があります。

tokenmt をカラーアウェアとして構成する

ツーレートの tokenmt をカラーアウェアとして構成するには、「カラーアウェアネス」を特に追加するパラメータを追加します。tokenmt をカラーアウェアとして構成する action 文の例を次に示します。

例 34–1　IPQoS 構成ファイル用のカラーアウェア tokenmt アクション

```
action {
    module tokenmt
    name meter1
```

例 34–1 IPQoS 構成ファイル用のカラーアウェア tokenmt アクション (続き)

```
    params {
          committed_rate 4000000
          peak_rate 8000000
          committed_burst 4000000
          peak_burst 8000000
          global_stats true
          red_action_name continue
          yellow_action_name continue
          green_action_name continue
          color_aware true
          color_map {0-20,22:GREEN;21,23-42:RED;43-63:YELLOW}
    }
}
```

color_aware パラメータを true に設定することによって、カラーアウェアを有効にできます。カラーアウェアにした tokenmt メーターは、以前の tokenmt アクションによってパケットが赤、黄、または緑にマーキング済みであるものと見なします。カラーアウェアの tokenmt は、ツーレートメーター用のパラメータに加え、パケットヘッダー内の DSCP も使用してパケットを評価します。

color_map パラメータは、パケットヘッダーの DSCP がマッピングされる配列を含みます。次の color_map 配列について説明します。

```
color_map {0-20,22:GREEN;21,23-42:RED;43-63:YELLOW}
```

DSCP が 0~20 および 22 のパケットは緑にマッピングされます。DSCP が 21 および 23~42 のパケットは赤にマッピングされます。DSCP が 43~63 のパケットは黄にマッピングされます。tokenmt は、デフォルトのカラーマップを格納します。ただし、このデフォルトは必要に応じて color_map パラメータを使用して変更できます。

*color*_action_name パラメータでは、continue を指定するとパケットの処理を完了できます。また、たとえば yellow_action_name mark22 のように、引数を指定してパケットをマーカーアクションに送信することもできます。

tswtclmt メータリングモジュール

tswtclmt メータリングモジュールは、時間ベースの「速度エスティメータ」を使用して、トラフィッククラスの平均帯域幅を見積もります。tswtclmt は必ず 3 つの結果 (outcome) を識別するメーターとして機能します。速度エスティメータは、フローの到着速度の見積もりを提供します。この速度は、一定期間すなわち「時間ウィンドウ」内の、トラフィックストリームの実行帯域幅の平均を見積もります。速度概算アルゴリズムは、RFC 2859 (*A Time Sliding Window Three Colour Marker*) に基づいています。

tswtclmt を構成するには、次のパラメータを使用します。

- committed_rate – 認定速度を bps で指定する
- peak_rate – 最大速度を bps で指定する
- window – タイムウィンドウをミリ秒で定義する。このタイムウィンドウに対して平均帯域幅の履歴が記録される

tswtclmt の技術的な詳細については、tswtclmt(7ipp) のマニュアルページを参照してください。tswtclmt に似た速度シェーパの一般的な情報については、 RFC 2963, A Rate Adaptive Shaper for Differentiated Services (http://www.ietf.org/rfc/rfc2963.txt?number=2963) を参照してください。

マーカーモジュール

IPQoS には 2 つのマーカーモジュール、すなわち dscpmk および dlcosmk が含まれます。このセクションでは、両方のマーカーの使用方法を説明します。dlcosmk は VLAN デバイスを使用する IPQoS システムでだけ利用可能であるため、通常は dscpmk を使用する必要があります。

dscpmk の技術的情報については、dscpmk(7ipp) のマニュアルページを参照してください。dlcosmk の技術情報については、dlcosmk(7ipp) のマニュアルページを参照してください。

パケット転送での dscpmk マーカーの使用

マーカーは、クラシファイアモジュールまたはメータリングモジュールによって処理されたあとのトラフィックフローを受け取ります。マーカーは、転送動作をトラフィックにマークします。転送動作とは、フローが IPQoS システムから送出されたあと、フローに対して行われるアクションです。トラフィッククラスに対して実行される転送動作は、「ホップ単位動作 *(PHB)*」に定義されます。PHB はトラフィッククラスに優先順位を割り当てます。これは、そのクラスのフローに割り当てられる、ほかのトラフィッククラスに対する相対的な優先度です。PHB は、IPQoS システムの隣接するネットワーク上での転送動作だけを制御します。PHB の詳細については、776 ページの「ホップ単位動作」を参照してください。

「パケット転送」とは、特定クラスのトラフィックを、ネットワーク上の次の宛先へ送信するプロセスを指します。IPQoS システムなどのホストの場合、パケットはホストからローカルネットワークストリームへ転送されます。Diffserv ルーターの場合、パケットはローカルネットワークからルーターの次のホップへ転送されます。

マーカーは、パケットヘッダー内の DS フィールドに、IPQoS 構成ファイル内で定義された既知の転送動作のマークを付けます。以後、IPQoS システムおよびあとに続く Diffserv 対応システムは、マークが変更されないかぎり、DS フィールド内の指示に従ってトラフィックを転送します。PHB を割り当てるため、IPQoS システム

は、パケットヘッダーの DS フィールドの値をマークします。この値は、DSCP (Differentiated Services Codepoint) と呼ばれます。Diffserv アーキテクチャーは、2 種類の転送動作、すなわち EF および AF を定義しており、各転送動作はそれぞれ異なる DSCP を使用します。DSCP の概要については、776 ページの「DS コードポイント」を参照してください。

IPQoS システムは、トラフィックフローの DSCP を読み取り、ほかの送信トラフィックフローに対する相対的な優先度を評価します。次に IPQoS システムは、並行するトラフィックフローすべての優先順位を定め、各フローを優先順位に従ってネットワーク上に送出します。

Diffserv ルーターは、送信トラフィックフローを受け取り、パケットヘッダー内の DS フィールドを読み取ります。DSCP を使用すると、ルーターで現在のトラフィックフローに優先順位を付け、スケジュールを設定できます。ルーターは、PHB で指示された優先順位に従って各フローを転送します。あとに続くホップ上の Diffserv 対応システムも同じ PHB を認識する場合を除いて、ネットワークの境界ルーターを越えて PHB を適用することはできません。

完全優先転送 (Expedited Forwarding、 EF) PHB

「完全優先転送」(EF) は、推奨される EF コードポイント 46 (101110) の付いたパケットが、ネットワークに送出される時に、可能なかぎり最良の扱いを受けることを保証します。完全優先転送は、しばしば専用回線に例えられます。コードポイント 46 (101110) を持つパケットには、宛先に向かう途中、すべての Diffserv ルーターによる優先待遇が保証されます。EF の技術情報については、「An Expedited Forwarding PHB (`http://www.ietf.org/rfc/rfc2598.txt`)」を参照してください。

