Oracle® Solaris 11.1 ネットワークの構成と管理

Part No: E36457–02
2012 年 10 月

ORACLE

121210@25097

目次

はじめに

『Oracle Solaris 11.1 ネットワークの構成と管理』へようこそ。このドキュメントは、Oracle Solaris 11.1 ネットワークの確立に関するシリーズの一部で、Oracle Solaris ネットワークを構成するための基本的なトピックおよび手順について説明しています。このドキュメントの記述は、Oracle Solaris 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、またはネットワークで必要となる任意のネットワークソフトウェアを構成する準備を整えるようにしてください。

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を管理する責任がある人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。

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、このドキュメント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.`

表 P–1 表記上の規則 *(続き)*

字体	説明	例
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	プレースホルダ: 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 のデフォルト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。コマンド例に示されるデフォルトのシステムプロンプトは、Oracle Solaris のリリースによって異なります。

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#`

◆ ◆ ◆ 第 1 章

1 ネットワーク配備の計画

この章では、ネットワーク設定を計画するときのさまざまな考慮点について簡単に説明します。これらの問題を考慮しておけば、費用対効果の高い組織化された方法でネットワークを配備しやすくなります。ネットワーク計画の詳細はこのドキュメントの範囲外です。一般的な指示のみが提供されます。

このドキュメントは、読者がネットワークの基本的な概念や用語に習熟していることを前提にしています。Oracle Solaris 11 で TCP/IP プロトコル群を実装する方法については、『Oracle Solaris 11 ネットワーキングの紹介』の「Oracle Solaris でのネットワークスタック」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、ネットワーク構成を計画するためのさまざまなタスクの一覧を示します。

タスク	説明	参照先
計画しているネットワークトポロジのハードウェア要件を識別します。	ユーザーのネットワークサイトで必要になる装置のタイプを決定します。	12 ページの「ネットワークハードウェアの決定」 特定の種類の機器については、機器メーカーのドキュメントを参照してください。

タスク	説明	参照先
使用する IP アドレスのタイプを決定し、登録済みの IP アドレスを取得します。	IPv4 のみのネットワーク、IPv6 のみのネットワーク、またはその両方のタイプの IP アドレスを使用するネットワークのいずれを配備するのかを選択します。インターネット上のパブリックネットワークと通信できるように、一意の IP アドレスを取得します。	13 ページの「ネットワークの IP アドレス指定形式の決定」 15 ページの「ネットワークの IP 番号の取得」
ネットワーク内のホストを識別する名付けスキームと、使用するネームサービスを決定します。	ネットワーク上のシステムに割り当てる名前のリストを作成し、NIS、LDAP、DNS、またはローカルの /etc ディレクトリ内のネットワークデータベースのいずれを使用するかを決定します。	16 ページの「ホスト名の管理」 16 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」
必要であれば、管理上の区分を確立し、サブネットの方針を設計します。	管理作業を分担するための区分を提供するために、サイトのネットワークをサブネットに分割する必要があるかどうかを決定します	18 ページの「サブネットの使用」
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。	18 ページの「ネットワーク上でのルーターの計画」
全体のネットワーク構成スキームで仮想ネットワークを作成するかどうかを決定します。	ネットワークのハードウェア設置面積を減らせるよう、システム内に仮想ネットワークを作成しなければいけない場合もあります。	『Oracle Solaris 11.1 での仮想ネットワークの使用』

ネットワークハードウェアの決定

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。

ハードウェアに関して下す必要のある計画上の決定のいくつかを、次に示します。

- 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

- ネットワークでサポート可能なホストシステムのタイプと数 (必要になる可能性のあるサーバーも含む)
- それらのシステムに装着するネットワークデバイス
- Ethernet など、使用するネットワークメディアのタイプ
- 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

注 – ルーターがどのように機能するかについては、18 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。ブリッジの概要については、『Oracle Solaris 11.1 ネットワークパフォーマンスの管理』の「ブリッジングの概要」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

ネットワークのアドレス指定スキームを計画するときには、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

IP アドレスの大まかな種類を次に示します。

IPv4 アドレス

これらの 32 ビットアドレスは、TCP/IP の元の IP アドレス指定形式です。その後、IETF は、IPv4 アドレスの不足や世界的なインターネットルーティングテーブルの容量不足に対する短期的および中期的な対応策として、クラスレスドメイン間ルーティング *(CIDR)* アドレスを開発しました。

詳細は、次のリソースを参照してください。

- Internet Protocol DARPA Internet Program Protocol Specification (`http://tools.ietf.org/html/rfc791`)
- Classless Inter-domain Routing (CIDR): The Internet Address Assignment and Aggregation Plan (`http://tools.ietf.org/html/rfc4632`)

次の表に、サブネットを CIDR 表記と小数点付き 10 進数形式の両方で示します。

表 1–1　CIDR 接頭辞と 10 進数での表現

CIDR ネットワーク接頭辞	ドット付き 10 進数でのサブネット表現	使用可能な IP アドレス
/19	255.255.224.0	8,192
/20	255.255.240.0	4,096
/21	255.255.248.0	2,048
/22	255.255.252.0	1,024
/23	255.255.254.0	512
/24	255.255.255.0	256
/25	255.255.255.128	128
/26	255.255.255.192	64
/27	255.255.255.224	32

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。詳細は、『Oracle Solaris 11.1 での DHCP の作業』の第 1 章「DHCP について (概要)」を参照してください。

IPv6 アドレス

128 ビットの IPv6 アドレスは、IPv4 で使用可能なアドレス空間よりも広大なアドレス空間を提供します。IPv6 アドレスは CIDR 形式の IPv4 アドレスと同様にクラスレスであり、サイトのネットワークを定義するアドレス部分を指定するために接頭辞を使用します。IPv6 アドレス指定についての詳細は、Internet Protocol, Version 6 (IPv6) Specification (`http://tools.ietf.org/html/rfc2460`)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。これらのプライベートアドレスは、プラ

イベートネットワーク内のネットワークトラフィックに対して使用されます。これらのアドレスはドキュメント内でも使用されます。

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

IPv6 アドレスの場合、`2001:db8::/32` という接頭辞は、このドキュメントの例だけで使用される特別な IPv6 接頭辞です。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。

プライベートネットワークがインターネット上の外部ネットワークと通信できるようにするには、ネットワーク用の登録済み IP 番号を適切な組織から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。

ネットワーク上のエンティティーへの名前付け

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、ホスト名を使用すれば、IP アドレスの場合よりも容易にシステムを識別できます。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

ネットワークを構成するシステム用の名付けスキームを計画します。サーバーとして機能し、複数の NIC を持つシステムでは、その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たホスト名を少なくとも 1 つ提供する必要があります。

ネットワーク上の 2 つのマシンが両方とも同じホスト名を持つことはできません。したがって、各ホスト名はそれぞれのシステムに固有でなければいけません。ただし、一意の名前が割り当てられたホストまたはシステムが複数の 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 では次の 3 種類のネームサービスから選択できます: ローカルファイル、NIS、および DNS。ネームサービスは、ホスト名、IP アドレス、Ethernet アドレスなど、ネットワーク上のマシンに関する重要な情報を維持します。また、ネームサービスのほかに、あるい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 のネームサービスの概要については、『Oracle Solaris 11.1 でのネームサービスおよびディレクトリサービスの作業』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。

OS のインストール時に、サーバー、クライアント、またはスタンドアロンシステムのホスト名と IP アドレスを入力します。インストールプログラムはこの情報を、ネットワークへのサービス提供時にネットワークサービスによって使用される `hosts` データベース内に追加します。

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。

ネームサービスとしては、次のいずれかを選択できます。

- NIS または DNS – NIS および DNS ネームサービスは、ネットワーク上のいくつかのサーバー上でネットワークデータベースを維持します。『Oracle Solaris 11.1 でのネームサービスおよびディレクトリサービスの作業』では、これらのネームサービスについて説明し、データベースの構成方法について解説しています。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。
- ローカルファイル – NIS、LDAP、または DNS を実装しない場合、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では、そのホストとルーターが管理ドメインの階層に編成されます。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。ドメイン名がほかの関係を持たない場合、組織はそのドメイン名を、`.com`、`.org`、`.edu`、`.gov` など、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。

サブネットの使用

サブネットの使用は、サイズや制御の問題を解決するための管理区分の必要性と関係しています。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。管理区分を作成してサブネットを使用すれば、複雑なネットワークの管理が容易になります。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。

- ネットワークのサイズ

  サブネットは、区分の場所が地理的に広範囲に分散している比較的小規模なネットワークでも役立ちます。

- ユーザーのグループが共有する共通のニーズ

  たとえば、単一の建物内のみに制限された、比較的少数のマシンをサポートするネットワークがあるとします。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タスクについては、35 ページの「ネットワーク上のコンポーネントシステムの構成」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、66 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

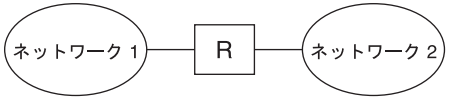
ネットワークトポロジは、ネットワークの組み合わせ方を定義します。ルーターは、ネットワークを相互に接続するエンティティーです。ルーターは、複数のネットワークインタフェースを持ち、IP 転送を実行するマシンです。ただし、43 ページの「IPv4 ルーターの構成」で説明されているとおりに正しく構成されるまで、システムはルーターとして機能できません。

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 1–1 基本的なネットワークトポロジ

1 台のルーターによって接続された 2 つのネットワーク

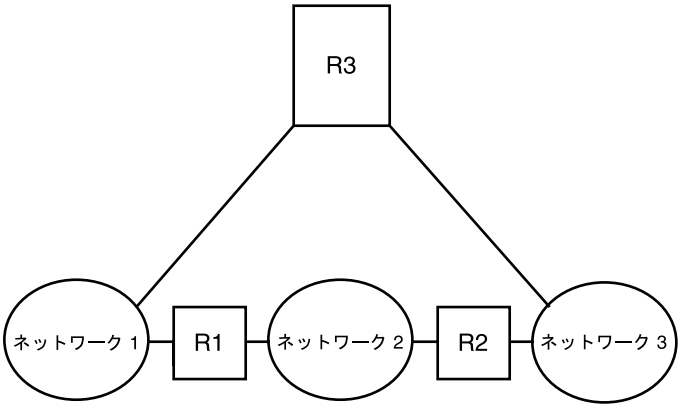


2 台のルーターによって接続された 3 つのネットワーク

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、より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1–2 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



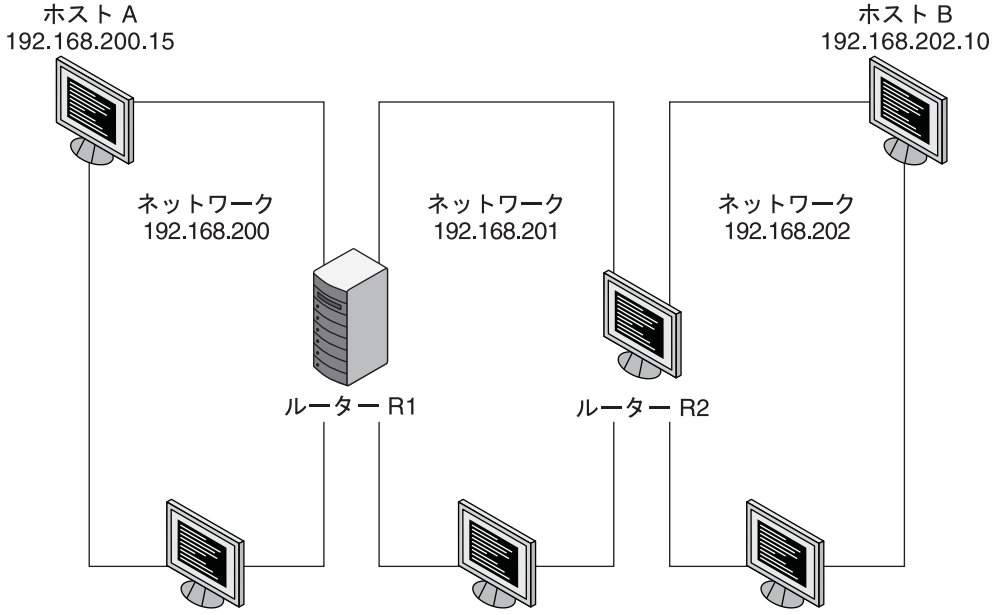
ルーターがどのようにパケットを転送するか

パケットヘッダーの一部である受信側の IP アドレスによって、パケットの経路制御方法が決まります。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、43 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 1–3　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。

仮想ネットワークの配備

この Oracle Solaris リリースでは、ゾーンと仮想ネットワークカード (VNIC) を構成することで単一のネットワーク内に仮想ネットワークを作成することがサポートされています。VNIC とは、物理 NIC 上で作成されるネットワークインタフェースのことです。ゾーンと VNIC の組み合わせは、多数の物理システムを含む大規模なデータセンターを少数のシステムに統合するための効果的な手段となります。仮想ネットワーキングの詳細は、『Oracle Solaris 11.1 での仮想ネットワークの使用』を参照してください。

◆ ◆ ◆ 第 2 章

IPv6 アドレス使用時の考慮点

この章では、第 1 章「ネットワーク配備の計画」に対する補足情報として、ネットワーク上で IPv6 アドレスを使用することにした場合の追加の考慮点について説明します。

IPv4 アドレスのほかに IPv6 アドレスも使用することを計画している場合、現在の ISP が両方のアドレスタイプをサポートしていることを確認してください。それ以外の場合、IPv6 アドレスをサポートする別の ISP を探す必要があります。

IPv6 の概念の概要については、Internet Protocol, Version 6 (IPv6) Specification (`http://www.ietf.org/rfc/rfc2460.txt`)を参照してください。

IPv6 の計画 (タスクマップ)

