



IPv6 の管理

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

Part No: 817-1176-10
2003 年 4 月

Copyright 2003 Sun Microsystems, Inc. 4150 Network Circle, Santa Clara, CA 95054 U.S.A. All rights reserved.

本製品およびそれに関連する文書は著作権法により保護されており、その使用、複製、頒布および逆コンパイルを制限するライセンスのもとにおいて頒布されます。サン・マイクロシステムズ株式会社の書面による事前の許可なく、本製品および関連する文書のいかなる部分も、いかなる方法によっても複製することが禁じられます。

本製品の一部は、カリフォルニア大学からライセンスされている Berkeley BSD システムに基づいていることがあります。UNIX は、X/Open Company, Ltd. が独占的にライセンスしている米国ならびに他の国における登録商標です。フォント技術を含む第三者のソフトウェアは、著作権により保護されており、提供者からライセンスを受けているものです。

Federal Acquisitions: Commercial Software—Government Users Subject to Standard License Terms and Conditions.

本製品に含まれる HG 明朝 L、HG-MincyoL-Sun、HG ゴシック B、および HG-GothicB-Sun は、株式会社リコーがリコービマジクス株式会社からライセンス供与されたタイプフェイスマスタをもとに作成されたものです。HG 平成明朝体 W3@X12 は、株式会社リコーが財団法人日本規格協会からライセンス供与されたタイプフェイスマスタをもとに作成されたものです。フォントとして無断複製することは禁止されています。

Sun、Sun Microsystems、docs.sun.com、AnswerBook、AnswerBook2 は、米国およびその他の国における米国 Sun Microsystems, Inc. (以下、米国 Sun Microsystems 社とします) の商標もしくは登録商標です。

サンのロゴマークおよび Solaris は、米国 Sun Microsystems 社の登録商標です。

すべての SPARC 商標は、米国 SPARC International, Inc. のライセンスを受けて使用している同社の米国およびその他の国における商標または登録商標です。SPARC 商標が付いた製品は、米国 Sun Microsystems 社が開発したアーキテクチャに基づくものです。

OPENLOOK、OpenBoot、JLE は、サン・マイクロシステムズ株式会社の登録商標です。

Wnn は、京都大学、株式会社アステック、オムロン株式会社で共同開発されたソフトウェアです。

Wnn6 は、オムロン株式会社、オムロンソフトウェア株式会社で共同開発されたソフトウェアです。© Copyright OMRON Co., Ltd. 1995-2000. All Rights Reserved. © Copyright OMRON SOFTWARE Co., Ltd. 1995-2002 All Rights Reserved.

「ATOK」は、株式会社ジャストシステムの登録商標です。

「ATOK Server/ATOK12」は、株式会社ジャストシステムの著作物であり、「ATOK Server/ATOK12」にかかる著作権その他の権利は、株式会社ジャストシステムおよび各権利者に帰属します。

本製品に含まれる郵便番号辞書 (7 桁/5 桁) は郵政事業庁が公開したデータを元に制作された物です (一部データの加工を行なっています)。

本製品に含まれるフェイスマーク辞書は、株式会社ビレッジセンターの許諾のもと、同社が発行する『インターネット・パソコン通信フェイスマークガイド '98』に添付のものを使用しています。© 1997 ビレッジセンター

Unicode は、Unicode, Inc. の商標です。

本書で参照されている製品やサービスに関しては、該当する会社または組織に直接お問い合わせください。

OPEN LOOK および Sun Graphical User Interface は、米国 Sun Microsystems 社が自社のユーザおよびライセンス実施権者向けに開発しました。米国 Sun Microsystems 社は、コンピュータ産業用のビジュアルまたはグラフィカル・ユーザインタフェースの概念の研究開発における米国 Xerox 社の先駆者としての成果を認めるものです。米国 Sun Microsystems 社は米国 Xerox 社から Xerox Graphical User Interface の非独占的ライセンスを取得しており、このライセンスは米国 Sun Microsystems 社のライセンス実施権者にも適用されます。

DtComboBox ウィジェットと DtSpinBox ウィジェットのプログラムおよびドキュメントは、Interleaf, Inc. から提供されたものです。(© 1993 Interleaf, Inc.)