相対的優先転送 (Assured Forwarding、 AF) PHB

「相対的優先転送」(AF) では、4 つの異なるクラスの転送動作をマーカーに指定できます。次の表に、クラス、各クラスに指定できる 3 つのドロップ優先度、および各優先度に対応する推奨 DSCP を示します。各 DSCP は、AF 値 (10 進数値およびバイナリ値) で表されます。

表 34–2 相対的優先転送のコードポイント

	クラス 1	クラス 2	クラス 3	クラス 4
低ドロップ優先度	AF11 =	AF21 =	AF31 =	AF41 =
	10 (001010)	18 (010010)	26 (011010)	34 (100010)
中ドロップ優先度	AF12 =	AF22 =	AF32 =	AF42 =
	12 (001100)	20 (010100)	28 (011100)	36 (100100)
高ドロップ優先度	AF13 =	AF23 =	AF33 =	AF43 =
	14 (001110)	22 (010110)	30 (011110)	38 (100110)

AF コードポイントは、各トラフィッククラスに差別化転送動作を提供する際のガイドとして、すべての Diffserv 対応システム上で使用できます。

これらのパケットが Diffserv ルーターに達すると、ルーターはパケットのコードポイントを、キュー内のほかのトラフィックの DSCP とともに評価します。次にルーターは、利用可能な帯域幅、およびパケットの DSCP により割り当てられた優先順位に応じて、パケットを転送またはドロップします。EF PHB の付いたパケットは、どの AF PHB の付いたパケットよりも広い帯域幅の使用が保証されます。

ネットワーク上の IPQoS システムと Diffserv ルーターとの間でパケットのマーキングを合致させて、パケットが意図したとおりに転送されるようにしてください。たとえば、ネットワーク上の IPQoS システムがパケットにコードポイント AF21 (010010)、AF13 (001110)、AF43 (100110)、および EF (101110) を付けるとします。この場合、AF21、AF13、AF43、および EF DSCP を、Diffserv ルーターの適切なファイルに追加する必要があります。

AF コードポイント表の技術的説明については、「Assured Forwarding PHB Group (`http://tools.ietf.org/html/rfc2597`)」を参照してください。ルーターの製造元である Cisco Systems と JuniperNetworks は、Web サイトに AF PHB の設定に関する詳細な情報を載せています。この情報を使用して、IPQoS システムおよびルーター用の AF PHB を定義できます。また、ルーター製造元のドキュメントには、自社製品での DS コードポイントの設定方法が含まれています。

マーカーへの DSCP の設定

DSCP の長さは 6 ビットです。DS フィールドの長さは 1 バイトです。DSCP を定義すると、マーカーは、DS コードポイントでパケットヘッダーの最初の 6 つの重みビットをマークします。残りの 2 ビットは、使用されません。

DSCP を定義するには、マーカーアクション文の中で次のパラメータを使用します。

```
dscp_map{0-63:DS_codepoint}
```

`dscp_map` パラメータは、(DSCP) 値を使用して生成する 64 要素の配列です。`dscp_map` は、`dscpmk` マーカーによって着信 DSCP を発信 DSCP にマップするために使用されます。

DSCP 値は、10 進表記で `dscp_map` に指定する必要があります。たとえば、EF コードポイント 101110 は 10 進数値 46 に変換する必要があり、その結果 `dscp_map{0-63:46}` になります。AF コードポイントの場合、表 34–2 で示されるさまざまなコードポイントを、`dscp_map` で使用するために 10 進数表記に変換する必要があります。

VLAN デバイスでの `dlcosmk` マーカーの使用

`dlcosmk` マーカーモジュールは、データグラムの MAC ヘッダー内に転送動作をマークします。VLAN インタフェースを持つ IPQoS システムでだけ、`dlcosmk` を使用できます。

dlcosmk は、「*VLAN* タグ」と呼ばれる 4 バイトを MAC ヘッダーに追加します。VLAN タグには、IEEE 801.D 標準に定義されている 3 ビットのユーザー優先順位値が含まれます。VLAN を認識する Diffserv 対応スイッチは、データグラム内のユーザー優先順位フィールドを読み取ることができます。801.D ユーザー優先順位値は、サービスクラス (CoS) マークを実装します。CoS マークは、商用スイッチで一般的に使われています。

次の表のサービスマークのクラスを定義することによって、dlcosmk マーカーアクションのユーザー優先値を使用できます。

表 34–3 801.D ユーザー優先順位値

サービスクラス	定義
0	ベストエフォート
1	背景
2	予備
3	エクセレントエフォート
4	制御された負荷
5	応答時間 100ms 未満のビデオ
6	応答時間 10ms 未満のビデオ
7	ネットワーク制御

dlcosmk の詳細は、dlcosmk(7ipp) のマニュアルページを参照してください。

VLAN デバイスを持つシステムでの IPQoS 構成

ここでは、VLAN デバイスを持つシステムでの IPQoS の実装方法を示す、単純なネットワークのシナリオを紹介します。このシナリオには、スイッチで接続された 2 つの IPQoS システム、すなわち machine1 および machine2 が含まれます。machine1 上の VLAN デバイスの IP アドレスは 10.10.8.1、machine2 上の VLAN デバイスの IP アドレスは 10.10.8.3 です。

machine1 向けの次の IPQoS 構成ファイルは、machine2 への切り替えによる、トラフィックのマーキングの簡単な解決策を示しています。

例 34–2 VLAN デバイスを持つシステムの IPQoS 構成ファイル

```
fmt_version 1.0
action {
        module ipgpc
          name ipgpc.classify

        filter {
```

例 34–2 VLAN デバイスを持つシステムの IPQoS 構成ファイル (続き)

```
                name myfilter2
                daddr 10.10.8.3
                class myclass
        }

        class {
                name myclass
                next_action mark4
        }
}

action {
        name mark4
        module dlcosmk
        params {
                cos 4
                next_action continue
        global_stats true
        }
}
```

この構成では、machine2 上の VLAN デバイスを着信先とする machine1 からのすべてのトラフィックが、dlcosmk マーカーに渡されます。mark4 マーカーアクションは、CoS が 4 でクラスが myclass のデータグラムに VLAN マークを追加するように dlcosmk に指示します。ユーザー優先値 4 は、2 台のマシン間の切り替えによって、machine1 からの myclass トラフィックフローへの制御された負荷転送を指定しなければならないことを示します。

flowacct モジュール

IPQoS の flowacct モジュールは、トラフィックフローに関する情報を記録します。このプロセスは、「フローアカウンティング」と呼ばれます。フローアカウンティングは、顧客への課金や特定クラスへのトラフィック量の評価に使用できるデータを作成します。

フローアカウンティングは、オプションです。通常、flowacct は、メーターまたはマーカーに処理されたトラフィックフローが、ネットワークストリームへ送出される前に通る、最後のモジュールです。Diffserv モデルでの flowacct の位置の図については、図 29–1 を参照してください。flowacct の詳細な技術情報については、flowacct(7ipp) のマニュアルページを参照してください。