次の表に、ネットワーク上での IPv6 の実装を計画する場合のさまざまな考慮点の一覧を示します。

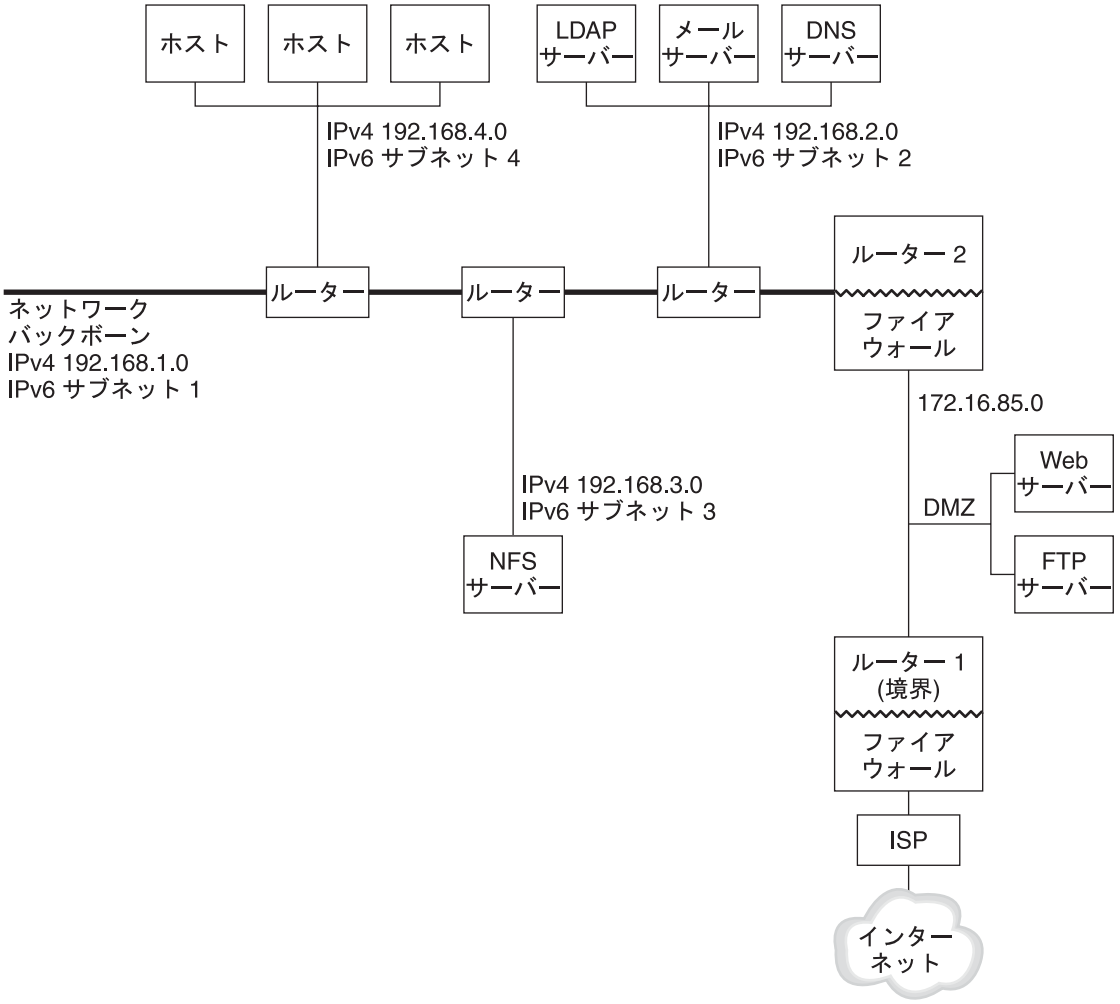
タスク	説明	手順
IPv6 をサポートするようにハードウェアを準備します。	IPv6 をサポートできるようにハードウェアをアップグレードします。	26 ページの「IPv6 のハードウェアサポートの確認」
アプリケーションが IPv6 をサポートすることを確認します。	使用するアプリケーションが IPv6 環境で動作できることを確認します。	29 ページの「IPv6 をサポートするようにネットワークサービスを構成する」
トンネルの使用について計画します。	ほかのサブネットまたは外部ネットワークへのトンネルを実行するルーターを判断します。	31 ページの「ネットワークでのトンネル使用の計画」

タスク	説明	手順
ネットワークのセキュリティー保護を行う方法を計画し、IPv6 セキュリティーポリシーを開発します。	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。 IP フィルタ、IP セキュリティーアーキテクチャー (IPsec)、インターネット鍵交換 (IKE)、およびこのリリースのその他のセキュリティー機能を使用するなど、セキュリティーの実装方法を決定します。	32 ページの「IPv6 実装のセキュリティーについて」 『Oracle Solaris 11.1 でのネットワークのセキュリティー保護』
ネットワーク上のシステム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。この手順には必要に応じて、ネットワークのサイト接頭辞の取得や IPv6 サブネットの計画も含まれます。	27 ページの「ノードの IPv6 アドレス指定計画の立案」

IPv6 ネットワークトポロジのシナリオ

IPv6 は通常、次の図に示すような IPv4 も使用されている混在ネットワークトポロジで使用されます。この図は、後続の各セクションでの IPv6 構成タスクの説明で、参照として使用されます。

図 2–1 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 2–1 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。

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。ユーザーは、ネットワーク上での IPv6 の配備中や配備後に IPv4 関連の処理を正常に実行できます。IPv4 をすでに使用している動作中のネットワーク上で IPv6 を配備するときは、進行中の処理の邪魔にならないようにしてください。

次の各セクションでは、IPv6 の実装を準備するときに考慮する必要のある領域について説明します。

IPv6 のハードウェアサポートの確認

次のクラスのハードウェアについては、メーカーのドキュメントで IPv6 の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 – このドキュメントで説明するすべての手順では、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、『Troubleshooting Network Issues』の「IPv4 Router Cannot Be Upgraded to IPv6」を参照してください。

IPv6 サーバーの NIC ごとに、近傍検索プロトコルを使用して ID を自動的に取得するのではなく、IPv6 アドレスのインタフェース ID 部分を手動で構成します。そうすれば、NIC が交換されたときに、その交換後の NIC にも同じインタフェース ID を適用できます。近傍検索プロトコルによって異なる ID が自動生成されると、サーバーで予期しない動作が発生する可能性があります。

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。

- 27 ページの「サイト接頭辞の取得」
- 27 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。

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。詳細は、『Troubleshooting Network Issues』の「Current ISP Does Not Support IPv6」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。

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 2–1 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 2–1 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)。

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 2–1 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`

IPv4 サブネット接頭辞	等価な IPv6 サブネット接頭辞
192.168.4.0/24	2001:db8:3c4d:4::/64

IPv6 をサポートするようにネットワークサービスを構成する

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 リリースまたは 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。詳細は、『Troubleshooting Network Issues』の「Problems After Upgrading Services to IPv6」を参照してください。

▼ 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ドキュメント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ドキュメントや関連するドキュメント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、32 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。

- メール

  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、30 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。詳細は、『Oracle Solaris 11.1 での IP サービス品質の管理』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 用の DNS サポートに関係する詳細情報については、『Oracle Solaris 11.1 でのネームサービスおよびディレクトリサービスの作業』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。

ネットワークでのトンネル使用の計画

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 2–1 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。

トンネルの構成手順については、112 ページの「トンネルの構成 (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情報については、103 ページの「IP トンネルの概要」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。詳細は、『Oracle Solaris 11.1 でのネットワークのセキュリティー保護』の第 6 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。詳細は、『Oracle Solaris 11.1 でのネットワークのセキュリティー保護』の第 9 章「インターネット鍵交換 (概要)」を参照してください。

◆ ◆ ◆ 第 3 章

IPv4 ネットワークの構成

ネットワーク構成は、ハードウェアを組み立てたあとに、TCP/IP プロトコルを実装するデーモン、ファイル、およびサービスを構成するという 2 段階で進みます。

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

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。IPv6 ネットワークに固有のタスクについては、第 4 章「ネットワークでの IPv6 の有効化」に記載されています。

注 – TCP/IP を構成する前に、第 1 章「ネットワーク配備の計画」に記載されている各種の計画タスクを確認してください。IPv6 アドレスを使用することを計画している場合、第 2 章「IPv6 アドレス使用時の考慮点」も参照してください。

この章では、次の内容について説明します。

ネットワーク構成 (タスクマップ)

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	手順
システムの IP インタフェースを構成します。	システムの IP インタフェースに IP アドレスを割り当てます。	『Oracle Solaris 11.1 での固定ネットワーク構成を使用したシステムの接続』の「IP インタフェースを構成する方法」
システムをローカルファイルモード用に構成します	システムの /etc ディレクトリにある特定の構成ファイルを編集し、nis/domain SMF サービスを構成します。	40 ページの「システムをローカルファイルモード用に構成する方法」
ネットワーク構成サーバーをセットアップします	in.tftp デーモンを有効にし、システムの /etc ディレクトリにあるほかの構成ファイルを編集します。	42 ページの「ネットワーク構成サーバーの設定方法」
システムをネットワーククライアントモード用に構成します	システムの /etc ディレクトリにある構成ファイルを編集します。	41 ページの「システムをネットワーククライアントモード用に構成する方法」
ネットワーククライアントのルーティング戦略を指定します	静的ルーティングまたは動的ルーティングのいずれかを使用するようにシステムを構成します。	52 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 54 ページの「単一インタフェースシステムで動的ルーティングを有効にする方法」。

ネットワーク構成を開始する前に

この Oracle Solaris リリースでは、システムのネットワーク構成はアクティブなネットワーク構成プロファイル *(NCP)* によって管理されます。アクティブな NCP がリアクティブ (automatic NCP など) の場合、システムのネットワーク構成は自動になります。アクティブな NCP が DefaultFixed の場合、システムのネットワーク構成モードは固定になります。リアクティブなネットワーク構成を備えたシステムは、固定ネットワーク構成を備えたシステムとは異なった動作をします。

実行する構成はすべてアクティブな NCP に適用されます。このため、どの構成手順を実行する前にも、まずどの NCP が正常にアクティブになっているかを知っておく必要があります。したがって、構成手順が完了したあと、システムは予想どおりに動作します。システム上でアクティブになっている NCP を確認するには、次のコマンドを入力します。

```
# netadm list
TYPE        PROFILE        STATE
ncp         DefaultFixed   online
ncp         Automatic      disabled
loc         Automatic      offline
```

```
loc         NoNet          offline
loc         User           offline
loc         DefaultFixed   online
```

ステータスがオンラインとしてリストされているプロファイルは、システム上でアクティブな NCP です。

システム上の NCP の詳細を表示するには、`netadm` コマンドで `-x` オプションを使用します。

```
netadm list -x
TYPE        PROFILE        STATE          AUXILIARY STATE
ncp         DefaultFixed   online         active
ncp         Automatic      disabled       disabled by administrator
loc         Automatic      offline        conditions for activation are unmet
loc         NoNet          offline        conditions for activation are unmet
loc         User           offline        conditions for activation are unmet
loc         DefaultFixed   online         active
```

プロファイルの種類を切り替える (リアクティブなプロファイルから固定プロファイルに切り替えるなど) には、次のコマンドを入力します。

```
# netadm enable -p ncp NCP-name
```

ここで、*NCP-name* は NCP タイプの名前です。

プロファイル管理によるネットワーク構成の概要については、『Oracle Solaris 11 ネットワーキングの紹介』の「ネットワーク構成プロファイル」を参照してください。NCP の詳細は、『Oracle Solaris 11.1 でのリアクティブネットワーク構成を使用したシステムの接続』を参照してください。

ネットワーク上のコンポーネントシステムの構成

ネットワークシステムを構成するときは、次の構成情報が必要です。

- 各システムのホスト名。
- 各システムの IP アドレスとネットマスク。ネットワークがサブネットに分割されている場合、各サブネットのシステムに適用するサブネット番号と IP アドレススキーマが必要です。それぞれのネットマスクも含みます。
- 各システムが属しているドメイン名。
- 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。Oracle Solaris でサポートされているルーターの詳細と、ルーティングプロトコルの一覧については、129 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。

注 – Oracle Solaris のインストール中にネットワークを構成できます。手順については、『Oracle Solaris 11.1 システムのインストール』を参照してください。

このドキュメントの手順では、OS をインストールしたあとにネットワークを構成することを想定しています。

以降のセクションの図 3–1 を、ネットワークのコンポーネントシステムを構成するための参照情報として使用してください。

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。

図 3–1　複数の IPv4 ルーターを備えた自律システム

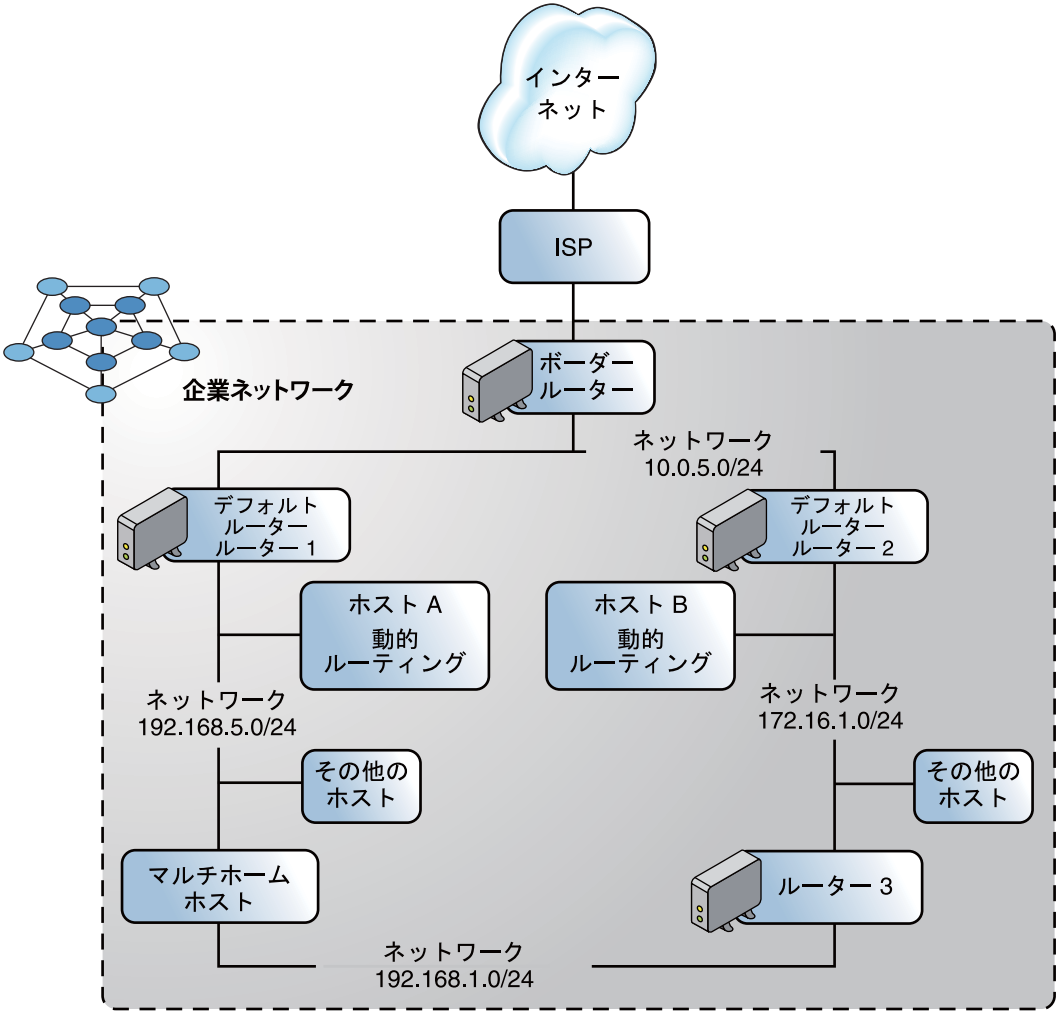


図 3–1 は、3 つのローカルネットワーク `10.0.5.0`、`172.16.1.0`、および `192.168.5.0` に分割された AS を示しています。ネットワークは次の種類のシステムで構成されています。