本書は、「現状のまま」をベースとして提供され、商品性、特定目的への適合性または第三者の権利の非侵害の黙示の保証を含みそれに限定されない、明示的であるか黙示的であるかを問わない、なんらの保証も行われぬものとします。

本製品が、外国為替および外国貿易管理法 (外為法) に定められる戦略物資等 (貨物または役務) に該当する場合、本製品を輸出または日本国外へ持ち出す際には、サン・マイクロシステムズ株式会社の事前の書面による承諾を得ることのほか、外為法および関連法規に基づく輸出手続き、また場合によっては、米国商務省または米国所轄官庁の許可を得ることが必要です。

原典: IPv6 Administration Guide

Part No: 817-0573-10

Revision A



030220@5115



目次

はじめに	7
1 IPv6 (概要)	11
IPv6 の機能	11
IPv6 のヘッダーと拡張機能	12
ヘッダーフォーマット	12
拡張ヘッダー	13
IPv6 アドレス指定	14
ユニキャストアドレス	16
集約グローバルユニキャストアドレス	16
ローカル用アドレス	17
組み込み IPv4 アドレスを伴った IPv6 アドレス	18
エニーキャストアドレス	19
マルチキャストアドレス	19
IPv6 のルーティング	20
IPv6 の近傍検索	21
ルーター通知	22
ルーター通知プレフィックス	22
ルーター通知メッセージ	23
近傍要請と不到達	23
IPv4 との比較	24
IPv6 ステートレスアドレス自動設定	26
ステートレス自動設定の条件	26
ステートフル自動設定モデル	26
ステートレス方式とステートフル方式をいつ使用するか	26
重複アドレスの検出アルゴリズム	27

自動設定プロセス	27
IPv6 モビリティ (移動性) サポート	29
IPv6 サービス品質 (QoS) 機能	30
フローラベル	30
トラフィッククラス	32
IPv6 セキュリティの強化	32
IPv4 ネットワーク経由の 6to4 トンネル	33
6to4 ルーティングの詳細	34
2 IPv6 の管理 (手順)	35
IPv6 ノードを有効にする	35
IPv6 ノードを有効にする (作業マップ)	36
▼ ノード上の IPv6 を有効にする方法	36
▼ Solaris IPv6 ルーターの設定方法	37
▼ 6to4 ルーターの設定方法	38
▼ NIS と NIS+ に対する IPv6 アドレスの追加方法	41
▼ DNS に対する IPv6 アドレスの追加方法	42
IPv6 の監視	43
IPv6 の監視 (作業マップ)	43
▼ インタフェースアドレス割り当ての表示方法	44
▼ ネットワーク状態の表示方法	45
▼ IPv6 関連コマンドの出力表示の制御方法	49
▼ IPv6 ネットワークトラフィックの監視方法	50
▼ すべてのマルチホームホストアドレスの探査方法	50
▼ すべてのルーターのトレース方法	51
IP 内 IP トンネルの設定	52
IP 内 IP トンネルを設定する (作業マップ)	52
▼ IPv4 トンネルによる IPv6 の設定方法	53
▼ IPv6 トンネル経由の IPv6 の設定方法	54
▼ IPv6 トンネル経由の IPv4 の設定方法	55
▼ IPv4 トンネル経由の IPv4 の設定方法	55
▼ トンネルインタフェースで通知するためのルーターの設定方法	56
▼ 6to4 リレールーターとの間の 6to4 トンネルの設定方法	57
IPv6 ネームサービス情報の表示	58
IPv6 ネームサービス情報を表示する (作業マップ)	59
▼ IPv6 ネームサービス情報の表示方法	59
▼ DNS IPv6 PTR レコードの正確な更新の確認方法	60