フローアカウンティングを有効にするには、flowacct に加えて、Oracle Solaris の exacct アカウンティング機能および acctadm コマンドを使用する必要があります。フローアカウンティングの設定の全手順については、837 ページの「フローアカウンティングの設定 (タスクマップ)」を参照してください。

flowacct パラメータ

flowacct モジュールは、フローレコードで構成されたフローテーブル内に、フローに関する情報を収集します。テーブル内の各エントリには、1 つのフローレコードが含まれます。フローテーブルは、表示できません。

フローレコードを測定してフローテーブルへ書き込むには、IPQoS 構成ファイル内で次の flowacct パラメータを定義します。

- timer – タイムアウトしたフローをフローテーブルから削除し、acctadm により作成されたファイルに書き込む間隔を、ミリ秒単位で定義する
- timeout – パケットフローがタイムアウトするまでの非アクティブな時間を、ミリ秒単位で定義する

注 – timer と timeout には異なる値を指定できます。

- max_limit – フローテーブルに格納可能なフローレコードの数に上限を設定する

flowacct パラメータの IPQoS 構成ファイルでの使用例については、823 ページの「IPQoS 構成ファイル内でフロー制御を構成する方法」を参照してください。

フローテーブル

flowacct モジュールは、flowacct インスタンスが認識するすべてのパケットフローを記録するフローテーブルを管理します。

フローは、flowacct の 8 タプルと呼ばれる、次のパラメータによって特定されます。

- 発信元アドレス
- 着信先アドレス
- 発信元ポート
- 着信先ポート
- DSCP
- ユーザー ID
- プロジェクト ID
- プロトコル番号

フローの 8 タプルのパラメータが変化しないかぎり、フローテーブルには 1 つのエントリだけが含まれます。max_limit パラメータにより、フローテーブルに含めることのできるエントリ数が決定されます。

フローテーブルは、IPQoS 構成ファイル内の timer パラメータに指定された間隔でスキャンされます。デフォルトは 15 秒です。IPQoS 構成ファイル内の timeout 間隔に指定された時間以上、IPQoS システムがパケットを認識しない場合、フローは「タ

イムアウト」します。デフォルトのタイムアウト間隔は 60 秒です。タイムアウトしたエントリは、acctadm コマンドを使用して作成されたアカウンティングファイルに書き込まれます。

flowacct レコード

flowacct レコードには、次の表に示される属性が含まれています。

表 34–4 flowacct レコードの属性

属性名	属性の内容	種類
src-addr-*address-type*	オリジネータの発信元アドレス。*address-type* は、IPQoS 構成ファイルの指定に従い、v4 (IPv4 の場合) または v6 (IPv6 の場合) になる	基本
dest-addr-*address-type*	パケットの着信先アドレス。*address-type* は、IPQoS 構成ファイルの指定に従い、v4 (IPv4 の場合) または v6 (IPv6 の場合) になる	基本
src-port	フローの起点となる発信元ポート	基本
dest-port	ブローの宛先となる着信先ポート番号	基本
プロトコル	フローのプロトコル番号	基本
total-packets	フロー内のパケット数	基本
total-bytes	フロー内のバイト数	基本
action-name	このフローを記録した flowacct アクションの名前	基本
creation-time	flowacct がそのフローのパケットを最初に認識した時間	拡張 (Extended) のみ
last-seen	そのフローのパケットを最後に認識した時間	拡張 (Extended) のみ
diffserv-field	フローの発信パケットヘッダー内の DSCP	拡張 (Extended) のみ
user	アプリケーションから取得される UNIX ユーザー ID またはユーザー名	拡張 (Extended) のみ
projid	アプリケーションから取得されるプロジェクト ID	拡張 (Extended) のみ

flowacct モジュールでの acctadm の使用

acctadm コマンドを使用して、flowacct により生成されるさまざまなフローレコードを格納するファイルを作成します。acctadm は、拡張アカウンティング機能と連動して動作します。acctadm の技術的情報については、acctadm(1M) のマニュアルページを参照してください。

flowacct モジュールは、フローを観察し、フローレコードにフローテーブルを入力します。次に flowacct は、timer に指定された間隔でパラメータと属性を評価します。last_seen 値に timeout 値を加えた時間以上パケットが検出されない場合、パケットはタイムアウトします。タイムアウトしたエントリはすべて、フローテーブルから削除されます。削除されたタイムアウトエントリは、timer パラメータに指定された時間が経過するたびに、アカウンティングファイルに書き込まれます。

acctadm を呼び出して flowacct モジュールで使用するには、次の構文を使用します。

```
acctadm -e file-type -f filename flow
```

`acctadm -e`	acctadm を -e オプションを指定して呼び出します。-e は、直後にタイプを指定することを示します。
file-type	収集する属性を指定します。*file-type* は、basic または extended に置き換える必要があります。各ファイルタイプの属性の一覧については、表 34–4 を参照してください。
`-f`*file-name*	フローレコードを格納するファイル *file-name* を作成します。
`flow`	acctadm を IPQoS 上で実行することを示します。

IPQoS 構成ファイル

このセクションでは、IPQoS 構成ファイル各部の詳細を説明します。IPQoS のブート時にアクティブになるポリシーは、/etc/inet/ipqosinit.conf ファイルに格納されています。このファイルは編集可能ですが、新しい IPQoS システムの場合、別の名前で構成ファイルを作成するのが最善の方法です。IPQoS 構成の適用とデバッグに関するタスクについては、第 31 章「IPQoS 構成ファイルの作成 (手順)」を参照してください。

IPQoS 構成ファイルの構文については、例 34–3 を参照してください。

この例では、次の表記上の規則に従います。

- computer-style type – 構成ファイル各部を説明する構文情報。このテキストは、入力しない
- **bold type** – IPQoS 構成ファイルに入力する必要のあるリテラルテキスト。たとえば、IPQoS 構成ファイルは、常に **fmt_version** で始める必要がある
- イタリック体 – 構成を説明する情報と置き換える変数テキスト。たとえば、*action-name* または *module-name* は、常に構成に関する情報で置き換える必要がある

例 34–3　IPQoS 構成ファイルの構文

```
file_format_version ::= fmt_version version

action_clause ::= action {
      name action-name
      module module-name
      params-clause |  ""
      cf-clauses
}
action_name ::= string
module_name ::= ipgpc | dlcosmk | dscpmk | tswtclmt | tokenmt | flowacct

params_clause ::= params {
      parameters
      params-stats |   ""
      }
parameters ::=     prm-name-value parameters |  ""
prm_name_value ::= param-name param-value

params_stats ::= global-stats boolean

cf_clauses ::= class-clause cf-clauses |
               filter-clause cf-clauses | ""

class_clause ::= class {
      name class-name
      next_action next-action-name
      class-stats | ""
                  }
class_name  ::= string
next_action_name  ::= string
class_stats ::= enable_stats boolean
boolean ::= TRUE | FALSE

filter_clause ::= filter {
                  name filter-name
                  class class–name
                  parameters
                  }
filter_name ::= string
```