- ルーターはルーティングプロトコルを使用して、ローカルネットワーク内で、または外部ネットワークに対して、ネットワークパケットを発信元から着信先に伝送またはルーティングする方法を管理します。Oracle Solaris でサポートされているルーティングプロトコルについては、130 ページの「Oracle Solaris のルーティングプロトコルの表」を参照してください。

  ルーターは次のように分類されます。

- ボーダールーターは、10.0.5.0 などのローカルネットワークを外部のサービスプロバイダに接続します。
- デフォルトルーターは、ローカルネットワーク内のパケットルーティングを管理し、それ自体にいくつかのローカルネットワークを含めることができます。たとえば、図 3–1 では、ルーター 1 は 192.168.5 のデフォルトルーターとして機能します。同時に、ルーター 1 は 10.0.5.0 の内部ネットワークにも接続されています。ルーター 2 のインタフェースは 10.0.5.0 および 172.16.1.0 の内部ネットワークに接続しています。
- パケット転送ルーターは、内部ネットワーク間でパケットを転送しますが、ルーティングプロトコルは実行しません。図 3–1 で、ルーター 3 はパケット転送ルーターで、172.16.1 および 192.168.5 ネットワークに接続されています。

- クライアントシステム
  - マルチホームシステム、つまり複数の NIC を持つシステム。Oracle Solaris では、これらのシステムはデフォルトで、同じネットワークセグメントの別のシステムに対してパケットを転送できます。
  - 単一インタフェースシステムでは、パケットの転送と受信の両方の構成情報をローカルルーターに依存しています。

システム構成モードの設定

このセクションでは、ローカルファイルモードまたはネットワーククライアントモードのいずれかで動作するシステムを設定する手順について説明します。ローカルファイルモードで動作するときは、システムはローカルディレクトリにあるファイルからすべての TCP/IP 構成情報を取得します。ネットワーククライアントモードでは、構成情報はリモートネットワーク構成サーバーによって、ネットワーク内のすべてのシステムに提供されます。

一般的に、ネットワーク内の次のようなサーバーはローカルファイルモードで動作します。

- 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
- ルーター

クライアントはいずれのモードでも動作できます。したがって、ネットワーク内では次の図に示すように、これらのモードを組み合わせて、さまざまなシステムを構成できます。

図 3–2 IPv4 ネットワークトポロジに属するシステムのシナリオ

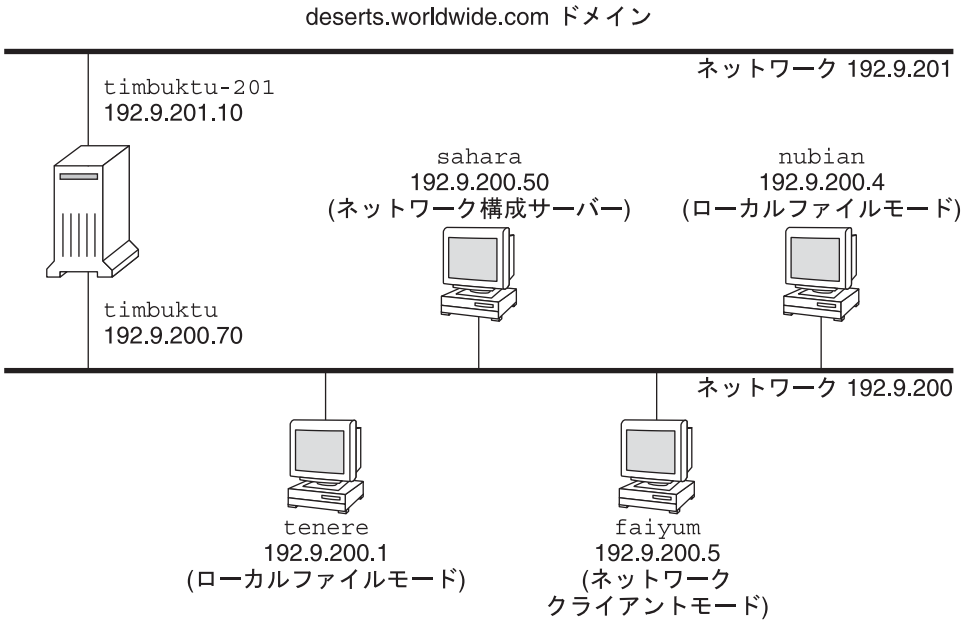


図 3–2 は、192.9.200 ネットワーク内のシステムを示しています。

- すべてのシステムは、組織ドメイン deserts.worldwide.com に属しています。
- sahara は構成サーバーです。これはサーバーとしてローカルファイルモードで動作し、TCP/IP 構成情報はシステムのローカルディスクから取得します。

注 – クライアントがネットワーククライアントモードで動作するように構成する場合、これらのクライアントに構成情報を提供するネットワーク構成サーバーを少なくとも 1 つ構成する必要があります。

- tenere、nubian、および faiyum はネットワーク内のクライアントです。tenere および nubian はローカルファイルモードで動作します。faiyum のローカルディスクに関係なく、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するように構成されています。
- timbuktu はルーターとして構成されているため、ローカルファイルモードで動作します。このシステムには 2 つの NIC が組み込まれ、それぞれ固有の IP インタフェースが構成されています。第 1 の IP インタフェースは timbuktu という名前で、ネットワーク 192.9.200 に接続します。第 2 の IP インタフェースは timbuktu-201 という名前で、ネットワーク 192.9.201 に接続します。

▼ システムをローカルファイルモード用に構成する方法

システムをローカルファイルモードで動作するように構成するには、この手順を使用します。

1. システムの IP インタフェースに、割り当て済み IP アドレスを構成します。

   手順については、『Oracle Solaris 11.1 での固定ネットワーク構成を使用したシステムの接続』の「IP インタフェースを構成する方法」を参照してください。

2. **/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。

3. **/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。

   このファイルは、デフォルトルーターの名前とルーターの IP アドレスも含む必要があります。

   a. (オプション) インストール後にシステム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。

   b. (オプション) **/usr** ファイルシステムが NFS マウントされている場合、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。

4. システムの完全修飾ドメインを **nis/domain** SMF サービスのプロパティーとして指定します。

   たとえば、次のように `nis/domain` SMF サービスの `domainname` プロパティーの値として `deserts.worldwide.com` を指定します。

   # **domainname** *domainname*

   このステップは永続的な変更をもたらします。

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

6. 該当する場合は、ネットマスクの情報を追加します。

   注 – DHCP サービスを使用中の場合、このステップをスキップします。

   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。

      エントリを作成するには、*network-number netmask* の形式を使用します。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 進ドット表記は、表 1–1 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スイッチの SMF プロパティー内のネットマスクの検索順序を変更したあとに、インスタンスをリフレッシュします。

```
# svccfg -s name-service/switch setprop config/host = astring: '"files nis"'
# svccfg -s name-service/switch:default refresh
```

7 システムをリブートします。

▼ システムをネットワーククライアントモード用に構成する方法

ネットワーククライアントモードに構成する各ホスト上で、次の手順を実行します。

始める前に ネットワーククライアントは、各自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、あるシステム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に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。

1 管理者になります。

詳細は、『Oracle Solaris 11.1 の管理: セキュリティーサービス』の「割り当てられている管理権限を使用する方法」を参照してください。

2 システムの IP インタフェースに、割り当て済み IP アドレスを構成します。

手順については、『Oracle Solaris 11.1 での固定ネットワーク構成を使用したシステムの接続』の「IP インタフェースを構成する方法」を参照してください。

3 **/etc/inet/hosts** ファイルに **localhost** という名前とループバックネットワークインタフェースの IP アドレスのみ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
```

4 **nis/domain** SMF サービスの **domainname** プロパティーに割り当て済みの値があれば削除します。

```
# domainname ''
```

このステップは永続的な変更をもたらします。

5 クライアントの **name-service/switch** サービスの検索パスが、ネットワークに対する同じサービス要件を反映していることを確認します。

▼ 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およびブートサーバーの設定の情報については、『Oracle Solaris 11.1 システムのインストール』を参照してください。

1 管理者になります。

詳細は、『Oracle Solaris 11.1 の管理: セキュリティーサービス』の「割り当てられている管理権限を使用する方法」を参照してください。

2 次のようにして **in.tftpd** デーモンをオンに設定します。

a. 指定されたネットワーク構成サーバーのルート (/) ディレクトリに移動します。

b. **/tftpboot** ディレクトリを作成します。

```
# 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。

c. 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
```

3 **/etc/inetd.conf** ファイルに **tftp** 行を追加します。

行は次のようになるはずです。

```
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。

4 **/etc/hosts** データベースに、ネットワーク上のすべてのクライアントのホスト名および **IP** アドレスを追加します。

5 **/etc/ethers** データベースに、ネットワーククライアントモードで動作するネットワーク上の各システムのエントリを作成します。

このデータベース内のエントリは、次の形式を使用します。

MAC Address *host name* *#comment*

詳細は、`ethers(4)` のマニュアルページを参照してください。

6 **/etc/bootparams** データベースに、ネットワーククライアントモードで動作するネットワーク上の各システムのエントリを作成します。

このデータベースの編集については、`bootparams(4)` のマニュアルページを参照してください。

7 **/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、結果となるサービスを使用可能にします。

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

8 **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 11.1 でのサービスと障害の管理』の第 1 章「サービスの管理 (概要)」を参照してください。

IPv4 ルーターの構成

ルーターは 2 つ以上のネットワーク間のインタフェースを提供します。したがって、ルーターの物理ネットワークインタフェースに、固有の名前と 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。『Oracle Solaris 11.1 での UUCP および PPP を使用したシリアルネットワークの管理』の「ダイアルアップ PPP リンクの計画」で説明しているように、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。

▼ 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　ルーターを物理的にネットワークに取り付けあとに、40 ページの「システムをローカルファイルモード用に構成する方法」の説明に従って、ルーターがローカルファイルモードで動作するように構成します。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. 管理者になります。

   詳細は、『Oracle Solaris 11.1 の管理: セキュリティーサービス』の「割り当てられている管理権限を使用する方法」を参照してください。

2. システム上の **NIC** に対して **IP** インタフェースを構成します。

   IP インタフェースを構成する詳しいステップについては、『Oracle Solaris 11.1 での固定ネットワーク構成を使用したシステムの接続』の「IP インタフェースを構成する方法」を参照してください。

   システムがパケットを経路指定するネットワークの IP アドレスを、各 IP インタフェースに構成するようにします。したがって、システムで `192.168.5.0` および `10.0.5.0` のネットワークに対応する場合、ネットワークごとに 1 つの NIC を構成する必要があります。

   注意 – IPv4 ルーターで DHCP を使用するように構成する場合、DHCP 管理について十分な知識を持っておく必要があります。

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

   たとえば、ルーター 1 の 2 つのインタフェースに割り当てた名前を、それぞれ `krakatoa` および `krakatoa-1` とします。`/etc/inet/hosts` ファイルのエントリは次のようになります。

   ```
   192.168.5.1      krakatoa        #interface for network 192.168.5.0
   10.0.5.1         krakatoa-1      #interface for network 10.0.5.0
   ```

4. このルーターがローカルファイルモードで動作するように構成するための残りのステップを実行します。

   40 ページの「システムをローカルファイルモード用に構成する方法」を参照してください。

5. サブネットに分割されたネットワークにルーターが接続されている場合、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。

   たとえば、`192.168.5.0` などの従来の IPv4 アドレス表記法の場合は、次のように入力します。

   ```
   192.168.5.0    255.255.255.0
   ```

6. ルーターで **IPv4** パケット転送を使用可能にします。

   ```
   # ipadm set-prop -p forwarding=on ipv4
   ```

7 (任意)ルーティングプロトコルを起動する。

次のいずれかのコマンド構文を使用します。

- # **routeadm -e ipv4-routing -u**
- # **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。

ルーティングプロトコルを開始するときに、ルーティングデーモン /usr/sbin/in.routed は自動的にルーティングテーブルを更新します。このプロセスのことを動的ルーティングと呼びます。ルーティングの種類の詳細は、46 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。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 3–1 ネットワークのデフォルトルーターを構成する

この例は、図 3–1 に基づきます。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 172.16.1.0、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 2 が 172.16.1.0 ネットワークのデフォルトルーターになるように構成する方法を示します。またこの例では、40 ページの「システムをローカルファイルモード用に構成する方法」に説明されているように、ルーター 2 がローカルファイルモードで動作するように構成されていることを想定しています。

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。

```
# dladm show-link
LINK     CLASS     MTU     STATE   BRIDGE   OVER
net0     phys      1500    up      --       --
net1     phys      1500    up      --       --
net2     phys      1500    up      --       --
# ipadm show-addr
ADDROBJ           TYPE     STATE        ADDR
lo0/v4            static   ok           127.0.0.1/8
net0/v4           static   ok           172.16.1.10/24
```

net0 だけが IP アドレスで構成されています。ルーター 2 をデフォルトルーターにするには、net1 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。

```
# ipadm create-ip net1
# ipadm create-addr -a 10.0.5.10/24 net1
# ipadm show-addr
ADDROBJ           TYPE     STATE        ADDR
lo0/v4            static   ok           127.0.0.1/8
net0/v4           static   ok           172.16.1.10/24
net1/v4           static   ok           10.0.5.10/24
```

次に、新たに構成したインタフェース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更新します。

```
# vi /etc/inet/hosts
127.0.0.1       localhost
172.16.1.10        router2        #interface for network 172.16.1
10.0.5.10          router2-out    #interface for network 10.0.5
# vi /etc/inet/netmasks
172.16.1.0    255.255.255.0
10.0.5.0      255.255.255.0
```

最後に、パケット転送と `in.routed` ルーティングデーモンを有効にします。

```
# ipadm set-prop -p forwarding=on ipv4
# 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 `172.16.1.0` 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。

- `172.16.1.0` ネットワークの各ホストを変更して、ホストがルーティング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、52 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、46 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がルーティングテーブルを管理します。ルーティングテーブルには、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、発信パケットを受け取り、それらをローカルネットワークの 1 ホップ外側に転送するシステムです。

次は、IPv4 のみのネットワーク上のシステムについての単純なルーティングテーブルです。

```
Routing Table: IPv4
  Destination           Gateway           Flags  Ref   Use   Interface
-------------------- -------------------- ----- ----- ------ ---------
default              172.16.1.10          UG        1    532   net0
224.0.0.0            10.0.5.100           U         1      0   net1
10.0.0.0             10.0.5.100           U         1      0   net1
127.0.0.1            127.0.0.1            UH        1     57   lo0
```

Oracle Solaris システムでは、静的および動的という 2 種類のルーティングを構成できます。1 つのシステムに、これらのルーティング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。動的ルーティング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 などのルーティングプロトコルを利用して、ネットワークトラフィックをルーティングし、テーブル内のルーティング情報を更新します。静的ルーティングの場合、ルーティング情報は `route` コマンドを使用して手動で維持されます。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ルーティングを構成するときは、特定のルーターやホストでどの種類のルーティング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。
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 3–1 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。

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、『Oracle Solaris 11.1 ネットワークパフォーマンスの管理』の第 5 章「IPMP の概要」の説明に従って 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 ルーティング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   net0
224.0.0.0            198.168.5.10          U      1  0       net0
default              192.168.5.10          UG     1  91908
127.0.0.1            127.0.0.1             UH     1  811302   lo0
```