	▼ NIS による IPv6 情報の表示方法	60
	▼ NIS+ による IPv6 情報の表示方法	61
	▼ ネームサービスに依存しない IPv6 情報の表示方法	61
3	IPv6 のファイルおよびコマンド (リファレンス)	63
	Solaris IPv6 実装の概要	63
	IPv6 ネットワークインタフェース設定ファイル	64
	IPv6 インタフェース設定ファイルのエントリ	65
	IPv6 向けに追加または変更されたユーティリティ	65
	ifconfig ユーティリティに対する IPv6 拡張機能	66
	6to4relay コマンド	68
	複数のネットワークインタフェースがあるノード	69
	IPv4 の動作	69
	IPv6 の動作	70
	IPv6 デーモン	70
	in.ndpd デーモン	70
	in.ripngd デーモン	73
	inetd インターネットサービスデーモン	73
	既存のユーティリティに対する IPv6 拡張機能	75
	netstat (1M)	75
	snoop (1M)	75
	route (1M)	76
	ping (1M)	76
	traceroute (1M)	76
	表示出力の制御	76
	IPv6 の Solaris トンネルインタフェース	77
	Solaris ネームサービスに対する IPv6 拡張機能	79
	/etc/inet/ipnodes ファイル	80
	IPv6 の NIS 拡張機能	81
	IPv6 の NIS+ 拡張機能	81
	IPv6 の DNS 拡張機能	81
	nsswitch.conf ファイルへの変更	81
	ネームサービスコマンドの変更	82
	NFS と RPC による IPv6 のサポート	83
	ATM 経由の IPv6 サポート	83

4 IPv4 から IPv6 への移行 (リファレンス)	85
移行条件	85
標準移行ツール	86
デュアルスタックの実装	86
ネームサービスの設定	87
IPv4 互換アドレスフォーマットの使用	88
トンネル機構	89
アプリケーションとの対話	90
IPv4 と IPv6 の相互運用性	91
サイト移行のシナリオ	92
移行機構としての 6to4	93
6to4 トンネルの構成	93
6to4 派生のアドレス指定	95
6to4 トンネルを介したパケットフロー	97
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	98
6to4 ルーターの既知の問題	101
その他の移行機構	102
用語集	103
索引	109

はじめに

『IPv6 の管理』では、Solaris™ オペレーティング環境にインストールされた IPv6 フレームワークの構成および管理について説明します。このマニュアルでは、SunOS™ 5.9 オペレーティングシステムがすでにインストールされているものとします。さらに、使用するネットワークソフトウェアが設定されているものとします。SunOS 5.9 オペレーティングシステムは Solaris 製品ファミリの一部であり、Solaris 共通デスクトップ環境 (CDE) などが含まれます。また、SunOS 5.9 は、AT&T System V リリース 4 オペレーティングシステムに準拠しています。

注 - Solaris オペレーティング環境は、SPARC® と x86 の 2 種類のハードウェア (プラットフォーム) で動作します。また、Solaris オペレーティング環境は、64 ビットアドレス空間でも 32 ビットアドレス空間でも動作します。このマニュアルで説明する情報は、章、節、注、箇条書き、図、表、例、またはコード例において特に明記しない限り、両方のプラットフォームおよびアドレス空間に該当します。

対象読者

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

内容の紹介

第1章では、IPv6として知られる新しいインターネットプロトコルの概要について説明します。

第2章では、サイトでIPv6を有効にするための作業として、IPv6ルーター、6to4ルーター、ネームサービス用のIPv6アドレスなどの設定、ルーター間に各種のトンネルを作成する方法、IPv6に対応したコマンドを使って診断する方法、IPv6ネームサービス情報の表示方法などについて説明します。

第3章では、IPv6のSolaris実装に伴う概念について説明します。

第4章では、IPv4からIPv6への移行の説明と、標準的な手法の概要について説明しています。

用語集では、主要なIPサービス用語の定義を提供します。

Sunのオンラインマニュアル

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

表記上の規則

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

表 P-1 表記上の規則

字体または記号	意味	例
AaBbCc123	コマンド名、ファイル名、ディレクトリ名、画面上のコンピュータ出力、コード例を示します。	.login ファイルを編集します。 ls -a を使用してすべてのファイルを表示します。 system%
AaBbCc123	ユーザーが入力する文字を、画面上のコンピュータ出力と区別して示します。	system% su password:
AaBbCc123	変数を示します。実際に使用する特定の名前または値で置き換えます。	ファイルを削除するには、rm <i>filename</i> と入力します。
『 』	参照する書名を示します。	『コードマネージャ・ユーザーズガイド』を参照してください。
「 」	参照する章、節、ボタンやメニュー名、強調する単語を示します。	第 5 章「衝突の回避」を参照してください。 この操作ができるのは、「スーパーユーザー」だけです。
\	枠で囲まれたコード例で、テキストがページ行幅を超える場合に、継続を示します。	sun% grep `^#define \ XV_VERSION_STRING`