次では、IPQoS 構成ファイルの各主要部分について説明します。

action 文

action 文を使用して、843 ページの「IPQoS アーキテクチャーと Diffserv モデル」で説明されているさまざまな IPQoS モジュールを呼び出します。

IPQoS 構成ファイルを新規作成する場合、必ずバージョン番号から始める必要があります。ついで、次の action 文を追加して、クラシファイアを呼び出す必要があります。

```
fmt_version 1.0

action {
    module ipgpc
    name ipgpc.classify
}
```

クラシファイア action 文の次に、params 句または class 句を記述します。

ほかのすべての action 文には次の構文を使用します。

```
action {
name action-name
module module-name
params-clause | ""
cf-clauses
}
```

name *action_name*	アクションに名前を付ける
module *module_name*	呼び出し予定の IPQoS モジュールを識別します。表 34–5 に記載のモジュールの 1 つでなければなりません。
params_clause	クラシファイアが処理するパラメータ (グローバル統計、次に処理するアクションなど) を指定する
cf_clauses	class 句または filter 句のゼロ以上のセット

モジュール定義

モジュールの定義によって、action 文のパラメータを処理するモジュールが示されます。IPQoS 構成ファイルには、次のモジュールを含めることができます。

表 **34–5** IPQoS モジュール

モジュール名	定義
ipgpc	IP クラシファイア
dscpmk	IP パケット内で DSCP 作成に使用するマーカー
dlcosmk	VLAN デバイスで使用するマーカー
tokenmt	トークンバケットメーター
tswtclmt	タイムスライディングウィンドウメーター
flowacct	フローアカウンティングモジュール

class 句

トラフィックのクラスごとに「class 句」を定義します。

IPQoS 構成内の残りのクラスを定義するには、次の構文を使用します。

```
class {

      name class-name
      next_action next-action-name
}
```

特定のクラスに関する統計情報収集を有効にするには、最初に ipgpc.classify アクション文でグローバル統計を有効にする必要があります。詳細は、857 ページの「action 文」を参照してください。

クラスに関する統計を収集したいときは、enable_stats TRUE 文を使用します。クラスの統計を収集する必要がない場合は、enable_stats FALSE を指定します。あるいは、 enable_stats 文を削除してもかまいません。

IPQoS 対応ネットワーク上のトラフィックは、特に定義しなければ「デフォルトクラス」になります。

filter 句

「フィルタ」は、トラフィックフローをクラスに分類するセレクタで構成されます。これらのセレクタは、クラス句で作成されたクラスのトラフィックへ適用する条件を、明確に定義します。パケットがもっとも高い優先順位のフィルタのセレクタすべてに一致する場合、パケットはそのフィルタのクラスのメンバーと見なされます。ipgpc クラシファイアと使用できるセレクタの完全なリストについては、表 34–1 を参照してください。

次の構文を持つ「*filter* 句」を使用して IPQoS 構成ファイル内にフィルタを定義します。

```
filter {
       name filter-name
       class class-name
       parameters (selectors)
       }
```

params 句

params 句には、アクション文で定義されたモジュールの処理方法が含まれます。params 句の構文を次に示します。

```
params {
            parameters
            params-stats | ""
        }
```

params 句では、モジュールに適用するパラメータを使用します。

params 句の `params-stats` 値は、`global_stats TRUE` または `global_stats FALSE` になります。`global_stats TRUE` 命令は、グローバル統計を呼び出した `action` 文に関する UNIX スタイルの統計を有効にします。`kstat` コマンドを使用して、統計情報を表示できます。クラス単位の統計を有効にする前に、`action` 文の統計を有効にする必要があります。

ipqosconf 構成ユーティリティー

IPQoS 構成ファイルを読んだり、UNIX カーネル内の IPQoS モジュールを構成したりするには、`ipqosconf` ユーティリティーを使用します。`ipqosconf` は、次のアクションを実行します。

- 構成ファイルを IPQoS カーネルモジュールに適用する (`ipqosconf -a` *filename*)
- カーネル内に現在常駐している IPQoS 構成ファイルを表示する (`ipqosconf -l`)
- マシンをリブートするたびに、現行の IPQoS 構成を読み取り、適用するようにする (`ipqosconf -c`)
- 現行の IPQoS カーネルモジュールをフラッシュする (`ipqosconf -f`)

技術的な情報は、`ipqosconf(1M)` のマニュアルページを参照してください。

用語集

3DES Triple-DES を参照してください。

AES Advanced Encryption Standard の略。対称 128 ビットブロックのデータ暗号技術。2000 年の 10 月、米国政府は暗号化標準としてこのアルゴリズムの Rijndael 方式を採用しました。AES は DES に代わる米国政府の標準として採用されています。

Blowfish 32 ビットから 448 ビットまでの可変長キーの対称ブロックの暗号化アルゴリズム。その作成者である Bruce Schneier 氏は、鍵を頻繁に変更しないアプリケーションに効果的であると述べています。

CA 認証局 (CA) を参照してください。

DEPRECATED アドレス IPMP グループ内でデータの発信元アドレスとして使用することのできない IP アドレス。通常、IPMP の検査用アドレスは DEPRECATED アドレスです。ただし、任意のアドレスに DEPRECATED のマークを付けて、そのアドレスが発信元アドレスとして使用されることを防止できます。

DES Data Encryption Standard。1975 年に開発され、1981 年に ANSI X.3.92 として ANSI で標準化された対称鍵の暗号化方式。DES では 56 ビットの鍵を使用します。

Diffie-Hellman アルゴリズム 公開鍵暗号化としても知られています。1976 年に Diffie 氏と Hellman 氏が開発した非対称暗号鍵協定プロトコルです。このプロトコルを使用すると、セキュリティー保護されていない伝達手段で、事前の秘密情報がなくても 2 人のユーザーが秘密鍵を交換できます。Diffie-Hellman は、IKE プロトコルで使用されます。

diffserv モデル IP ネットワークで差別化サービスを実装するための IETF (Internet Engineering Task Force) のアーキテクチャー標準。主なモジュールとして、クラシファイア、メーター、マーカー、スケジューラ、およびドロッパがあります。IPQoS では、クラシファイア、メーター、およびマーカーの各モジュールを実装します。diffserv モデルについては、RFC 2475 (*An Architecture for Differentiated Services*) に解説されています。

DSA デジタル署名アルゴリズム。512 ビットから 4096 ビットまでの可変長鍵の公開鍵アルゴリズム。米国政府標準である DSS は最大 1024 ビットです。この場合、DSA では入力に SHA-1 を使用します。