2 管理者になります。

詳細は、『Oracle Solaris 11.1 の管理: セキュリティーサービス』の「割り当てられている管理権限を使用する方法」を参照してください。

3 (オプション) ルーティング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。
`-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 3–2 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、図 3–1 のルーター 2 に静的ルートを追加する方法を示しています。静的ルートは AS のボーダールーター `10.0.5.150` で必要です。

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              172.16.1.10          UG        1    249 ce0
224.0.0.0            172.16.1.10          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 `172.16.1.10` 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              172.16.1.10          UG        1    249 ce0
224.0.0.0            172.16.1.10          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 の詳細は、『Oracle Solaris 11.1 ネットワークパフォーマンスの管理』の第 5 章「IPMP の概要」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。

- 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 – マルチホームホストのいずれかのインタフェースでファイアウォールの種類が異なるときは、ホストのパケットの予期しない混乱を回避するようにしてください。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、『Oracle Solaris 11.1 の管理: 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製のファイアウォールのドキュメントを参照してください。

▼ マルチホームホストの作成方法

1. 管理者になります。

   詳細は、『Oracle Solaris 11.1 の管理: セキュリティーサービス』の「割り当てられている管理権限を使用する方法」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ます。

   『Oracle Solaris 11.1 での固定ネットワーク構成を使用したシステムの接続』の「IP インタフェースを構成する方法」を参照してください。

3. パケット転送が有効な場合、このサービスを無効にします。

```
# ipadm show-prop -p forwarding ipv4
PROTO PROPERTY     PERM CURRENT      PERSISTENT   DEFAULT      POSSIBLE
ipv4  forwarding   rw   on           --           off          on,off

ipadm set-prop -p forwarding=off ipv4
```

4. (オプション) マルチホームホストの動的ルーティングをオンに設定します。

   次のいずれかのコマンド構文を使用します。

   - # **routeadm -e ipv4-routing -u**
   - # **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。

例 3–3 マルチホームホストの構成

次の例は、図 3–1 に示す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
LINK     CLASS     MTU     STATE   BRIDGE   OVER
net0     phys      1500    up      --       --
net1     phys      1500    up      --       --

# ipadm show-addr
ADDROBJ        TYPE     STATE        ADDR
lo0/v4         static   ok           127.0.0.1/8
net0/v4        static   ok           192.168.5.82/24
```

dladm show-link コマンドの報告は、hostc に 2 つのデータリンクがあることを示しています。ただし、net0 だけに IP アドレスが構成されています。hostc をマルチホームホストとして構成するには、net1 に、同じ 192.168.5.0 ネットワーク内の IP アドレスを構成します。net1 のベースとなる物理 NIC がネットワークに物理的に接続されていることを確認してください。

```
# ipadm create-ip net1
# ipadm create-addr static -a 192.168.5.85/24 net1
# ipadm show-addr
ADDROBJ        TYPE     STATE        ADDR
lo0/v4         static   ok           127.0.0.1/8
net0/v4        static   ok           192.168.5.82/24
net1/v4        static   ok           192.168.5.85/24
```

次に、net1 インタフェースを /etc/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        hostc    #primary network interface for host3
192.168.5.85        hostc-2  #second interface
```

次に、パケット転送が hostc 上で実行中の場合、このサービスをオフにします。

```
# ipadm show-prop -p forwarding ipv4
PROTO PROPERTY     PERM CURRENT   PERSISTENT   DEFAULT      POSSIBLE
ipv4  forwarding   rw   on        --           off          on,off

# ipadm set-prop -p forwarding=off ipv4

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

```
              IPv6 routing   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が現在有効になっていることを示しています。

単一インタフェースシステムのルーティングの構成

単一インタフェースシステムは、静的ルーティングまたは動的ルーティングのいずれかで構成できます。静的ルーティングでは、ホストはデフォルトルーターのサービスを利用してルーティング情報を取得する必要があります。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

次の手順を使用して、マルチホームホストで静的ルーティングを構成することもできます。

1. 管理者になります。

   詳細は、『Oracle Solaris 11.1 の管理: セキュリティーサービス』の「割り当てられている管理権限を使用する方法」を参照してください。

2. システムが属するネットワークの **IP** アドレスを使用して、システムの **IP** インタフェースを構成します。

   手順については、『Oracle Solaris 11.1 での固定ネットワーク構成を使用したシステムの接続』の「IP インタフェースを構成する方法」を参照してください。

3. テキストエディタを使用して、**/etc/defaultrouter** ファイルを作成または変更し、システムが使用するルーターの **IP** アドレスを追加します。

4. 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。

5. ルーティングがオフになっていることを確認します。

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

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

```
# svcadm disable route:default
```

6 パケット転送がオフになっていることを確認します。

```
# # ipadm show-prop -p forwarding ipv4
PROTO PROPERTY     PERM CURRENT   PERSISTENT   DEFAULT      POSSIBLE
ipv4  forwarding   rw   on        --           off          on,off

# ipadm set-prop -p forwarding=off ipv4
```

例 3–4 単一インタフェースシステムの静的ルーティングを構成する

次の例は、図 3–1 に示す 172.16.1.0 ネットワーク上にある単一インタフェースシステム hostb に静的ルーティングを構成する方法を示しています。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。この例は、システムの IP インタフェースがすでに構成されていることを想定しています。

まず、管理者権限で hostb にログインします。次に、システムに /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

# vi /etc/defaultrouter
172.16.1.10
```

IP アドレス 172.16.1.10 はルーター 2 に属しています。

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

# ipadm show-prop -p forwarding ipv4
PROTO PROPERTY     PERM CURRENT   PERSISTENT   DEFAULT      POSSIBLE
ipv4  forwarding   rw   on          --           off          on,off

# ipadm set-prop -p forwarding=off ipv4

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

          Routing services   "route:default ripng:default"

# svcadm disable route:default
```

▼ 単一インタフェースシステムで動的ルーティングを有効にする方法

ルーティングプロトコルを使用した動的ルーティングは、システム上でルーティングを管理するもっとも簡単な方法です。

1. 管理者になります。

   詳細は、『Oracle Solaris 11.1 の管理: セキュリティーサービス』の「割り当てられている管理権限を使用する方法」を参照してください。

2. システムが属するネットワークの **IP** アドレスを使用して、システムの **IP** インタフェースを構成します。

   手順については、『Oracle Solaris 11.1 での固定ネットワーク構成を使用したシステムの接続』の「IP インタフェースを構成する方法」を参照してください。

3. **/etc/defaultrouter** ファイル内にエントリがあれば削除します。

   /etc/defaultrouter ファイルが空の場合、システムは強制的に動的ルーティングを使用します。

4. パケット転送が無効になっていることを確認します。

   ```
   # ipadm set-prop -p forwarding=off ipv4
   ```

5. システムのルーティング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 -e ipv4-routing -u**
   - **# svcadm enable route:default**

例 3–5 単一インタフェースシステムで動的ルーティングを実行する

次の例は、図 3–1 に示すネットワーク 192.168.5.0 上にある単一インタフェースシステム hosta に動的ルーティングを構成する方法を示しています。システムはルーター 1 をデフォルトルーターとして使用します。この例は、システムの IP インタフェースがすでに構成されていることを想定しています。

まず、管理者権限で hosta にログインします。次に、/etc/defaultrouter ファイルがシステムに存在する場合はそれを削除します。

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

# rm defaultrouter
```

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

          Routing services   "route:default ripng:default"

# svcadm enable route:default

# ipadm show-prop -p forwarding ipv4
PROTO PROPERTY     PERM CURRENT   PERSISTENT   DEFAULT      POSSIBLE
ipv4  forwarding   rw   on        --           off          on,off

# ipadm set-prop -p forwarding=off ipv4
```

ネットワークへのサブネットの追加

サブネットを使用しないネットワークからサブネットを使用するネットワークに変更する場合、次の一覧に含まれるタスクを実行します。この一覧では、サブネットスキーマをすでに準備していることを想定しています。

- 新しいサブネット番号を持つ一連の IP アドレスを、サブネットに属するシステムに割り当てます。

  参照情報については、『Oracle Solaris 11.1 での固定ネットワーク構成を使用したシステムの接続』の「IP インタフェースを構成する方法」を参照してください。
- 正しい IP アドレスとネットマスクを各システムの `/etc/netmasks` ファイルに追加します。
- 各システムの `/etc/inet/hosts` ファイルを、ホスト名に対応する正しい IP アドレスで改訂します。
- サブネット内のすべてのシステムをリブートします。

次の手順はサブネットと密接に関係しています。当初はサブネットを用いずにネットワークを構成し、ずっとあとでサブネットを実装する場合、次の手順を実行して変更を実装します。

▼ 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、『Oracle Solaris 11.1 での固定ネットワーク構成を使用したシステムの接続』の「IP 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。

1 管理者になります。

詳細は、『Oracle Solaris 11.1 の管理: セキュリティーサービス』の「割り当てられている管理権限を使用する方法」を参照してください。

2 IP アドレス**ipadm** コマンドを使って変更します。

ipadm コマンドでは、IP アドレスを直接変更することはできません。最初に、修正対象の IP アドレスを表すアドレスオブジェクトを削除します。次に、同じアドレスのオブジェクト名を使って、新しいアドレスを割り当てます。

```
# ipadm delete-addr addrobj
# ipadm create-addr -a IP-address interface
```

3 該当する場合、**system/identity: node SMF** サービス内のホスト名エントリを変更します。

```
# hostname newhostname
```

このステップは永続的な変更をもたらします。

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

5 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。

6 システムをリブートします。

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

例 3–6 IP アドレスおよびホスト名を変更する

この例では、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。プライマリネットワークインタフェース `net0` の IP アドレスが `10.0.0.14` から `192.168.34.100` に変わります。

```
# ipadm show-addr
ADDROBJ      TYPE     STATE   ADDR
lo0/v4       static   ok      127.0.0.1/8
net0/v4      static   ok      10.0.0.14/24

# ipadm delete-addr net0/v4
# ipadm create-addr -a 192.168.34.100/24 net0
# hostname mynewhostname

# ipadm show-addr
ADDROBJ         TYPE     STATE   ADDR
lo0/v4          static   ok      127.0.0.1/8
net0/v4         static   ok      192.168.34.100/24

# hostname
mynewhostname
```

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 11.1 での固定ネットワーク構成を使用したシステムの接続』の「IP インタフェースを構成する方法」を参照してください。

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 11.1 でのサービスと障害の管理』の第 1 章「サービスの管理 (概要)」で説明しています。

`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、127 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 管理者になります。

詳細は、『Oracle Solaris 11.1 の管理: セキュリティーサービス』の「割り当てられている管理権限を使用する方法」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 11.1 でのサービスと障害の管理』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 3–7 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。『Oracle Solaris 11.1 での sendmail サービスの管理』の「sendmail の version 8.12 からの TCP ラッパーのサポート」で説明しているように、`sendmail` アプリケーションでも TCP ラッパーを使用できます。

1 管理者になります。

詳細は、『Oracle Solaris 11.1 の管理: セキュリティーサービス』の「割り当てられている管理権限を使用する方法」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 `/usr/sfw/man` ディレクトリから見つけることができます。

◆ ◆ ◆ 第 4 章

ネットワークでの IPv6 の有効化

この章では、IPv6 をネットワーク上で有効にするタスク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。

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 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 用にシステムを構成する方法

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。146 ページの「自動構成プロセス」で説明しているように、それらのインタフェースは最初、自動構成プロセスを介して IPv6 アドレスを取得します。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、66 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。

1 適切なコマンドを使用して **IP** インタフェースを構成します。

『Oracle Solaris 11.1 での固定ネットワーク構成を使用したシステムの接続』の「IP インタフェースを構成する方法」を参照してください。

注 – IP アドレスを割り当てるときには、必ず次のように正しいオプションを使用して IPv6 アドレスを割り当てます。

```
# ipadm create-addr -T addrconf interface
```

アドレスをさらに追加する場合は、次の構文を使用します。

```
# ipadm create-addr -a ipv6-address interface
```

---

2 **IPv6** デーモン **in.ndpd** を起動します。

```
# /usr/lib/inet/in.ndpd
```

3 **(**オプション**)** 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、69 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 133 ページの「`ndpd.conf` 構成ファイル」を参照してください。

5 (オプション) IP インタフェースのステータスをその IPv6 構成とともに表示するには、次のコマンドを入力します。

```
# ipadm show-addr
```

例 4–1 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、net0 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。

```
# ipadm show-addr
ADDROBJ   TYPE     STATE   ADDR
lo0/v4    static   ok      127.0.0.1/8
net0/v4   static   ok      172.16.27.74/24
```

このシステムに現在構成されているインタフェースは、net0 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。

```
# ipadm create-addr -T addrconf net0
# ipadm create-addr -a 2001:db8:3c4d:15:203/64 net0
# /usr/lib/inet/in.ndpd

# ipadm show-addr
ADDROBJ       TYPE       STATE   ADDR
lo0/v4        static     ok      127.0.0.1/8
net0/v4       static     ok      172.16.27.74/24
net0/v6       addrconf   ok      fe80::203:baff:fe13:14e1/10
lo0/v6        static     ok      ::1/128
net0/v6a      static     ok      2001:db8:3c4d:15:203/64

# route -p add -inet6 default fe80::203:baff:fe13:14e1
```

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、66 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、65 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、74 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、71 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 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。サーバー上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。

interface `StatelessAddrConf false`

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には、次のエントリを使用します。

```
ifdefault 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133 ページの「`ndpd.conf` 構成ファイル」を参照してください。

2 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

このセクションでは、IPv6 ルーターを構成するためのタスクについて説明します。サイトの要件によっては、一部のタスクのみの実行が必要な場合があります。

▼ IPv6 対応のルーターを構成する方法

次の手順では、システムがすでに IPv6 用に構成されているものとします。手順については、63 ページの「IPv6 インタフェースの構成」を参照してください。

1 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。

```
# ipadm set-prop -p forwarding=on ipv6
```

2 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。

- `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。

3 **/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、133 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。

4 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。

5 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 net0 および net1 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 net0 AdvSendAdvertisements 1
prefix 2001:0db8:3c4d:15::0/64 net0

if net1 AdvSendAdvertisements 1
prefix 2001:0db8:3c4d:16::0/64 net1
```

6 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 4–2 IPv6 インタフェースを表示する ipadm show-addr 出力