コード例は次のように表示されます。

■ C シェル

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

■ C シェルのスーパーユーザー

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

■ Bourne シェルおよび Korn シェル

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

■ Bourne シェルおよび Korn シェルのスーパーユーザー

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

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

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

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

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

一般規則

- このマニュアルでは、「x86」という用語は、Intel 32 ビット系列のマイクロプロセッサチップ、および AMD が提供する互換マイクロプロセッサチップを意味します。

第 1 章

IPv6 (概要)

インターネットプロトコルバージョン 6 (IPv6 : Internet Protocol version 6) はインターネットプロトコル (IP) の新しいバージョンです。IPv6 は、現在の IPv4 から飛躍的に進歩しています。IPv4 から IPv6 には無理なく移行することができます。規定されている移行メカニズムを使用することにより、現在の運用に混乱を生じることなく IPv6 ネットワークを展開できます。IPv6 ではアドレス領域が増加しています。また、シンプルになったヘッダーフォーマット、認証とプライバシーのサポート、アドレス割り当ての自動設定を採用し、サービス品質を一新してインターネット機能を強化しました。

この章では、以下の内容について説明します。

- 11 ページの「IPv6 の機能」
- 12 ページの「IPv6 のヘッダーと拡張機能」
- 14 ページの「IPv6 アドレス指定」
- 20 ページの「IPv6 のルーティング」
- 21 ページの「IPv6 の近傍検索」
- 26 ページの「IPv6 ステートレスアドレス自動設定」
- 29 ページの「IPv6 モビリティ (移動性) サポート」
- 30 ページの「IPv6 サービス品質 (QoS) 機能」
- 32 ページの「IPv6 セキュリティの強化」
- 33 ページの「IPv4 ネットワーク経由の 6to4 トンネル」

IPv6 の機能

IPv4 から IPv6 への変更内容は、次のように分類できます。

- 拡張されたルーティングとアドレス指定機能 – IPv6 では IP アドレスサイズを 32 ビットから 128 ビットに拡大して、サポートするアドレス指定階層を広げています。さらに、IPv6 はアドレス可能なノード数を増やし、アドレスの自動設定も容易にしています。

スコープフィールドの追加により、マルチキャストアドレスに対するマルチキャストルーティングのスケラビリティを強化しました。

エニーキャストアドレスという新しいタイプのアドレスを定義しました。エニーキャストアドレスは、ノードセットを識別します。エニーキャストアドレスに送信されたパケットはノードの1つに配信されます。IPv6 ソースルートではエニーキャストアドレスを使用して、ノードでトラフィックフローのパスを制御できます。

- ヘッダーフォーマットの簡略化 – IPv4 ヘッダーフィールドが一部削除されたり、オプションになったりしました。この変更によってパケット処理の共通部分の処理コストが削減されます。また、アドレスのサイズは増えましたが、IPv6 ヘッダーの帯域幅コストは可能な限り少なくなりました。IPv6 アドレスの長さは、IPv4 アドレスの4倍ですが、IPv6 ヘッダーのサイズはIPv4の2倍に抑えられています。
- オプションサポートの強化 – IP ヘッダーオプションのコード化の方法を変更したため、転送効率が改善されました。また、オプションの長さに関する制限が緩和されています。さらに、将来新しいオプションを導入する際の柔軟性が高くなりました。
- サービス品質の機能 – この機能は、送信側が特別な処理を必要とする特定のトラフィックフローに属するパケットのラベルを指定できます。たとえば、デフォルト以外の品質サービスやリアルタイムサービスなどです。
- 認証機能と機密機能 – IPv6 には認証、データの完全性、機密性をサポートする拡張機能の定義が組み込まれています。