DS コードポイント **(DSCP)** IP ヘッダーの DS フィールドに含まれていて、パケットの転送方法を指示する 6 ビットの値。

header IP ヘッダーを参照してください。

HMAC メッセージ認証を行うためのキー付きハッシュ方法。HMAC は秘密鍵認証アルゴリズムの 1 つです。HMAC は秘密共有鍵と併用して、MD5 、SHA-1 などの繰り返し暗号化のハッシュ関数で使用します。HMAC の暗号の強さは、基になるハッシュ関数のプロパティーによって異なります。

ICMP インターネット制御メッセージプロトコル (Internet Control Message Protocol)。エラーの処理や制御メッセージの交換に使用されます。

ICMP エコー要求パケット 応答を促すためにインターネット上のマシンに送信されるメッセージ。そのようなパケットは一般に “ping” パケットといわれています。

IKE インターネット鍵交換。IPsec セキュリティーアソシエーション (SA) 用の認証された鍵情報の供給を自動化します。

IP インターネットプロトコル (IP)、IPv4、IPv6 を参照してください。

IPMP グループ IP マルチパスグループ。ネットワークの可用性と利用率を向上させるために相互に入れ替え可能なものとしてシステムで扱われる、一連のネットワークインタフェースと一連のデータアドレスで構成されます。IPMP グループは、そのすべての IP インタフェースとデータアドレスも含めて、IPMP インタフェースによって表されます。

IPQoS diffserv モデル 標準に加えて、仮想 LAN に対するフローアカウンティングや 802.1 D マーカーの実装を行うソフトウェア機能。IPQoS を使用すると、IPQoS 構成ファイル内に定義したとおりに、さまざまなレベルのネットワークサービスを顧客やアプリケーションに提供できます。

IPsec IP セキュリティー。IP データグラムを保護するためのセキュリティーアーキテクチャー。

IPv4 インターネットプロトコルのバージョン 4。単に IP と呼ばれることもあります。このバージョンは 32 ビットのアドレス空間をサポートしています。

IPv6 インターネットプロトコルのバージョン 6。128 ビットのアドレス空間をサポートしています。

IP スタック TCP/IP はしばしば「スタック」と呼ばれます。データ交換のクライアントエンドとサーバーエンドですべてのデータが通過する層 (TCP、IP、場合によってはそのほかを含む) のことを意味します。

IP データグラム IP 経由で転送される情報パケット。IP データグラムはヘッダーとデータを含みます。ヘッダーにはデータグラムのソースと宛先のアドレスが含まれます。ヘッダーのその他のフィールドには、複数のデータグラムを宛先で識別し、再結合するための情報が含まれます。

IP 内 **IP** カプセル化 IP パケット内で IP パケットをトンネリングするためのメカニズム。

IP ヘッダー インターネットパケットを固有に識別する 20 バイトのデータ。ヘッダーには、パケットの送信元と送信先のアドレスが含まれています。さらに、ヘッダー内のオプションによって、新しいバイトを追加できます。

IP リンク リンク層でノード間の通信に使用される通信設備や通信メディア。リンク層とは IPv4 および IPv6 のすぐ下の層です。例としては、Ethernet (ブリッジされたものも含む) や ATM ネットワークなどがあります。1 つまたは複数の IPv4 サブネット番号またはネットワーク接頭辞が IP リンクに割り当てられます。同じサブネット番号またはネットワーク接頭辞を複数の IP リンクに割り当てることはできません。ATM LANE では、IP リンクは 1 つのエミュレートされた LAN です。ARP を使用する場合、ARP プロトコルの有効範囲は単一の IP リンクです。

MD5 デジタル署名などのメッセージ認証に使用する繰り返し暗号化のハッシュ関数。1991 年に Rivest 氏によって開発されました。

MTU 最大転送単位。リンクに転送できるサイズ (オクテット単位)。たとえば、Ethernet の MTU は 1500 オクテットです。

NAT ネットワークアドレス変換を参照してください。

Perfect Forward Secrecy (PFS) PFS では、データ伝送を保護するために使用される鍵が、追加の鍵を導き出すために使用されることはありません。さらに、データ伝送を保護するために使用される鍵のソースが、追加の鍵を導き出すために使用されることはありません。

PFS は認証された鍵交換だけに適用されます。Diffie-Hellman アルゴリズムも参照してください。

PKI Public Key Infrastructure。インターネットトランザクションに関係する各関係者の有効性を確認および承認する、デジタル署名、認証局、ほかの登録機関のシステム。

plumb する 物理インタフェース名に関連するデバイスを開く動作のこと。インタフェースが plumb されると、そのデバイスを IP プロトコルが使用できるようにストリームが設定されます。システムの現在のセッション中にインタフェースを plumb するには、`ifconfig` コマンドを使用します。

RSA デジタル署名と公開鍵暗号化システムを取得するための方法。その開発者である Rivest 氏、Shamir 氏、Adleman 氏によって 1978 年に最初に公開されました。

SA セキュリティーアソシエーション (SA)を参照してください。

SADB セキュリティーアソシエーションデータベース。暗号化鍵と暗号化アルゴリズムを指定するテーブル。鍵とアルゴリズムは、安全なデータ転送で使用されます。

SCTP 「ストリーム制御転送プロトコル」を参照してください。

SHA-1 セキュリティー保護されたハッシュアルゴリズム。メッセージ要約を作成するために 2^{64} 文字以下の長さを入力するときに操作します。SHA-1 アルゴリズムは DSA に入力されます。

smurf 攻撃 リモートロケーションから IP ブロードキャストアドレス または複数のブロードキャストアドレスに向けられた ICMP echo request パケットを使用して、深刻なネットワークの輻輳や中断を引き起こすこと。

SPD セキュリティーポリシーデータベース (SPD) を参照してください。

SPI セキュリティーパラメータインデックス (SPI) を参照してください。

TCP/IP TCP/IP (伝送制御プロトコル/インターネットプロトコル) は、インターネットの基本的な通信言語またはプロトコルです。プリベートネットワーク (イントラネットやエクストラネット) の通信プロトコルとしても使用されます。

Triple-DES Triple-Data Encryption Standard。対称鍵暗号化システムの 1 つ。Triple-DES ではキーの長さとして 168 ビットが必要です。Triple-DES を「3DES」と表記することもあります。

アドレスの移行 あるネットワークインタフェースから別のネットワークインタフェースにアドレスを移動する処理を指します。アドレスの移行は、インタフェースに障害が発生した際のフェイルオーバー、またはインタフェースが回復した際のフェイルバックの一部として実行されます。

アドレスプール Mobile IP では、ホームアドレスを必要とするモバイルノードが利用する、ホームネットワーク管理者によって指定された一連のアドレス。

インターネットプロトコル **(IP)** インターネットを介してデータをあるコンピュータから別のコンピュータに送信するための方法またはプロトコル。

エージェント通知 Mobile IP では、ホームエージェントおよび外来エージェントが、モバイルノードがリンク上に存在することを通知するために定期的に送信するメッセージ。