次の例に、66 ページの「IPv6 ルーターの構成」の手順を完了したあとに表示されるような ipadm show-addr コマンドの出力を示します。

```
ADDROBJ       TYPE       STATE   ADDR
lo0/v4        static     ok      127.0.0.1/8
net0/v4       static     ok      172.16.15.232/24
net1/v4       static     ok      172.16.16.220/24
net0/v6       addrconf   ok      fe80::203:baff:fe11:b115/10
lo0/v6        static     ok      ::1/128
net0/v6a      static     ok      2001:db8:3c4d:15:203:baff:fe11:b115/64
net1/v6       addrconf   ok      fe80::203:baff:fe11:b116/10
net1/v6a      static     ok      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。*interface*/v6 のようなアドレスオブジェクト名を含むエントリには、そのインタフェースのリンクローカルアドレスが表示されています。*interface*/v6add のようなアドレスオブジェクト名を含むエントリには、グローバル IPv6 アドレスが表示されています。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。v6add という指定はランダムに定義された文字列です。net0/mystring や net0/ipv6addr などのように、*interface* が IPv6 アドレスの作成先となるインタフェースを表しているかぎり、アドレスオブジェクト名の二番目の部分としてほかの文字列を定義できます。

参照
- IPv6 ネットワークトポロジで識別されたルーターからのトンネルを構成するには、112 ページの「dladm コマンドによるトンネルの構成と管理」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、68 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、74 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、133 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。IPv6 対応のインタフェースでは、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、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。

一般的な 3 つのタスクを次の順番で実行する必要があります。

1. IPv6 アドレスの自動構成を無効にします。65 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。
2. ホストの一時アドレスを作成します。69 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。
3. インタフェース ID の IPv6 トークンを構成します。72 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。

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. 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。
   64 ページの「IPv6 用にシステムを構成する方法」を参照してください。

2. **`/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
```

3 (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。

4 (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、100 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。

---

5 (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。

6 **in.ndpd** デーモンの構成を変更します。

```
# pkill -HUP in.ndpd
# /usr/lib/inet/in.ndpd
```

7 例 4–4 で示すように、**ipadm show-addr** コマンドを発行して一時アドレスが作成されたことを確認します。

コマンド出力では、一時アドレスの `CURRENT` フィールドに `t` フラグが表示されます。

例 4–3 /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 4–4 一時アドレスを有効にした状態での ipadm show-addr コマンドの出力

次に、一時アドレスを作成したあとの ipadm show-addr コマンドの出力の例を示します。このサンプル出力には IPv6 関連の情報のみが含まれています。

```
# ipadm show-addr -o all
ADDROBJ   TYPE     STATE CURRENT PERSISTENT ADDR
lo0/v6    static   ok    U----   ---        ::1/128
net0/v6   addrconf ok    U----   ---        fe80::a00:20ff:feb9:4c54/10
net0/v6a  static   ok    U----   ---        2001:db8:3c4d:15:a00:20ff:feb9:4c54/64
net0/?    addrconf ok    U--t-   ---        2001:db8:3c4d:15:7c37:e7d1:fc9c:d2cb/64
```

アドレスオブジェクト net0/? の CURRENT フィールドに t フラグが設定されています。このフラグは、対応するアドレスに一時インタフェース ID が含まれ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、75 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、72 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には、第 5 章「TCP/IP ネットワークの管理」を参照してください。

IPv6 トークンの構成

IPv6 アドレスの 64 ビットのインタフェース ID は、トークンとも呼ばれます。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 **トークンの構成対象となるインタフェースが存在しており、かつそのインタフェースで IPv6 アドレスが 1 つも構成されていないことを確認します。**

注 – 構成済みの IPv6 アドレスがインタフェースに一切含まれていないことを確認します。

```
# ipadm show-if
IFNAME   CLASS      STATE   ACTIVE   OVER
lo0      loopback   ok       yes     ---
net0     ip         ok       yes     ---

# ipadm show-addr
ADDROBJ      TYPE       STATE   ADDR
lo0/v4       static     ok      127.0.0.1/8
```

この出力は、ネットワークインタフェース `net0` が存在しており、IPv6 アドレスは一切構成されていないことを示しています。

2. 形式 *xxxx*:*xxxx*:*xxxx*:*xxxx* に従って、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。

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 ipadm 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が持つようにします。

   ```
   # ipadm create-addr -T addrconf -i interface-ID interface
   ```

   たとえば、インタフェース net0 をトークンで構成するには、次のコマンドを使用します。

   ```
   # ipadm create-addr -T addrconf -i ::1a:2b:3c:4d/64 net0
   ```

   注 – トークンを使用してアドレスオブジェクトが作成されると、そのトークンはもう変更できなくなります。

4. 変更に合わせて、IPv6 デーモンを更新します。

   ```
   # pkill -HUP in.ndpd
   ```

例 4–5 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

次の例では、IPv6 アドレスとトークンを使用して net0 を構成する方法を示します。

```
# ipadm show-if
IFNAME   CLASS      STATE    ACTIVE     OVER
lo0      loopback   ok       yes        ---
net0     ip         ok       yes        ---

# ipadm show-addr
ADDROBJ      TYPE       STATE   ADDR
lo0/v4       static     ok      127.0.0.1/8

# ipadm create-addr -T addrconf -i ::1a:2b:3c:4d/64 net0
# pkill -HUP in.ndpd
# ipadm show-addr
ADDROBJ      TYPE       STATE   ADDR
lo0/v6       static     ok      ::1/128
net0/v6      addrconf   ok      fe80::1a:2b:3c:4d/10
net0/v6a     addrconf   ok      2002:a08:39f0:1:1a:2b:3c:4d/64
```

トークンの構成が完了したあと、アドレスオブジェクト net0/v6 は、1a:2b:3c:4d がインタフェース ID として構成されたリンクローカルアドレスと別のアドレスの両方を持っています。net0/v6 が作成されたあとで、このトークンはこのインタフェースではもう変更できなくなります。

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、75 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 5 章「TCP/IP ネットワークの管理」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

この手順では、ネットワークのサーバーで IPv6 を有効にするための一般的な手順を提供します。IPv6 の実装方法によっては一部の手順が変わる可能性があります。

1. **サーバーの IP インタフェースで IPv6 を有効にします。**
   手順については、63 ページの「IPv6 インタフェースの構成」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、66 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 対応のインタフェースを置き換えないで使用する場合は、146 ページの「自動構成プロセス」で説明しているように、自動構成された IPv6 アドレスを使用します。
   - 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、69 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、72 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。

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 の管理の詳細は、『Oracle Solaris 11.1 でのネームサービスおよびディレクトリサービスの作業』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. 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*

2. **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 の管理の詳細は、『Oracle Solaris 11.1 でのネームサービスおよびディレクトリサービスの作業』を参照してください。

例 4–6 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 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 4–7 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 **4–8** 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 .byname
  ```

  指定した *hostname* についての情報が表示されます。

◆ ◆ ◆ 第 5 章

TCP/IP ネットワークの管理

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。次の項目について説明します。

注 – ネットワークインタフェースを監視する場合は、『Oracle Solaris 11.1 での固定ネットワーク構成を使用したシステムの接続』の「IP インタフェースとアドレスの監視」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 実装を計画するには、第 2 章「IPv6 アドレス使用時の考慮点」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には、第 4 章「ネットワークでの IPv6 の有効化」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節を示しています。

タスク	説明	参照先
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。	81 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ルーティング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。	84 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。転送の問題をトラブルシュートするときに役立ちます。	84 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。	86 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、`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。このファイル内の `DEFAULT_IP` 変数を設定します。	89 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。	96 ページの「IPv6 ネットワークトラフィックを監視する方法」
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。	93 ページの「すべてのルートをトレースする方法」

注 – ネットワークインタフェースを監視する場合は、『Oracle Solaris 11.1 での固定ネットワーク構成を使用したシステムの接続』の「IP インタフェースとアドレスの監視」を参照してください

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。ルーティング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 5–1 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 5–2 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 5–3 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 5–4 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 または ethers データベース内の誤ったアドレスが原因である可能性があります。

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
net0  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
net0  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ルーティング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 5–5 すべてのソケットとルーティング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 5–6 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 5–7 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、ローカルホストのルーティング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 ルーティングテーブルを表示します。**

```
$ netstat -r
```

例 5–8 netstat コマンドによるルーティングテーブルの出力

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  net0
10.0.0.14            myhost               U         1      0  net0
default              distantrouter        UG        1      2  net0
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      net0:1
fe80::/10             fe80::1a2b:3c4d:5e6f:12a2           U     1   23     net0
ff00::/8              fe80::1a2b:3c4d:5e6f:12a2           U     1   0      net0
default               fe80::1a2b:3c4d:5e6f:12a2           UG    1   0      net0
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 ルーティングテーブルには、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。

パラメータ	説明
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 5–9 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
```

パケットロスという統計は、ホストがパケットを失っているかどうかを示します。ping が失敗した場合は、ipadm コマンドと netstat コマンドによって報告されたネットワークのステータスをチェックします。『Oracle Solaris 11.1 での固定ネットワーク構成を使用したシステムの接続』の「IP インタフェースとアドレスの監視」および81 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

次のタスクに、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 – netstat コマンドの -f フラグは、inet_type ファイルに設定された値をオーバーライドします。

例 5–10 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。

  ```
  % ipadm show-addr
  ADDROBJ       TYPE       STATE   ADDR
  lo0/v4        static     ok      127.0.0.1/8
  net0/v4       static     ok      10.46.86.54/24
  lo0/v6        static     ok      ::1/128
  net0/v6       addrconf   ok      fe80::a00:fe73:56a8/10
  net0/v6add    static     ok      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。

  ```
  % ipadm show-addr
  ADDROBJ       TYPE       STATE   ADDR
  lo0/v4        static     ok      127.0.0.1/8
  net0/v4       static     ok      10.46.86.54/24
  ```

▼ IPv4 ルーティング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。

- 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 5–11 in.routed デーモンのネットワークログ

次の例に、90 ページの「IPv4 ルーティング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 net0 #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  net0  <NET_SYN>
Add    10.10.48.85/25  -->10.10.48.112      metric=0  net0  <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 **in.ndpd** デーモンのトレースを起動します。

```
# /usr/lib/inet/in.ndpd -t
```

2 トレースを終了するには、**Ctrl-C** を押します。

例 5–12 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 net0
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 net0
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 コマンドによるルーティング情報の表示

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、ルーティングの誤構成やルーティング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 5–13 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 5–14 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. 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。

   ```
   # ipadm show-if
   ```

   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。

2. **Example 5–15** に示すように、例 **5–15** 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。

3. **Ctrl-C** キーを押してプロセスを停止します。

例 5–15 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/net (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)
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 **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。

2 **snoop** 出力取り込みファイルを検査します。

```
# snoop -i filename
```

例 **5–16** 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** 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。

3. 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。

- **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 5–17 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
```

IP 層デバイスを使用したパケット監視

IP の監視機能を強化するため、IP 層デバイスが Oracle Solaris に導入されています。これらのデバイスは、システムのネットワークインタフェースに関連付けられたアドレスを含むすべてのパケットにアクセスできます。アドレスには、ローカルアドレスのほか、非ループバックインタフェースまたは論理インタフェースにホストされたアドレスも含まれます。監視可能なトラフィックには、IPv4、IPv6 のどちらのアドレスが含まれていてもかまいません。したがって、システムに向かうすべてのトラフィックを監視できます。トラフィックには、ループバック IP トラフィック、リモートマシンからのパケット、システムから送信されるパケット、またはすべての転送トラフィックが含まれる場合があります。

大域ゾーンの管理者は IP 層デバイスを使用することで、ゾーン間のトラフィックやゾーン内のトラフィックを監視できます。非大域ゾーンの管理者も、そのゾーンによって送受信されるトラフィックを監視できます。

IP 層でトラフィックを監視するために、snoop コマンドに新しいオプション -I が追加されています。このオプションは、コマンドが、ベースとなるリンク層デバイスではなく新しい IP 層デバイスを使用してトラフィックデータを表示することを指定します。

▼ IP 層でパケットをチェックする方法

1 必要であれば、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。

```
# ipadm show-if
```

2 特定のインタフェースの IP トラフィックを取得します。

```
# snoop -I interface [-V | -v]
```

パケットのチェック例

すべての例は次のシステム構成に基づいています。

```
# ipadm show-addr
ADDROBJ      TYPE       STATE   ADDR
lo0/v4       static     ok      127.0.0.1/8
net0/v4      static     ok      192.68.25.5/24
lo0/?        static     ok      127.0.0.1/8
net0/?       static     ok      172.0.0.3/24
net0/?       static     ok      172.0.0.1/24
lo0/?        static     ok      127.0.0.1/8
```

2 つのゾーン sandbox と toybox が次の IP アドレスを使用しているとします。

- sandbox – 172.0.0.3
- toybox – 172.0.0.1

snoop -I コマンドは、システム上のさまざまなインタフェースに対して発行できます。表示されるパケット情報は、ユーザーが大域ゾーン、非大域ゾーンのいずれの管理者であるかに依存します。

例 5–18 ループバックインタフェース上のトラフィック

```
# snoop -I lo0
Using device ipnet/lo0 (promiscuous mode)
   localhost -> localhost    ICMP Echo request (ID: 5550 Sequence number: 0)
   localhost -> localhost    ICMP Echo reply (ID: 5550 Sequence number: 0)
```

冗長出力を生成するには、-v オプションを使用します。

```
# snoop -v -I lo0
Using device ipnet/lo0 (promiscuous mode)
IPNET:  ----- IPNET Header -----
```

```
IPNET:
IPNET:  Packet 1 arrived at 10:40:33.68506
IPNET:  Packet size = 108 bytes
IPNET:  dli_version = 1
IPNET:  dli_type = 4
IPNET:  dli_srczone = 0
IPNET:  dli_dstzone = 0
IPNET:
IP:   ----- IP Header -----
IP:
IP:   Version = 4
IP:   Header length = 20 bytes
...
```

IP 層でのパケット監視のサポートのために、監視対象となるパケットの前に新しい ipnet ヘッダーが導入されています。発信元 ID と着信先 ID の両方が示されます。「0」の ID は、トラフィックが大域ゾーンから生成されていることを示します。

例 5–19　ローカルゾーンの net0 デバイスでのパケットフロー

```
# snoop -I net0
Using device ipnet/net0 (promiscuous mode)
toybox -> sandbox TCP D=22 S=62117 Syn Seq=195630514 Len=0 Win=49152 Options=<mss
sandbox -> toybox TCP D=62117 S=22 Syn Ack=195630515 Seq=195794440 Len=0 Win=49152
toybox -> sandbox TCP D=22 S=62117 Ack=195794441 Seq=195630515 Len=0 Win=49152
sandbox -> toybox TCP D=62117 S=22 Push Ack=195630515 Seq=195794441 Len=20 Win=491
```

この出力には、システム内のさまざまなゾーンで発生したトラフィックが表示されています。ローカルでほかのゾーンに配信されるパケットも含め、`net0` の IP アドレスに関連するすべてのパケットを表示できます。冗長出力を生成すれば、パケットのフローに関連するゾーンを確認できます。

```
# snoop -I net0 -v port 22
IPNET:  ----- IPNET Header -----
IPNET:
IPNET:  Packet 5 arrived at 15:16:50.85262
IPNET:  Packet size = 64 bytes
IPNET:  dli_version = 1
IPNET:  dli_type = 0
IPNET:  dli_srczone = 0
IPNET:  dli_dstzone = 1
IPNET:
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:         .... ..0. = not ECN capable transport
IP:         .... ...0 = no ECN congestion experienced
IP:   Total length = 40 bytes
IP:   Identification = 22629
```