IPv6 のヘッダーと拡張機能

IPv6 プロトコルは、基本 IPv6 ヘッダー、IPv6 拡張ヘッダーを含むヘッダーセットを定義します。

ヘッダーフォーマット

図 1-1 は、IPv6 ヘッダーに使用される要素とその順序を示します。

バージョン	トラフィッククラス	フローラベル	
ペイロードの長さ	次のヘッダー	ホップ制限	
ソースアドレス			
宛先アドレス			

図 1-1 IPv6 ヘッダーフォーマット

次に各ヘッダーフィールドの機能について説明します。

- バージョン - 4 ビットインターネットプロトコルバージョン番号。IPv6 では 6
- トラフィッククラス - 8 ビットトラフィッククラスフィールド。32 ページの「トラフィッククラス」を参照してください。
- フローラベル - 20 ビットフィールド。30 ページの「IPv6 サービス品質 (QoS) 機能」を参照してください。
- ペイロードの長さ - オクテット単位で表す 16 ビット符号なし整数。IPv6 ヘッダーに続くパケットの残り
- 次のヘッダー - 8 ビットセクタ。IPv6 ヘッダーのすぐ後ろに続くヘッダーのタイプを識別する。IPv4 プロトコルフィールドとして同じ値を使用する。13 ページの「拡張ヘッダー」を参照してください。
- ホップ制限 - 8 ビット符号なし整数。パケットを送信するノードごとに値が 1 ずつ減る。ホップ制限がゼロになるとパケットが廃棄される
- ソースアドレス - 128 ビット。パケットの初期送信側のアドレス。14 ページの「IPv6 アドレス指定」を参照してください。
- 宛先アドレス - 128 ビット。パケットの予定受信側のアドレス。オプションのルーティングヘッダーがある場合、必ずしも受信側とは限らない

拡張ヘッダー

IPv6 には、IPv4 から強化されたオプション機能があります。IPv6 オプションは、IPv6 ヘッダーとトランスポート層の間の独立した拡張ヘッダーにあります。パケットが最終的な宛先に到着するまで、その配送パスに存在するルーターは、ほとんどの場

合 IPv6 拡張ヘッダを確認または処理しません。そのため、オプションがあるパケットを処理するルーターの性能が大幅に改善されました。IPv4 では、オプションがある場合、ルーターですべてのオプションを調べる必要がありました。

IPv4 オプションとは異なり、IPv6 拡張機能ヘッダーの長さを任意に指定できます。またパケットに組み込むことのできるオプションの合計数が 40 バイト以内に限定されない点があります。この機能とその処理方法によって、IPv4 では非現実的であった機能を IPv6 オプションが使用できるようになりました。その良い例が IPv6 認証オプションとセキュリティカプセル化オプションです。

後続のオプションヘッダー (およびそのあとのトランスポートプロトコル) を処理する際の性能を強化するため、IPv6 オプションは常に 8 オクテットの整数倍の長さです。これにより、後続ヘッダーのバイト境界が維持されています。

次の IPv6 拡張ヘッダーが現在、定義されています。

- ルーティング – 拡張ルーティング (IPv4 ルーズソースルートにあたる)
- 断片化 – 断片化および再結合
- 認証 – 整合性および認証、セキュリティ
- カプセル化 – 機密性
- ホップバイホップオプション – ホップごとの処理が必要な特別なオプション
- 宛先オプション – 宛先ノードが判断するオプション情報

IPv6 アドレス指定

IPv6 アドレスは 128 ビット長であり、個々のインタフェースおよびインタフェースセットの識別子です。IPv6 アドレスはすべて、ホストやルーターではなくインタフェースに割り当てることができます (IPv4 ではホストやルーターに割り当てられる)。各インタフェースの所属先は 1 つのノードだけなので、インタフェースのユニキャストアドレスは、そのノードの識別子として使用できます。1 つのインタフェースには、任意のタイプの複数の IPv6 アドレスを割り当てることができます。

IPv6 アドレスには、ユニキャスト、エニーキャスト、マルチキャストの 3 種類のタイプがあります。

- ユニキャストアドレスは、1 つのインタフェースを識別する
- エニーキャストアドレスは、インタフェースのセットを識別する。エニーキャストアドレスに送信されたパケットはそのセットのメンバーの 1 つに配信される
- マルチキャストアドレスは、インタフェースのグループを識別する。マルチキャストアドレスに送信されるパケットは、グループにあるすべてのインタフェースに配信される。

IPv6 では、ブロードキャストアドレスの代わりにマルチキャストアドレスが使われます。

IPv4 が 32 ビットであるのに対し、IPv6 は 128 ビットと 4 倍のビット数のアドレスをサポートします。したがって、計算上はそのアドレス領域は IPv4 のアドレス領域の 40 億倍の大きさになります。実際にはアドレスの割り当てとルーティングでは階層を作成する必要があり、アドレス領域の利用効率が減少するため、結果として、利用できるアドレス数は減少します。ただし当面は、IPv6 で提供するアドレス領域で十分です。