エージェント発見 Mobile IP では、モバイルノードが移動している場合は、自分の現在の場所および外部ネットワーク上での自分の気付アドレスを決定すること。

エニーキャストアドレス (一般的に別のノードに属する) インタフェースグループに割り当てられる IPv6 アドレス。エニーキャストアドレスに送られたパケットは、そのアドレスを持つ、プロトコルに基づき「最も近い」インタフェースに配送されます。パケットの経路制御は、経路制御プロトコルの距離測定に応じて決定されます。

エニーキャストグループ 同じエニーキャスト IPv6 アドレスからなるインタフェースグループ。IPv6 の Oracle Solaris 実装は、エニーキャストアドレスやグループの作成をサポートしていません。ただし、Oracle Solaris IPv6 ノードはトラフィックをエニーキャストグループに送信できます。

解釈ドメイン **(DOI)** データ形式や、ネットワークトラフィック交換タイプ、セキュリティー関連情報の命名規約を定義します。セキュリティー関連情報の例としては、セキュリティーポリシーや、暗号化アルゴリズム、暗号化モードなどがあります。

回復検出 障害の発生後、NIC や NIC から第 3 層の装置への経路が、正しく動作し始めたことを検出する処理。

外部ネットワーク モバイルノードのホームネットワーク以外のネットワーク。

外来エージェント モバイルノードが移動する外部ネットワーク上のルーターまたはサーバー。

鍵管理 セキュリティーアソシエーション (SA) を管理する方法。

仮想 **LAN (VLAN)** デバイス IP プロトコルスタックの Ethernet (データリンク) レベルでトラフィック転送を行うネットワークインタフェース。

仮想ネットワーク	ソフトウェアおよびハードウェアのネットワークリソースとネットワーク機能を組み合わせたもの。単一のソフトウェアエンティティとしてまとめて管理されます。「内部」仮想ネットワークは、ネットワークリソースを単一のシステムに統合したもので、「ワンボックスネットワーク (network in a box)」と呼ばれることもあります。
仮想ネットワークインタフェース **(VNIC)**	物理的なネットワークインタフェースで構成されているかどうかに関係なく、仮想ネットワーク接続を提供する擬似インタフェース。排他的 IP ゾーンなどのコンテナが VNIC 上に構成されて、仮想ネットワークを形成します。
仮想プライベートネットワーク **(VPN)**	インターネットのような公共ネットワーク内でトンネルを利用する、単独の、安全で論理的なネットワーク。
カプセル化	ヘッダーとペイロードを 1 番目のパケット内に配置し、そのパケットを 2 番目のパケットのペイロード内に配置すること。
カプセル化セキュリティーペイロード **(ESP)**	データグラムに対して認証と完全性を提供する拡張ヘッダー。ESP は、IP セキュリティーアーキテクチャー (IPsec) の 5 つのコンポーネントの 1 つです。
キーストア名	管理者がストレージ領域 (つまり、キーストア) に与える、ネットワークインタフェースカード (NIC) 上の名前。キーストア名は、「トークン」、「トークン ID」とも呼ばれます。
気付アドレス	モバイルノードの一時的アドレス。モバイルノードを外来ネットワークに接続するとき、トンネル出口として使用します。
逆方向トンネル	モバイルノードの気付アドレスで始まり、ホームエージェントで終わるトンネル。
近傍検索	接続されているリンク上にあるほかのホストをホストが特定できるようにするための IP メカニズム。
近傍通知	近傍要請メッセージに対する応答、またはデータリンク層アドレスの変更を通知するために、ノードが自発的に近傍通知メッセージを送ること。
近傍要請	近傍のリンク層アドレスを決定するために、ノードによって送信される要請。また、キャッシュされたリンク層アドレスによって近傍が到達可能であるかを確認します。
クラス	IPQoS では、似たような特性を共有するネットワークフローのグループ。クラスは、IPQoS 構成ファイル内に定義します。
クラスレスドメイン間経路制御 **(CIDR)** アドレス	ネットワーククラス (クラス A、B、C) に基づかない IPv4 アドレス形式。CIDR アドレスの長さは 32 ビットです。標準的な IPv4 10 進ドット表記形式にネットワーク接頭辞を付加したものを使用します。この接頭辞はネットワーク番号とネットワークマスクを定義します。
結果 **(outcome)**	トラフィックの計測結果に基づいて実行されるアクション。IPQoS メーターには、赤、黄、および緑の 3 種類の結果 (outcome) があり、IPQoS 構成ファイル内に定義されます。

結合テーブル	Mobile IP では、ホームアドレスを、残りの有効期間と与えられた時間を含む気付アドレスに関連付けるホームエージェント表。
検査用アドレス	IPMP グループ内の IP アドレスで、検査信号用の発信元アドレスまたは宛先アドレスとして使用する必要があり、データトラフィック用の発信元アドレスまたは宛先アドレスとして使用してはならないもの。
公開鍵暗号化	2 つの鍵を使用する暗号化システム。公開鍵はだれでも知ることができます。非公開鍵は、メッセージの受信者だけが知っています。IKE により、IPsec の公開鍵が提供されます。
再実行攻撃	IPsec では、パケットが侵入者によって捕捉されるような攻撃のこと。格納されたパケットは、あとで元のパケットを置き換えるか繰り返します。そのような攻撃を防止するために、パケットを保護している秘密鍵が存在している間、値が増加を続けるフィールドをパケットに含めることができます。
最小カプセル化	ホームエージェント、外来エージェント、およびモバイルノードによってサポートされる任意の形態の IPv4 内 IPv4 トンネリング。最小カプセル化は、IP 内 IP カプセル化よりも 8 ないし 12 バイト少ないオーバヘッドしか持ちません。
サイトローカルアドレス	単一サイト上でアドレスを指定するために使用します。
自動構成	ホストが、サイト接頭辞とローカル MAC アドレスからその IPv6 アドレスを自動的に構成する処理。
順方向トンネル	ホームエージェントから始まり、モバイルノードの気付アドレスで終わるトンネル。
障害検出	インタフェースや、インタフェースからインターネット層デバイスまでのパスが動作していないことを検出する処理。IP ネットワークマルチパス (IPMP) には、障害検出のタイプとして、リンクベース (デフォルト) と検証ベース (オプション) があります。
証明書失効リスト **(CRL)**	CA が無効とした公開鍵証明書のリスト。CRL は、IKE を使用して管理される CRL データベースに格納されます。
スタック	IP スタックを参照してください。
ステートフルパケットフィルタ	アクティブな接続の状態を監視し、そこから得た情報を使ってパケットフィルタを通過させるネットワークパケットを決めるファイアウォール。要求と応答を追跡、照合することによって、ステートフルパケットフィルタは、要求と一致しない応答を選別できます。
ステートレス自動構成	ホストがそれ自身の IPv6 アドレスを生成する処理。その生成は、ホスト自身の MAC アドレスと、ローカル IPv6 ルーターによって表明される IPv6 接頭辞を結合することによって行われます。
ストリーム制御転送プロトコル	TCP と似た方法で接続指向の通信を行う転送層プロトコル。さらに、このプロトコルは、接続のエンドポイントの 1 つが複数の IP アドレスをもつことができる複数ホーム機能をサポートします。