例 5–19 ローカルゾーンの net0 デバイスでのパケットフロー (続き)

```
IP:   Flags = 0x4
IP:         .1.. .... = do not fragment
IP:         ..0. .... = last fragment
IP:   Fragment offset = 0 bytes
IP:   Time to live = 64 seconds/hops
IP:   Protocol = 6 (TCP)
IP:   Header checksum = 0000
IP:   Source address = 172.0.0.1, 172.0.0.1
IP:   Destination address = 172.0.0.3, 172.0.0.3
IP:   No options
IP:
TCP:  ----- TCP Header -----
TCP:
TCP:  Source port = 46919
TCP:  Destination port = 22
TCP:  Sequence number = 3295338550
TCP:  Acknowledgement number = 3295417957
TCP:  Data offset = 20 bytes
TCP:  Flags = 0x10
TCP:        0... .... = No ECN congestion window reduced
TCP:        .0.. .... = No ECN echo
TCP:        ..0. .... = No urgent pointer
TCP:        ...1 .... = Acknowledgement
TCP         .... 0... = No push
TCP         .... .0.. = No reset
TCP:        .... ..0. = No Syn
TCP:        .... ...0 = No Fin
TCP:  Window = 49152
TCP:  Checksum = 0x0014
TCP:  Urgent pointer = 0
TCP:  No options
TCP:
```

この ipnet ヘッダーは、パケットが大域ゾーン (ID 0) から Sandbox (ID 1) に向かっていることを示しています。

例 5–20 ゾーンを特定してトラフィックを監視する

```
# snoop -I hme0 sandboxsnoop -I net0 sandbox
Using device ipnet/hme0 (promiscuous mode)
toybox -> sandbox TCP D=22 S=61658 Syn Seq=374055417 Len=0 Win=49152 Options=<mss
sandbox -> toybox TCP D=61658 S=22 Syn Ack=374055418 Seq=374124525 Len=0 Win=49152
toybox -> sandbox TCP D=22 S=61658 Ack=374124526 Seq=374055418 Len=0 Win=49152
#
```

ゾーンを特定してパケットを監視する機能は、複数のゾーンを含むシステムで役立ちます。現在のところ、ゾーンを特定するときに使用できるのは、ゾーン ID だけです。ゾーン名による snoop の使用はサポートされていません。

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 ルーティングアドレス、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、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 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
```

2 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
```

3 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。

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 /etc/inet/ipaddrsel.conf
```

5 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. **`/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
   ```

2. 必要に応じて、*filename* 内のポリシーテーブルを編集します。

3. 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 6 章

IP トンネルの構成

この章では、IP トンネルについて説明するほか、Oracle Solaris でトンネルを構成および維持するための手順を示します。

IP トンネルの概要

IP トンネルは、ドメイン内のプロトコルが中間のネットワークでサポートされないときにドメイン間でデータパケットを転送するための手段を提供します。たとえば、大部分のネットワークで IPv4 プロトコルが使用されている環境では、IPv6 プロトコルの導入時に、IPv6 ネットワークは境界の外側での通信手段を必要とします。トンネルを使用すれば通信が可能となります。IP トンネルは、IP を使用して到達可能な 2 つのノード間で仮想リンクを提供します。したがって、このリンクを使用すれば IPv4 ネットワーク経由で IPv6 パケットを転送でき、2 つの IPv6 サイト間での IPv6 通信を実現できます。

Oracle Solaris 11 での IP トンネル管理

この Oracle Solaris リリースではトンネル管理が改訂され、新しいネットワークデータリンク管理モデルと一貫性を持つようになりました。トンネルは、`dladm` の新しいサブコマンドを使用して作成および構成されるようになりました。さらにトンネルでは、新しい管理モデルのその他のデータリンク機能も使用できるようになりました。たとえば、管理者によって選択された名前のサポートにより、トンネルに意味のある名前を割り当てることができます。`dladm` サブコマンドの詳細は、`dladm(1M)` のマニュアルページを参照してください。

トンネルのタイプ

トンネリングでは、IP パケットが別のパケット内にカプセル化されます。このカプセル化によって、パケットは、パケットのプロトコルをサポートしない中間のネットワークを介して宛先に到達できます。

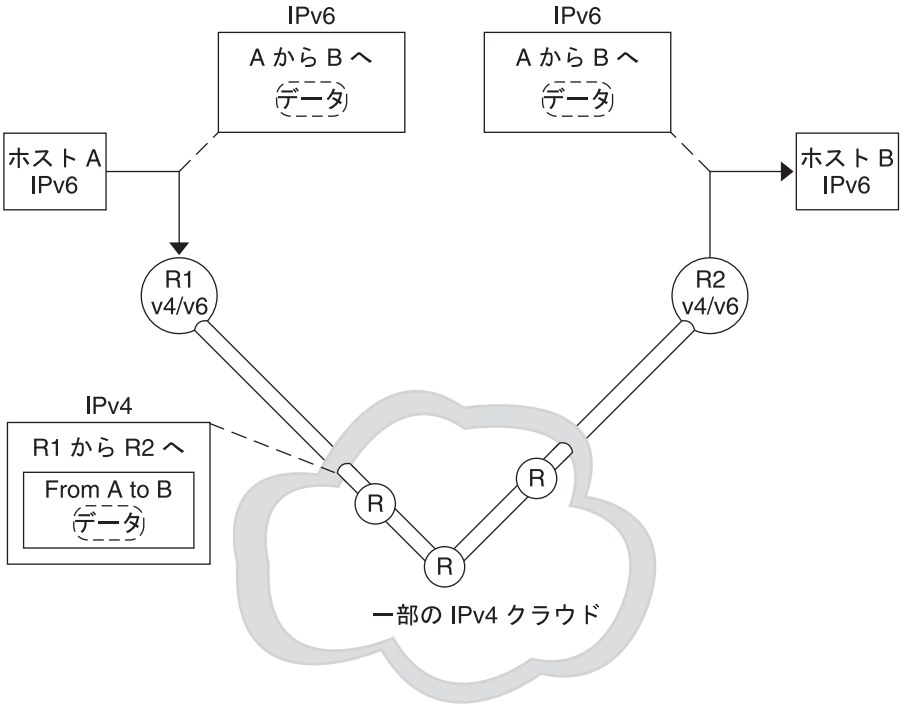
トンネルは、パケットカプセル化のタイプごとに異なります。Oracle Solaris でサポートされるトンネルのタイプは、次のとおりです。

- *IPv4* トンネル – IPv4 または IPv6 パケットが IPv4 ヘッダー内にカプセル化され、事前に構成されたユニキャスト IPv4 宛先に送信されます。トンネルを通過するパケットをより具体的に示すため、IPv4 トンネルは *IPv4 over IPv4* トンネルまたは *IPv6 over IPv4* トンネルとも呼ばれます。
- *IPv6* トンネル – IPv4 または IPv6 パケットが IPv6 ヘッダー内にカプセル化され、事前に構成されたユニキャスト IPv6 宛先に送信されます。トンネルを通過するパケットをより具体的に示すため、IPv6 トンネルは *IPv4 over IPv6* トンネルまたは *IPv6 over IPv6* トンネル とも呼ばれます。
- *6to4* トンネル – IPv6 パケットが IPv4 ヘッダー内にカプセル化され、パケット単位で自動的に決定される IPv4 宛先に送信されます。この決定は、6to4 プロトコル内に定義されたアルゴリズムに基づきます。

IPv6 と IPv4 を組み合わせたネットワーク環境でのトンネル

IPv6 ドメインを持つほとんどのサイトは、ほかの IPv6 ドメインと通信する際に、IPv6 のみのネットワークよりも数多く存在している IPv4 ネットワークをたどります。次の図に、IPv4 ルーター (図中の “R”) を通る 2 つの IPv6 ホスト間でのトンネルメカニズムを示します。

図 6–1 IPv6 トンネルメカニズム



この図のトンネルは、2 つのルーター間で IPv4 ネットワーク経由で仮想的なポイントツーポイントリンクを持つように構成された 2 つのルーターから構成されています。

IPv6 パケットが 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経由で宛先の境界ルーターに転送され、そこでそのカプセル化が解除されます。次に、そのルーターは個々の IPv6 パケットを宛先のノードに転送します。

6to4 トンネル

Oracle Solaris には、IPv4 アドレス指定から IPv6 アドレス指定に移行するための推奨の暫定的な手段として、6to4 トンネルが含まれています。6to4 トンネルを使用すると、孤立した IPv6 サイトが、IPv6 をサポートしない IPv4 ネットワーク上の自動トンネルを介して通信できます。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。

このセクションでは、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 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

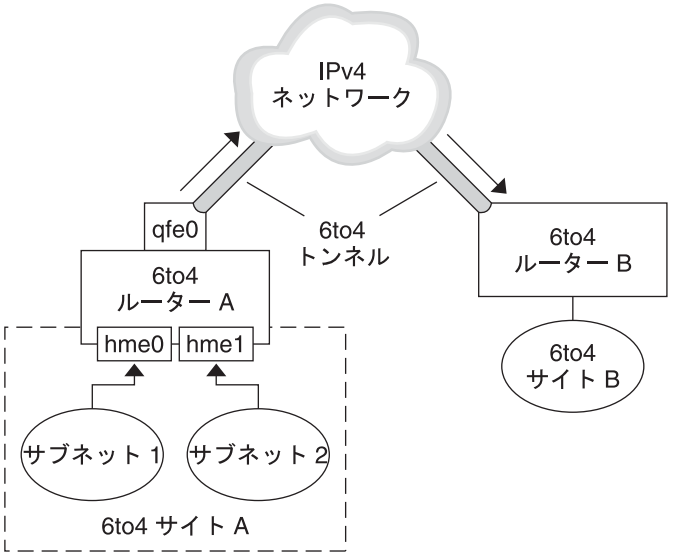
次の表では、6to4 トンネルを構成するための追加タスクについて説明し、有用な追加情報の入手先を示しています。

タスクまたは詳細	参照先
6to4 トンネルの構成タスク	117 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 のセキュリティー問題	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 6–2　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 6–2 で使用した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 ルーティング手続きを使用し 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 6–3　6to4 サイトと 6to4 リレールーター間のトンネル

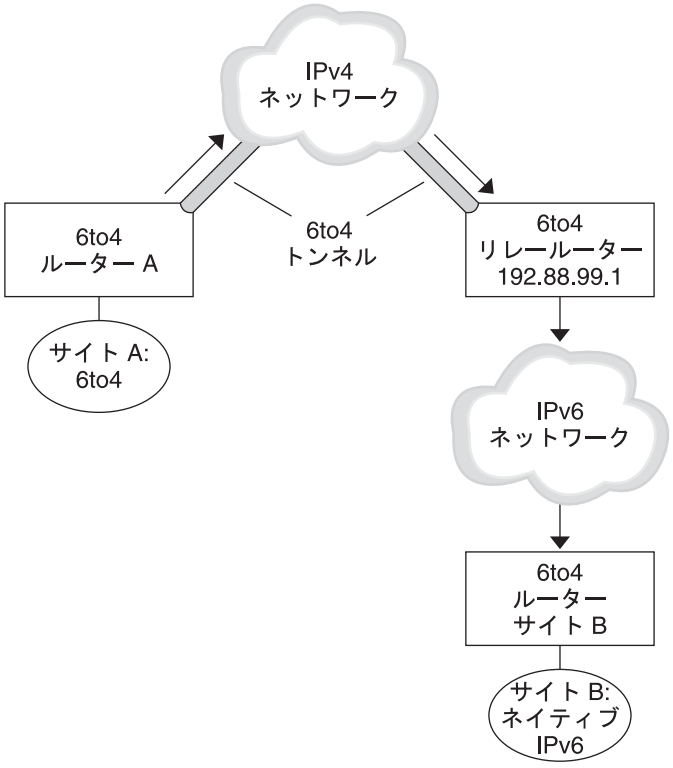


図 6–3 の 6to4 サイト A は、ネイティブ IPv6 サイト B のノードと通信する必要があります。図には、IPv4 ネットワーク経由でサイト A から 6to4 トンネルに向かうトラフィックのパスが示されています。このトンネルは、6to4 ルーター A と 6to4 リレールーターをエンドポイントとして使用しています。6to4 リレールーターより先は IPv6 ネットワークであり、IPv6 サイト B はこのネットワークに接続されています。

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 6–3 で使用した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 ルーティング手続きを使用し 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。

トンネルの配備

IP トンネルを正しく配備するには、2 つの主要タスクを実行する必要があります。まず、トンネルリンクを作成します。次に、そのトンネル上で IP インタフェースを構成します。このセクションでは、トンネルとそれらに対応する IP インタフェースを作成するための要件について簡単に説明します。

トンネルを作成するための要件

トンネルを正常に作成するには、次の要件を監視する必要があります。

- リテラル IP アドレスの代わりにホスト名を使用する場合、それらの名前はトンネルのタイプと互換性のある有効な IP アドレスに解決される必要があります。
- 作成する IPv4 または IPv6 トンネルは、構成済みの別のトンネルと同じトンネル発信元アドレスとトンネル着信先アドレスを共有することはできません。
- 作成する IPv4 または IPv6 トンネルは、既存の 6to4 トンネルと同じトンネル発信元アドレスを共有することはできません。
- 6to4 トンネルを作成する場合、そのトンネルは、構成済みの別のトンネルと同じトンネル発信元アドレスを共有することはできません。

ネットワークでのトンネルの設定については、31 ページの「ネットワークでのトンネル使用の計画」を参照してください。

トンネルと IP インタフェースの要件

各トンネルタイプは、そのトンネル上で構成される IP インタフェースに対して特定の IP アドレス要件を持ちます。要件は次の表に要約されています。

表 6–1　トンネルと IP インタフェースの要件

トンネルタイプ	トンネル上で許可される IP インタフェース	IP インタフェースの要件
IPv4 トンネル	IPv4 インタフェース	ローカルとリモートのアドレスは手動で指定されます。
	IPv6 インタフェース	`ipadm create-addr -T addrconf` コマンドの発行時に、ローカルとリモートのリンクローカルアドレスが自動的に設定されます。詳細は、`ipadm(1M)` のマニュアルページを参照してください。
IPv6 トンネル	IPv4 インタフェース	ローカルとリモートのアドレスは手動で指定されます。
	IPv6 インタフェース	`ipadm create-addr -T addrconf` コマンドの発行時に、ローカルとリモートのリンクローカルアドレスが自動的に設定されます。詳細は、`ipadm(1M)` のマニュアルページを参照してください。
6to4 トンネル	IPv6 インタフェースのみ	`ipadm create-ip` コマンドの発行時に、デフォルトの IPv6 アドレスが自動的に選択されます。詳細は、`ipadm(1M)` のマニュアルページを参照してください。

6to4 トンネルのデフォルトの IPv6 インタフェースアドレスは、`ipadm` コマンドで別の IPv6 アドレスを指定することによってオーバーライドできます。