アドレスの先頭ビットでは IPv6 アドレスのタイプを指定します。この先頭ビットがある可変長フィールドをフォーマットプレフィックス (FP) といいます。次の表は、これらのプレフィックス (接頭辞) の初期割り当てです。

表 1-1 フォーマットプレフィックスの割り当て

割り当て	プレフィックス (バイナリ)	アドレス領域の端数
予約	0000 0000	1/256
割り当てなし	0000 0001	1/256
NSAP 割り当てに予約	0000 001	1/128
IPX 割り当てに予約	0000 010	1/128
割り当てなし	0000 011	1/128
割り当てなし	0000 1	1/32
割り当てなし	0001	1/16
集約グローバルユニキャストアドレス	001	1/8
割り当てなし	010	1/8
割り当てなし	011	1/8
ニュートラル相互接続ベースユニキャストアドレスに予約	100	1/8
割り当てなし	101	1/8
割り当てなし	110	1/8
割り当てなし	1110	1/16
割り当てなし	1111 0	1/32
割り当てなし	1111 10	1/64
割り当てなし	1111 110	1/128
割り当てなし	1111 1110 0	1/512
リンクローカル用アドレス	1111 1110 10	1/1024
サイトローカル用アドレス	1111 1110 11	1/1024

表 1-1 フォーマットプレフィックスの割り当て (続き)

割り当て	プレフィックス(バイナリ)	アドレス領域の端数
マルチキャストアドレス	1111 1111	1/256
6to4 ルーターアドレスに予約	1000 0000 0000 10	

割り当てには、集約グローバルユニキャストアドレス、ローカル用アドレス、マルチキャストアドレスの直接割り当てがサポートされています。NSAP (ネットワークサービスアクセスポイント) アドレス、IPX (相互ネットワークパケット交換プロトコル) アドレス、ニュートラル相互接続アドレスには空間が予約されています。残りのアドレス領域は将来用に割り当てなしになっています。残りのアドレス領域は、既存の領域の拡張部分に利用できます。たとえば、集約グローバルユニキャストアドレスへの追加として使用されます。残りの領域は、新規用としても使用されます。たとえば、個別のロケータと個別の識別子として使用されます。なお、エニーキャストアドレスはユニキャストアドレス空間の範囲外に割り当てられるため、ここには示していません。

初期設定で、アドレス領域の約 15 パーセントが割り当てられます。残りの 85 パーセントは将来用に予約されています。

ユニキャストアドレス

IPv6 ユニキャストアドレスの割り当て形式は、次のとおりです。

- 集約グローバルユニキャストアドレス
- ニュートラル相互接続ユニキャストアドレス
- NSAP アドレス
- IPX 階層アドレス
- サイトローカル用アドレス
- リンクローカル用アドレス
- IPv4 対応ホストアドレス

その他のアドレスタイプは、あとから定義できます。

集約グローバルユニキャストアドレス

集約グローバルユニキャストアドレスは、グローバル通信に使用するアドレスです。CIDR (クラスレス相互ドメインルーティング) における IPv4 アドレスに機能的に似ています。表 1-2 に、そのフォーマットをまとめます。

表 1-2 集約グローバルユニキャストアドレスのフォーマット

3 ビット	13 ビット	8 ビット	24 ビット	16 ビット	64 ビット
FP	TLA ID	RES	NLA ID	SLA ID	Interface ID

FP	フォーマットプレフィックス (001)
TLA ID	最上位集約識別子
RES	将来用に予約
NLA ID	次レベル集約識別子
SLA ID	サイトレベル集約識別子
INTERFACE ID	インタフェース識別子

最初の 48 ビットはパブリックトポロジを表します。次の 16 ビットは各サイトのトポロジを表します。

最初の 3 ビットは集約グローバルユニキャストアドレスとしてアドレスを識別します。次のフィールドである TLA ID はルーティング階層の最上位レベルです。その次の 8 ビットは将来用に予約されています。NLA ID フィールドは TLA ID を割り当てられた組織が、アドレス指定階層の作成と、サイトの識別に使用します。

SLA ID フィールドは、組織で各ローカルアドレス指定階層の作成とサブネットを識別するときに使用します。SLA ID フィールドの使い方は IPv4 のサブネットと似ていますが、組織別に割り当てることのできるサブネット数をはるかに多いところが異なります。16 ビット SLA ID フィールドがサポートするサブネットの数は 65,535 です。Interface ID は、リンク上のインタフェースを識別するために使用します。Interface ID はそのリンク上で一意である必要があります。また、より広い範囲で一意とすることが可能です。通常、インタフェース識別子はインタフェースのリンクのアドレスと同じです。または、インタフェースのリンク層のアドレスに基づいた値です。