スプーフィング	コンピュータに不正にアクセスするために、メッセージが、信頼されるホストから来たかのように見える IP アドレスを使ってコンピュータにメッセージを送信すること。IP のなりすましを行うために、ハッカーはまず、さまざまなテクニックを使って、信頼されるホストの IP アドレスを見つけ、次にパケットヘッダーを変更します。それによって、パケットは、そのホストから来たかのように見えます。
セキュリティーアソシエーション **(SA)**	1 つのホストから 2 つめのホストにセキュリティー属性を指定するアソシエーション。
セキュリティーパラメータインデックス **(SPI)**	受信したパケットを復号化するために使用する、SADB (セキュリティーアソシエーションデータベース) 内の行を特定する整数値。
セキュリティーポリシーデータベース **(SPD)**	パケットにどのレベルの保護を適用するかを指定するデータベース。SPD は、IP トラフィックをフィルタして、パケットを破棄すべきか、検証済みとして通過させるべきか、IPsec で保護すべきかを決めます。
セレクタ	ネットワークストリームからトラフィックを選択するために、特定クラスのパケットに適用される条件を具体的に定義する要素。セレクタは、IPQoS 構成ファイル内のフィルタ句に定義します。
双方向トンネル	双方向にデータグラムを送信するトンネル。
待機	グループ内のほかの物理インタフェースに障害が発生するまでデータの伝送には使用されない物理インタフェース。
対称鍵暗号化	メッセージの送信側と受信側が 1 つの共通鍵を共有する暗号化システム。この共通鍵は、メッセージを暗号化および復号化するために使用されます。対称鍵は、IPsec での大量データ転送の暗号化に使用します。対称鍵システムの一例として DES があります。
データアドレス	データの発信元アドレスまたは宛先アドレスとして使用できる IP アドレス。データアドレスは IPMP グループの一部であり、グループ内の任意のインタフェース上でトラフィックの送受信に使用できます。さらに、IPMP グループ内の 1 つのインタフェースが機能している場合は、IPMP グループのデータアドレスのセットを継続的に使用することができます。
データグラム	IP データグラムを参照してください。
デジタル署名	送信側を一意に識別する、電子的に転送されたメッセージに添付されるデジタルコード。
デュアルスタック	IPv4 と IPv6 に関するネットワーク層の TCP/IP プロトコルスタック。このスタック以外は同一です。Oracle Solaris のインストール時に IPv6 を使用可能にすると、ホストはデュアルスタックバージョンの TCP/IP を受け取ります。
盗聴	コンピュータネットワーク上で盗聴すること。普通のテキストによるパスワードなどの情報をネットワークから自動的に選別するプログラムの一部としてしばしば使用されます。

動的再構成 **(DR)**	進行中の操作にほとんど、またはまったく影響を与えることなく、システムを実行しながらシステムを再構成できるようにする機能。Oracle からの Sun プラットフォームのすべてが、DR をサポートしているわけではありません。Oracle からの Sun プラットフォームの一部は、NIC など特定のタイプのハードウェアの DR のみをサポートする場合があります。
動的パケットフィルタ	ステートフルパケットフィルタを参照してください。
登録	モバイルノードが、ホームにないときに自分の気付アドレスを自分のホームエージェントおよび外来エージェントに登録すること。
トンネル	カプセル化される間データグラムが通過するパス。カプセル化を参照してください。
認証局 **(CA)**	デジタル署名および公開鍵と非公開鍵のペアの作成に使用するデジタル証明書を発行する、公証された第三者機関または企業。CA は、一意の証明書を付与された個人が当該の人物であることを保証します。
認証ヘッダー	IP データグラムに対し認証と完全性を提供する拡張ヘッダー。機密性は提供されません。
ネットワークアクセス識別子 **(NAI)**	user@domain 形式でモバイルノードを一意に特定するために使用します。
ネットワークアドレス変換	NAT。あるネットワークで使用されている IP アドレスを、別のネットワークで認識されている異なる IP アドレスに変換すること。必要となる大域 IP アドレスの数を抑えるために使用されます。
ネットワークインタフェースカード **(NIC)**	ネットワークへのインタフェースになる、ネットワークアダプタカード。NIC によっては、`igb` カードなど複数の物理インタフェースを装備できるものもあります。
ノード	IPv6 では、IPv6 が有効なシステムのこと。ホストかルーターかは問いません。
パケット	通信回線上で、1 単位として送られる情報の集合。IP ヘッダー や ペイロード を含みます。
パケットフィルタ	指定するパケットのファイアウォールの通過を許可するようにも許可しないようにも構成できるファイアウォール機能。
パケットヘッダー	IP ヘッダーを参照してください。
ハッシュ値	テキストの文字列から生成される数値。ハッシュ関数は、転送されるメッセージが改ざんされないようにするために使用します。一方向のハッシュ関数の例としては、MD5 と SHA-1 があります。
汎用経路制御カプセル化 **(GRE)**	ホームエージェント、外来エージェント、およびモバイルノードによってサポートされる任意の形態のトンネリング。ほかの任意の (または同じ) ネットワーク層プロトコルの配信パケット内で任意のネットワーク層プロトコルのパケットをカプセル化できるようにします。