同様に、IPv4 または IPv6 トンネル上の IPv6 インタフェースに対して自動的に設定されたリンクローカルアドレスをオーバーライドするために、そのトンネルのホストファイル内で別の発信元アドレスと着信先アドレスを指定できます。

dladm コマンドによるトンネルの構成と管理

このセクションでは、dladm コマンドを使用してトンネルを構成する手順について説明します。

dladm サブコマンド

この Oracle Solaris リリースから、トンネルの管理が IP インタフェースの構成から分離されました。IP トンネルのデータリンクの側面は、dladm コマンドで管理されるようになりました。さらに、IP トンネルインタフェースを含む IP インタフェースの構成は、ipadm コマンドで実行されます。

dladm の次のサブコマンドを使用して、IP トンネルを構成します。

- create-iptun
- modify-iptun
- show-iptun
- delete-iptun
- set-linkprop

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。

注 – IP トンネルの管理は、IPsec の構成と密接に関連しています。たとえば、IPsec 仮想プライベートネットワーク (VPN) は、IP トンネリングの主な用途の 1 つです。Oracle Solaris でのセキュリティーの詳細は、『Oracle Solaris 11.1 でのネットワークのセキュリティー保護』の第 6 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 を構成する場合は、『Oracle Solaris 11.1 でのネットワークのセキュリティー保護』の第 7 章「IPsec の構成 (タスク)」を参照してください。

トンネルの構成 (タスクマップ)

タスク	説明	手順
IP トンネルを作成します。	ネットワーク経由の通信に使用されるトンネルを構成します。	113 ページの「IP トンネルを作成および構成する方法」
トンネルの構成を変更します。	トンネルの発信元アドレスや着信先アドレスなど、トンネルの元のパラメータを変更します。	121 ページの「IP トンネルの構成を変更する方法」
トンネルの構成を表示します。	特定のトンネル、またはシステムのすべての IP トンネルの構成情報を表示します。	122 ページの「IP トンネルの構成を表示する方法」

タスク	説明	手順
トンネルを削除します。	トンネルの構成を削除します。	123 ページの「IP トンネルを削除する方法」

▼ IP トンネルを作成および構成する方法

1 トンネルを作成します。

```
# dladm create-iptun [-t] -T type -a [local|remote]=addr,... tunnel-link
```

このコマンドで使用可能なオプションまたは引数は、次のとおりです。

`-t`
: 一時的なトンネルを作成します。このコマンドはデフォルトでは永続的なトンネルを作成します。

注 – トンネル上で永続的 IP インタフェースを構成するには、永続的なトンネルを作成し、`-t` オプションは使用しないようにする必要があります。

`-T` *type*
: 作成するトンネルのタイプを指定します。この引数は、どのトンネルタイプを作成する場合も必要です。

`-a [local|remote]=`*address,...*
: ローカルアドレスとリモートトンネルアドレスに対応するリテラル IP アドレスまたはホスト名を指定します。これらのアドレスは有効であり、かつシステム内ですでに作成されている必要があります。トンネルのタイプに応じて、アドレスを 1 つだけ指定するか、ローカルアドレスとリモートアドレスの両方を指定します。ローカルアドレスとリモートアドレスの両方を指定する場合は、それらのアドレスをコンマで区切ります。

- `IPv4` トンネルが機能するためには、ローカルとリモートの IPv4 アドレスが必要です。
- `IPv6` トンネルが機能するためには、ローカルとリモートの IPv6 アドレスが必要です。
- `6to4` トンネルが機能するためには、ローカル IPv4 アドレスが必要です。

注 – 永続的な IP トンネルデータリンクの構成でホスト名をアドレスとして使用した場合、それらのホスト名が構成ストレージに保存されます。次回以降のシステムブート時に、トンネル作成時に使用された IP アドレスとは異なる IP アドレスに名前が解決された場合、トンネルは新しい構成を取得します。

tunnel-link

IP トンネルリンクを指定します。ネットワークリンク管理での意味のある名前のサポートにより、トンネル名が作成対象トンネルのタイプに制限されなくなりました。代わりに、管理者によって選択された名前をトンネルに割り当てることができます。トンネル名は、*mytunnel0* のように、文字列と物理接続点 (PPA) 番号から構成されます。意味のある名前の割り当てを制御する規則については、『Oracle Solaris 11 ネットワーキングの紹介』の「有効なリンク名のための規則」を参照してください。

トンネルリンクを指定しなかった場合には、次の命名規則に従って名前が自動的に提供されます。

- IPv4 トンネルの場合: `ip.tun#`
- IPv6 トンネルの場合: `ip6.tun#`
- 6to4 トンネルの場合: `ip.6to4tun#`

は、作成するトンネルタイプで使用可能な PPA 番号のうち、もっとも小さい番号になります。

2 (オプション) ホップ制限またはカプセル化制限の値を設定します。

```
# dladm set-linkprop -p [hoplimit=value] [encaplimit=value] tunnel-link
```

hoplimit　IPv6 上でのトンネリング用のトンネルインタフェースのホップ制限を指定します。*hoplimit* は、IPv4 上でのトンネリングの IPv4 生存時間 (TTL) フィールドに相当します。

encaplimit　1 つのパケットで許可される入れ子のトンネリングのレベル数を指定します。このオプションは IPv6 トンネルにのみ適用されます。

1 つのパケットで許可される入れ子のトンネリングのレベル数を指定します。このオプションは IPv6 トンネルにのみ適用されます。

注 – hoplimit と encaplimit に設定する値は、許容範囲内にとどまっている必要があります。hoplimit と encaplimit はトンネルリンクのプロパティーです。したがって、これらのプロパティーは、ほかのリンクプロパティーと同じ dladm サブコマンドを使って管理します。サブコマンドは、dladm set-linkprop、dladm reset-linkprop、および dladm show-linkprop です。dladm コマンドで使用されるリンク管理用のさまざまなサブコマンドについては、dladm(1M) のマニュアルページを参照してください。

3 トンネル上で IP インタフェースを作成します。

```
# ipadm create-ip tunnel-interface
```

ここで、*tunnel-interface* ではトンネルリンクと同じ名前を使用します。

4 ローカルおよびリモートの IP アドレスをトンネルインタフェースに割り当てます。

```
# ipadm create-addr [-t] -a local=address,remote=address  interface
```

-t	トンネル上の永続的な IP 構成ではなく一時的な IP 構成を示します。このオプションを使用しない場合、IP インタフェースの構成は永続的な構成になります。
-a local=*address* ,remote=*address*	トンネルインタフェースの IP アドレスを指定します。local と remote で表される発信元と着信先の両方の IP アドレスが必要です。ローカルとリモートのアドレスは、IPv4 または IPv6 のいずれかのアドレスを使用できます。
interface	トンネルインタフェースを指定します。

ipadm コマンドや、トンネルインタフェースなどの IP インタフェースを構成するための各種オプションの詳細は、ipadm(1M) のマニュアルページおよび『Oracle Solaris 11.1 での固定ネットワーク構成を使用したシステムの接続』を参照してください。

5 /etc/hosts ファイルにトンネル構成情報を追加します。

6 (オプション) トンネルの IP インタフェース構成のステータスを確認します。

```
# ipadm show-addr interface
```

例 6–1 IPv4 トンネル上での IPv6 インタフェースの作成

この例では、永続的な IPv6 over IPv4 トンネルを作成する方法を示します。

```
# dladm create-iptun -T ipv4 -a local=63.1.2.3,remote=192.4.5.6 private0
# dladm set-linkprop -p hoplimit=200 private0
```

```
# ipadm create-ip private0
# ipadm create-addr -T addrconf private0
# ipadm show-addr private/
ADDROBJ       TYPE     STATE   ADDR
private0/v6   static   ok      fe80::a08:392e/10 --> fe80::8191:9a56
```

代替アドレスを追加するには、同じ構文を使用します。たとえば、次のようにしてグローバルアドレスを追加できます。

```
# ipadm create-addr -a local=2001:db8:4728::1, \
remote=2001:db8:4728::2 private0
# ipadm show-addr private0/
ADDROBJ          TYPE       STATE   ADDR
private0/v6      addrconf   ok      fe80::a08:392e/10 --> fe80::8191:9a56
private0/v6a     static     ok      2001:db8:4728::1 --> 2001:db8:4728::2
```

IPv6 アドレスの接頭辞 2001:db8 は、ドキュメントの例で特別に使用される特殊な IPv6 接頭辞です。

例 6–2 IPv4 トンネル上での IPv4 インタフェースの作成

この例では、永続的な IPv4 over IPv4 トンネルを作成する方法を示します。

```
# dladm create-iptun -T ipv4 -a local=63.1.2.3,remote=192.4.5.6 vpn0
# ipadm create-ip vpn0
# ipadm create-addr -a local=10.0.0.1,remote=10.0.0.2 vpn0
# ipadm show-addr
ADDROBJ   TYPE     STATE   ADDR
lo0/v4    static   ok      127.0.0.1
vpn0/v4   static   ok      10.0.0.1-->10.0.0.2
```

さらに、このトンネル上を通過するパケットに対してセキュリティー保護された接続を提供するために、IPsec ポリシーを構成することもできます。IPsec の構成については、『Oracle Solaris 11.1 でのネットワークのセキュリティー保護』の第 7 章「IPsec の構成 (タスク)」を参照してください。

例 6–3 IPv6 トンネル上での IPv6 インタフェースの作成

この例では、永続的な IPv6 over IPv6 トンネルを作成する方法を示します。

```
# dladm create-iptun -T ipv6 -a local=2001:db8:feed::1234,remote=2001:db8:beef::4321 \
tun0
# ipadm create-ip tun0
# ipadm create-addr -T addrconf tun0
# ipadm show-addr
ADDROBJ   TYPE       STATE   ADDR
lo0/v6    static     ok      ::1/128
tun0/v6   addrconf   ok      2001:db8:feed::1234 --> 2001:db8:beef::4321
```

グローバルアドレスや代替のローカルおよびリモートアドレスなどのアドレスを追加するには、次のように ipadm コマンドを使用します。

```
# ipadm create-addr \
-a local=2001:db8::4728:56bc,remote=2001:db8::1428:57ab tun0
# ipadm show-addr tun0
ADDROBJ    TYPE      STATE ADDR
tun0/v6    addrconf  ok    2001:db8:feed::1234 --> 2001:db8:beef::4321
tun0/v6a   static    ok    2001:db8::4728:56bc --> 2001:db8::1428:57ab
```

▼ 6to4 トンネルを構成する方法

6to4 トンネルでは、6to4 ルーターは、ネットワークの 6to4 サイト内のノードに対して IPv6 ルーターとして機能する必要があります。このため、6to4 ルーターを構成するときには、そのルーターを、その物理インタフェース上で IPv6 ルーターとしても構成する必要があります。IPv6 ルーティングの詳細は、152 ページの「IPv6 のルーティング」を参照してください。

1 **6to4 トンネルを作成します。**

 # dladm create-iptun -T 6to4 -a local=*address tunnel-link*

このコマンドで使用可能なオプションまたは引数は、次のとおりです。

-a local=*address* トンネルのローカルアドレスを指定します。これが有効なアドレスであるためには、それがすでにシステム内に存在している必要があります。

tunnel-link IP トンネルリンクを指定します。ネットワークリンク管理での意味のある名前のサポートにより、トンネル名が作成対象トンネルのタイプに制限されなくなりました。代わりに、管理者が選択した名前をトンネルに割り当てることができます。トンネル名は、*mytunnel0* のように、文字列と PPA 番号から構成されます。意味のある名前の割り当てを制御する規則については、『Oracle Solaris 11 ネットワーキングの紹介』の「有効なリンク名のための規則」を参照してください。

2 **トンネルの IP インタフェースを作成します。**

 # ipadm create-ip *tunnel-interface*

ここで、*tunnel-interface* ではトンネルリンクと同じ名前を使用します。

3 **(オプション) トンネルで使用するための代替 IPv6 アドレスを追加します。**

4 **6to4 ルーティングを通知するために /etc/inet/ndpd.conf ファイルを編集して次の 2 行を追加します。**

```
if subnet-interface AdvSendAdvertisements 1
IPv6-address subnet-interface
```

1 行目は、広告を受け取るサブネットを指定します。*subnet-interface* は、サブネットが接続されているリンクを表します。2 行目の IPv6 アドレスの接頭辞は、6to4 トンネルの IPv6 アドレスで使用される 6to4 接頭辞 `2000` になっている必要があります。

`ndpd.conf` ファイルの詳細は、`ndpd.conf(4)` のマニュアルページを参照してください。

5 **IPv6 転送を有効にします。**

```
# ipadm set-prop -p forwarding=on ipv6
```

6 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。

7 **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、75 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 6–4 6to4 トンネルの作成

この例ではサブネットのインタフェースが `bge0` になっていますが、このインタフェースは、該当の手順で `/etc/inet/ndpd.conf` 内で参照されます。

この例では 6to4 トンネルを作成する方法を示します。6to4 トンネル上で構成できるのは IPv6 インタフェースだけです。

```
# dladm create-iptun -T 6to4 -a local=192.168.35.10 tun0
# ipadm create-ip tun0
# ipadm show-addr
ADDROBJ       TYPE     STATE   ADDR
lo0/v4        static   ok      127.0.0.1/8
net0/v4       static   ok      192.168.35.10/24
lo0/v6        static   ok      ::1/128
tun0/_a       static   ok      2002:c0a8:57bc::1/64

# ipadm create-addr -a 2002:c0a8:230a::2/16 tun0
# ipadm create-addr -a 2002:c0a8:230a::3/16 tun0
# ipadm show-addr tun0
ADDROBJ       TYPE     STATE   ADDR
lo0/v4        static   ok      127.0.0.1/8
net0/v4       static   ok      192.168.35.10/24
lo0/v6        static   ok      ::1/128
tun0/_a       static   ok      2002:c0a8:57bc::1/64
tun0/v6       static   ok      2002:c0a8:230a::2/16
tun0/v6a      static   ok      2002:c0a8:230a::3/16

# vi /etc/inet/ndpd.conf
```

```
if bge0 AdvSendAdvertisements 1
2002:c0a8:57bc::1/64 bge0

# ipadm set-prop -p forwarding=on ipv6
```

6to4 トンネルでは IPv6 アドレスの接頭辞は 2002 です。

▼ 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。『Troubleshooting Network Issues』の「Security Issues When Tunneling to a 6to4 Relay Router」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。

- 113 ページの「IP トンネルを作成および構成する方法」の説明に従ってサイトの 6to4 ルーターを構成する
- 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 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。

2 **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
```

3 **(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 6–5 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
```

▼ IP トンネルの構成を変更する方法

- トンネルの構成を変更します。

  # dladm modify-iptun -a [local|remote]=*addr,...* *tunnel-link*

  既存のトンネルのタイプは変更できません。したがって、-T *type* オプションはこのコマンドでは許可されません。次のトンネルパラメータだけが変更できます。

  `-a [local|remote]=`*address,...*

  ローカルアドレスとリモートトンネルアドレスに対応するリテラル IP アドレスまたはホスト名を指定します。トンネルのタイプに応じて、アドレスを 1 つだけ指定するか、ローカルアドレスとリモートアドレスの両方を指定します。ローカルアドレスとリモートアドレスの両方を指定する場合は、それらのアドレスをコンマで区切ります。