非対称鍵暗号化	メッセージの送受信側で異なる鍵を使用してメッセージの暗号化および暗号解除を行う暗号化システム。非対称鍵を使用して、対称鍵暗号に対するセキュリティー保護されたチャネルを作成します。Diffie-Hellman アルゴリズムは、非対称鍵プロトコルの例です。対称鍵暗号化と比較してください。
ファイアウォール	組織のプライベートネットワークやイントラネットをインターネットから切り離し、外部からの進入を防止するためのデバイスまたはソフトウェア。ファイアウォールには、フィルタリングや、プロキシサーバー、NAT (ネットワークアドレス変換) などを組み込むことができます。
フィルタ	クラスの特性を IPQoS 構成ファイル内に定義するための規則セット。IPQoS システムでは、IPQoS 構成ファイル内に定義されたフィルタに適合するトラフィックフローを選択して処理します。パケットフィルタを参照してください。
フェイルオーバー	ネットワークアクセスを障害が検出されたインタフェースから正常な物理インタフェースに切り替える処理。ネットワークアクセスには、IPv4 のユニキャスト、マルチキャスト、およびブロードキャストと、IPv6 のユニキャストとマルチキャストが含まれます。
フェイルバック	ネットワークアクセスを、回復が検出されたインタフェースに戻す処理。
負荷分散	インバウンドまたはアウトバウンドのトラフィックを一連のインタフェースに分散する処理。負荷分散を使用すると、より高いスループットを達成できます。ただし、負荷分散が行われるのは、データが複数の接続を経由して複数の標識に送信される場合だけです。負荷分散には、インバウンドトラフィック用のインバウンド負荷分散とアウトバウンドトラフィック用のアウトバウンド負荷分散の 2 種類があります。
物理インタフェース	リンクへのシステムの接続。この接続は通常、デバイスドライバとネットワークインタフェースカード (NIC) として実装されます。NIC によっては、`igb` のように複数の接続点を持つものもあります。
プライベートアドレス	インターネット経由で経路制御ができない IP アドレス。プライベートアドレスは、インターネット接続を必要としない社内ネットワークのホストで使用できます。このようなアドレスはAddress Allocation for Private Internets (`http://www.ietf.org/rfc/rfc1918.txt?number=1918`) で定義され、しばしば "1918" アドレスと呼ばれます。
フローアカウンティング	IPQoS では、トラフィックフローに関する情報を蓄積、記録する処理のこと。フローアカウンティングを確立するには、`flowacct` モジュールのパラメータを IPQoS 構成ファイル内に定義します。
ブロードキャストアドレス	アドレスのホスト部分のビットがすべてゼロ (10.50.0.0) か 1 (10.50.255.255) である IPv4 ネットワークアドレス。ローカルネットワーク上のマシンからブロードキャストアドレスに送信されたパケットは、同じネットワーク上のすべてのマシンに配信されます。
プロキシサーバー	Web ブラウザなどのクライアントアプリケーションと別のサーバーの間にあるサーバー。要求をフィルタするために使用されます (たとえば、特定の Web サイトへのアクセスを防ぐ)。
プロトコルスタック	IP スタックを参照してください。

ペイロード　パケットで伝送されるデータ。ペイロードには、パケットを宛先に送るために必要なヘッダー情報は含まれません。

ホームアドレス　モバイルノードに長期間割り当てられた IP アドレス。このアドレスは、インターネットあるいは企業ネットワークに接続されたときにも変更されません。

ホームエージェント　モバイルノードのホームネットワーク上のルーターまたはサーバー。

ホームネットワーク　モバイルノードのホームアドレスのネットワーク接頭辞と一致するネットワーク接頭辞を持つネットワーク。

ホスト　パケット転送を行わないシステム。Oracle Solaris をインストールされると、システムはデフォルトでホストになります。つまり、このシステムはパケットを転送できません。通常、ホストは 1 つの物理インタフェースをもちます。ただし、複数のインタフェースをもつこともできます。

ホップ　2 つのホストを分離するルーターの数を判別するための手段。たとえば、始点ホストと終点ホストが 3 つのルーターで分離されている場合、ホストは互いに 4 ホップ離れています。

ホップ単位動作 **(Per-Hop Behavior、PHB)**　トラフィッククラスに割り当てられる優先順位。PHB は、そのクラスのフローに割り当てられる、ほかのトラフィッククラスに対する相対的な優先度を示します。

マーカー　1. diffserv アーキテクチャーおよび IPQoS のモジュールの 1 つ。パケットの転送方法を指示する値を IP パケットの DS フィールドに付けます。IPQoS 実装では、このマーカーモジュールは `dscpmk` です。

2. IPQoS 実装のモジュールの 1 つ。ユーザー優先順位の値を Ethernet データグラムの仮想 LAN タグに付けます。ユーザー優先順位の値は、VLAN デバイスを備えたネットワーク上でデータグラムが転送される方法を示します。このモジュールは `dlcosmk` と呼ばれます。

マルチキャストアドレス　特定の方法でインタフェースのグループを特定する IPv6 アドレス。マルチキャストアドレスに送信されるパケットは、グループにあるすべてのインタフェースに配信されます。IPv6 マルチキャストアドレスには、IPv4 ブロードキャストアドレスに似た機能があります。

マルチホームホスト　複数の物理インタフェースをもち、パケット転送を行わないシステム。マルチホームホストでは経路制御プロトコルを実行できます。

メーター　特定クラスのトラフィックフローの速度を測定する diffserv アーキテクチャーのモジュール。IPQoS 実装には、`tokenmt` および `tswtclmt` という 2 つのメーターがあります。

メッセージ認証コード **(MAC)**　データの整合性を保証し、データの出所を明らかにするコード。MAC は盗聴行為には対応できません。

モバイルノード　自分の IP ホームアドレスを使用して既存の通信をすべて維持しながら、接続点をネットワーク間で変更するホストまたはルーター。

モビリティーエージェント	ホームエージェントまたは外来エージェント。
モビリティー結合	ホームアドレスと気付アドレスとを関連付けます。その関連付けの残りの有効期間も含みます。
モビリティーセキュリティーアソシエーション	認証アルゴリズムのような、ノード間のセキュリティー対策の集合。2 つのノード間で交換されるモバイル IP プロトコルメッセージに適用されます。
ユーザー優先順位	サービスクラスのマークを実装する 3 ビットの値。VLAN デバイスのネットワーク上で Ethernet データグラムが転送される方法を定義します。
ユニキャストアドレス	IPv6 が有効なノードの単一インタフェースを識別する IPv6 アドレス。ユニキャストアドレスは、サイト接頭辞や、サブネット ID、インタフェース ID などからなります。
リダイレクト	特定の終点に到達するために、ホストに対して最適な最初のホップノードを、ルーターが通知すること。
リンク - ローカル・アドレス	IPv6 では、自動アドレス構成などのために、単一リンク上でアドレスを指定するために使用することを表します。デフォルトでは、リンク - ローカル・アドレスはシステムの MAC アドレスから作成されます。
リンク層	IPv4/IPv6 のすぐ下の層。
ルーター	複数のインタフェースを通常もち、経路制御プロトコルを実行し、パケットを転送するシステム。システムが PPP リンクのエンドポイントである場合は、ルーターとしてのインタフェースを 1 つだけもつようなシステムを構成できます。
ルーター広告	ルーターが、各種のリンクパラメータおよびインターネットパラメータと共に、その存在を定期的にあるいはルーター要請メッセージに応じて通知すること。
ルーター発見	ホストが、接続されているリンク上にあるルーターを特定すること。
ルーター要請	ホストがルーターに対し、次に予定されている時刻ではなく、ただちにルーター広告メッセージを送信するように要求すること。
ローカル使用アドレス	ローカルの経路制御可能な範囲だけを対象とするユニキャストアドレス (サブネット内またはネットワーク内)。また、ローカルまたはグローバルな一意の範囲を対象とすることもできます。

索引

C

D

E

F

K

L

M

N

O

P

Q

R

S

T

U

V

W

き

く

け

さ

す

せ

そ

た

つ

て

と

な

に

ね

ろ