  - `IPv4` トンネルが機能するためには、ローカルとリモートの IPv4 アドレスが必要です。
  - `IPv6` トンネルが機能するためには、ローカルとリモートの IPv6 アドレスが必要です。
  - `6to4` トンネルが機能するためには、ローカル IPv4 アドレスが必要です。

  永続的な IP トンネルデータリンクの構成でホスト名をアドレスとして使用した場合、それらのホスト名が構成ストレージに保存されます。次回以降のシステムブート時に、トンネル作成時に使用された IP アドレスとは異なる IP アドレスに名前が解決された場合、トンネルは新しい構成を取得します。

  トンネルのローカルとリモートのアドレスを変更する場合には、変更しているトンネルのタイプとこれらのアドレスが矛盾しないことを確認してください。

注 – トンネルリンクの名前を変更する場合、`modify-iptun` サブコマンドを使用しないでください。代わりに、`dladm rename-link` を使用します。

 # dladm rename-link *old-tunnel-link new-tunnel-link*

同様に、`hoplimit` や `encaplimit` などのトンネルプロパティーを変更するために `modify-iptun` コマンドを使用しないでください。代わりに、`dladm set-linkprop` コマンドを使用してそれらのプロパティーの値を設定します。

例 6–6 トンネルのアドレスとプロパティーの変更

この例は 2 つの手順から構成されています。まず、IPv4 トンネル vpn0 のローカルとリモートのアドレスが一時的に変更されます。あとでシステムがリブートされるときに、このトンネルはまた元のアドレスを使用するようになります。2 番目の手順では、vpn0 の hoplimit を 60 に変更します。

```
# dladm modify-iptun -t -a local=10.8.48.149,remote=192.1.2.3 vpn0

# dladm set-linkprop -p hoplimit=60 vpn0
```

▼ IP トンネルの構成を表示する方法

- **IP トンネルの構成を表示します。**

  # dladm show-iptun [-p] -o *fields* **[***tunnel-link***]**

  次のオプションがこのコマンドで使用できます。

  `-p` マシンによる解析が可能な形式で情報を表示します。この引数はオプションです。

  `-o` *fields* 特定のトンネル情報を提供するフィールドのうち、選択されたフィールドを表示します。

  tunnel-link 構成情報を表示するトンネルを指定します。この引数はオプションです。トンネル名が省略された場合、このコマンドはシステム上のすべてのトンネルの情報を表示します。

例 6–7 すべてのトンネルに関する情報の表示

この例では、システム上に存在しているトンネルは、1 つだけです。

```
# dladm show-iptun
LINK     TYPE     FLAGS     LOCAL          REMOTE
tun0     6to4     --        192.168.35.10  --
vpn0     ipv4     --        10.8.48.149    192.1.2.3
```

例 6–8 選択されたフィールドをマシンによる解析が可能な形式で表示する

この例では、トンネル情報を含む特定のフィールドのみが表示されています。

```
# dladm show-iptun -p -o link,type,local
tun0:6to4:192.168.35.10
vpn0:ipv4:10.8.48.149
```

▼ IP トンネルのプロパティーを表示する方法

- トンネルリンクのプロパティーを表示します。

  # dladm show-linkprop [-c] [-o *fields***] [***tunnel-link***]**

  次のオプションがこのコマンドで使用できます。

  `-c` マシンによる解析が可能な形式で情報を表示します。この引数はオプションです。

  `-o` *fields* リンクのプロパティーに関する特定の情報を提供する、選択されたフィールドを表示します。

  tunnel-link プロパティーに関する情報を表示するトンネルを指定します。この引数はオプションです。トンネル名が省略された場合、このコマンドはシステム上のすべてのトンネルの情報を表示します。

例 6–9 トンネルのプロパティーの表示

この例では、トンネルのすべてのリンクプロパティーを表示する方法を示します。

```
# dladm show-linkprop tun0
LINK     PROPERTY     PERM     VALUE     DEFAULT     POSSIBLE
tun0     autopush     --       --        --          --
tun0     zone         rw       --        --          --
tun0     state        r-       up        up          up,down
tun0     mtu          r-       65515     --          576-65495
tun0     maxbw        rw       --        --          --
tun0     cpus         rw       --        --          --
tun0     priority     rw       high      high        low,medium,high
tun0     hoplimit     rw       64        64          1-255
```

▼ IP トンネルを削除する方法

1. トンネル上に構成されている IP インタフェースを、インタフェースのタイプに応じて適切な構文を使用して **unplumb** します。

   # ipadm delete-ip *tunnel-link*

   注 – トンネルを正常に削除するには、そのトンネル上で既存の IP インタフェースが plumb されていてはいけません。

   ---

2. IP トンネルを削除します。

   # dladm delete-iptun *tunnel-link*

このコマンドのオプションは -t だけです。これにより、トンネルは一時的に削除されます。システムをリブートするときに、トンネルが復元されます。

例 6–10 IPv6 インタフェースが構成された IPv6 トンネルの削除

この例では、永続的なトンネルが永遠に削除されます。

```
# ipadm delete-ip ip6.tun0
# dladm delete-iptun ip6.tun0
```

◆ ◆ ◆ 第 7 章

7 IPv4 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。

この章では、次の内容について説明します。

TCP/IP 構成ファイル

ネットワークでは、そのネットワークの動作方法を制御するさまざまなファイルやデータベース内に構成情報が格納されます。このセクションでは、これらのファイルについて簡単に説明します。一部のファイルでは、ネットワークに対して変更を実装した際に更新や維持が必要となります。その他のファイルでは、管理はほとんどあるいはまったく必要ありません。

`/etc/defaultrouter` このファイルには、ネットワークに直接接続されているルーターの IP インタフェース名が含まれます。システム内のこのファイルが存在はオプションです。このファイルが存在する場合、システムは静的ルーティングをサポートするよう構成されます。

`/etc/inet/hosts` このファイルには、ネットワーク内での IPv4 アドレスと、それらのアドレスの構成先となる対応するインタフェース名が含まれます。NIS または DNS ネームサービス、あるいは LDAP ディレクトリサービスを使用している場合には、`hosts.byname` のような、サーバー内に存在する別のデータベース内にホスト情報が格納されます。詳細

	は、『Oracle Solaris 11.1 でのネームサービスおよびディレクトリサービスの作業』を参照してください。
/etc/inet/netmasks	このファイルには、192.168.0.0 などのネットワーク番号と、255.255.255.0 のような、そのネットワーク番号のネットマスク情報が含まれます。NIS または LDAP を使用するネットワークでは、この情報はサーバーのネットマスクデータベース内に格納されます。詳細は、netmasks(4) のマニュアルページを参照してください。
/etc/bootparams	このファイルには、ネットワーククライアントモードでブートするように構成されたシステムのブートプロセスを決定するパラメータが含まれます。詳細は、38 ページの「システム構成モードの設定」を参照してください。ローカルファイルモードを使用しない場合、このファイルは、ネームサービスが使用する bootparams データベースを作成するための基礎となります。このファイルの内容や形式に関する具体的な情報を入手するには、bootparams(4) のマニュアルページを参照してください。
/etc/ethers	このファイルは、ホスト名とその MAC アドレスを関連付けます。このファイルは、各システムがネットワーククライアントとして構成されたネットワーク内で使用される ethers データベースを作成するための基礎となります。詳細は、ethers(4) のマニュアルページを参照してください。
/etc/inet/networks	このファイルは、ネットワーク名とネットワーク番号を関連付けます。データベース内の各エントリをさらに明確化するためのコメントも追加できます。アプリケーションはこのファイルを使用することで、ネットワーク番号の代わりにネットワーク名を使用したり表示したりできます。たとえば、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。このファイルには、ルーター経由でローカルネットワークに接続されているすべてのサブネットワークを含める必要があります。詳細は、networks(4) のマニュアルページを参照してください。
/etc/inet/protocols	このファイルには、システムにインストールされている TCP/IP プロトコルとそのプロトコル番号の一覧が含まれます。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。詳細は、protocols(4) のマニュアルページを参照してください。

`/etc/inet/services` このファイルには、TCP および UDP サービスの名前とそれぞれの既知のポート番号の一覧が含まれます。このリストは、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。通常は、このファイルは管理を必要としません。詳細は、`services(4)` のマニュアルページを参照してください。

inetd インターネットサービスデーモン

`inetd` デーモンは、システムのブート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 コマンドの使用手順については、『Oracle Solaris 11.1 でのサービスと障害の管理』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、58 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、127 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。

name-service/switch SMF サービス

`name-service/switch` SMF サービスは、構成情報用のネットワークデータベースの検索順を定義します。デフォルトドメインなど、以前は構成ファイル内に格納されていた一部のネットワーク構成情報が変換され、この SMF サービスのプロパティーと

なりました。この SMF サービスのプロパティーによって、システム上でのネームサービスの実装が決まります。プロパティーの一覧を次に示します。

```
% svccfg -s name-service/switch listprop config
config                      application
config/value_authorization  astring             solaris.smf.value.name-service.switch
config/default              astring             files
config/password             astring             "files nis"
config/group                astring             "files nis"
config/host                 astring             "files dns nis"
config/network              astring             "nis [NOTFOUND=return] files"
config/protocol             astring             "nis [NOTFOUND=return] files"
config/rpc                  astring             "nis [NOTFOUND=return] files"
config/ether                astring             "nis [NOTFOUND=return] files"
config/netmask              astring             "files nis"
config/bootparam            astring             "nis [NOTFOUND=return] files"
config/publickey            astring             "nis [NOTFOUND=return] files"
config/netgroup             astring             nis
config/automount            astring             "files nis"
config/alias                astring             "files nis"
config/service              astring             "files nis"
config/printer              astring             "user nis"
config/auth_attr            astring             "files nis"
config/prof_attr            astring             "files nis"
config/project              astring             "files nis"
```

各プロパティーに設定された値により、パスワード、別名、ネットワークマスクなどの、ネットワークユーザーに影響を及ぼす情報を、どのネームサービス内で検索するかが決まります。この例では、自動マウントとパスワードのプロパティーは、`files` と `nis` に設定されています。したがって、自動マウント情報とパスワード情報はファイルと NIS サービスから取得されます。

あるネームサービスから別のネームサービスに変更するには、`name-service/switch` SMF サービスの対応するプロパティーを設定することで、その選択したネームサービスを有効にする必要があります。

たとえば、ネットワーク上の LDAP ネームサービスを使用する必要があるとします。この SMF サービスの次のプロパティーを構成する必要があります。

- `config/default` でファイルと LDAP が使用されるように設定する必要があります。
- `config/host` でファイルと DNS が使用されるように設定する必要があります。
- `config/netgroup` で LDAP が使用されるように設定する必要があります。
- `config/printer` でユーザー、ファイル、および LDAP が使用されるように設定する必要があります。

したがって、次のコマンドを入力してこれらのプロパティーを正しく設定する必要があります。

```
# svccfg -s name-service/switch setprop config/default = astring: '"files ldap"'
# svccfg -s name-service/switch setprop config/host = astring: '"files dns"'
# svccfg -s name-service/switch setprop config/netgroup = astring: '"ldap"'
# svccfg -s name-service/switch setprop config/printer = astring: '"user files ldap"'
# svccfg -s name-service/switch:default refresh
```

ネームサービススイッチの詳細は、『Oracle Solaris 11.1 でのネームサービスおよびディレクトリサービスの作業』を参照してください。

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、ネットワークで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。

- 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は、`/etc/inet` および `/etc` ディレクトリ内の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 のブートファイルやデータファイルは、ネットワークデータベースに直接対応しません。

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の情報については、『Oracle Solaris 11.1 でのネームサービスおよびディレクトリサービスの作業』を参照してください。

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 7–1 および表 7–2 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 によって実行されます。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、ルーターからルーティング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。

Oracle Solaris のルーティングプロトコルの表

次の表では、Oracle Solaris でサポートされているすべてのルーティングプロトコルの一覧を示します

表 7–1 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	手順
ルーティング情報プロトコル (RIP)	`in.routed`	IPv4 パケットのルーティングおよびルーティングテーブルの維持を行う IGP	43 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	52 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 54 ページの「単一インタフェースシステムで動的ルーティングを有効にする方法」
RIPng (Routing Information Protocol, next generation) プロトコル	`in.ripngd`	IPv6 パケットのルーティングおよびルーティングテーブルの維持を行う IGP	66 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	63 ページの「IPv6 インタフェースの構成」

次の表に、Oracle Solaris でもサポートされているオープンソースの Quagga ルーティングプロトコル群を一覧表示します。

表 7–2 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 パケットのルーティングおよび近傍へのルーティングテーブルの通知を行います。
RIPng	`ripngd`	IPv6 距離ベクトル型 IGP。IPv6 パケットのルーティングおよびルーティングテーブルの維持を行います。
OSPF (Open Shortest Path First) プロトコル	`ospfd`	パケットのルーティング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ルーティングのための IPv4 および IPv6 EGP。

◆ ◆ ◆ 第 8 章

IPv6 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 対応ネットワークを構成するタスクについては、第 4 章「ネットワークでの IPv6 の有効化」を参照してください。IP トンネルのあらゆる情報については、第 6 章「IP トンネルの構成」を参照してください。

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。

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 8–1 /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 8–2 /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。

表 8–2 /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
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、66 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、72 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。
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、69 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、69 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。

表 8–2 /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
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、69 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、69 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 8–3 /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 8–1 /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。

```
# 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
```

例 8–1 /etc/inet/ndpd.conf ファイル (続き)

```
#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
```

/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 8–4 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、100 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、138 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 8–4　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、`ipadm` 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ることができます。`ipadm` コマンドの詳細は、`ipadm(1M)` のマニュアルページを参照してください。
- 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、108 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、113 ページの「IP トンネルを作成および構成する方法」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 8–2 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 8–3 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 8–4 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 `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、84 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、96 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、88 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、92 ページの「traceroute コマンドによるルーティング情報の表示」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。

-a	ステートレスおよびステートフルアドレス自動構成をオフに設定します。
-d	デバッグを有効にします。
-f *config-file*	デフォルトの /etc/inet/ndpd.conf ファイルではなく、構成を読み取るファイルを指定します。
-t	すべての発信および着信パケットの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 構成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 8–1 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 ルーティングを有効にした場合、`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 UDP ポート番号を指定します。

`-P` — ポイズンリバースの使用を打ち切ります。

`-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。

`-t` — 送受信されたすべてのパケットを標準出力に出力します。

`-v` — ルーティングテーブルへのすべての変更 (タイムスタンプを含む) を標準出力に出力します。

`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 11.1 でのサービスと障害の管理』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、58 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、『Programming Interfaces Guide』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)。

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。

- 150 ページの「近傍検索と ARP および関連する IPv4 プロトコルとの比較」

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。

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。

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。

索引

H

I

U

V

あ

い

う

え

か

ろ