ローカル用アドレス

ローカル用アドレスは、ローカルにルーティング可能な範囲のみを対象とするユニキャストアドレスです。使用できるのは、サブネット内または加入者ネットワーク内に限定されます。ローカル用アドレスは、プラグアンドプレイのローカル通信を行うために、サイト内で使用します。また、グローバルアドレスを使用するためのブートストラップ操作を行うために使用されます。

ローカル用のユニキャストアドレスには、リンクローカルとサイトローカルの 2 種類があります。リンクローカル用アドレスは、Ethernet や フレームリレーリンクのような単一のデータリンク層方式で使用します。サイトローカル用は単一サイトで使用します。次の表は、リンクローカル用アドレスフォーマットを示したものです。

表 1-3 リンクローカル用アドレスフォーマット

10 ビット	54 ビット	64 ビット
1111111010	0	Interface ID

リンクローカル用アドレスは自動アドレス設定などの目的で1つのリンク上のアドレス指定に使用します。

表 1-4 は、サイトローカル用アドレスフォーマットです。

表 1-4 サイトローカル用アドレス

10 ビット	38 ビット	16 ビット	64 ビット
1111111011	0	Subnet ID	Interface ID

どちらのタイプのローカルアドレスでも、インタフェース ID はそれを使用するドメインで一意的な識別子である必要があります。通常は、識別子としてノードの IEEE-802 48 ビットアドレスを使用します。Subnet ID は、サイト内の特定のサブネットを識別します。Subnet ID と Interface ID を組み合わせてローカルアドレスを作成します。これで大規模なプライベートインターネットを構築することができ、その他のアドレス割り当てを行う必要はありません。

現在グローバルインターネットに接続していない組織はローカルアドレスを使用できます。ローカルアドレスを使うだけでグローバルインターネットアドレス領域からのアドレスプレフィックスを要求する必要はありません。この組織が将来インターネットに接続する場合、Subnet ID と Interface ID をグローバルプレフィックスと組み合わせてグローバルアドレスを作成することができます。たとえば、Registry ID、Provider ID、Subscriber ID の組み合わせでグローバルアドレスを作成できます。この拡張機能は IPv4 に対する大幅な改善点です。IPv4 では、プライベート (非グローバル) な IPv4 アドレスを使うサイトは、インターネットに接続する場合に手動で番号を指定し直す必要があります。IPv6 の場合、番号は自動的に指定し直されます。

組み込み IPv4 アドレスを伴った IPv6 アドレス

IPv6 移行機能では、ホストとルーターが IPv4 ルーティングインフラストラクチャのもとで IPv6 パケットを動的にトンネル処理できる方式を採用しています。この方式を利用した IPv6 ノードには、下位 32 ビットに IPv4 アドレスを保存した特別な IPv6 ユニキャストアドレスが割り当てられます。このタイプのアドレスを *IPv4 互換 IPv6 アドレス* といいます。次の表にそのフォーマットを示します。

表 1-5 IPv4 互換 IPv6 アドレスフォーマット

80 ビット	16 ビット	32 ビット
--------	--------	--------

表 1-5 IPv4 互換 IPv6 アドレスフォーマット (続き)

0000.....0000	0000	IPv4 アドレス
---------------	------	-----------

組み込み IPv4 アドレスを保存する第 2 のタイプの IPv6 アドレスも定義されています。このアドレスは IPv6 アドレス領域内の IPv4 アドレスを表すときに使用します。このアドレスは主に、アプリケーション、API、オペレーティングシステムの実装内で使用します。このタイプのアドレスを *IPv4 マップ IPv6 アドレス* といいます。次の表にそのフォーマットを示します。

表 1-6 IPv4 マップ IPv6 アドレスフォーマット

80 ビット	16 ビット	32 ビット
0000.....0000	FFFF	IPv4 アドレス

エニーキャストアドレス

IPv6 エニーキャストアドレスは複数のインタフェースに割り当てるアドレスです。通常は、エニーキャストアドレスは異なるノードに所属しています。エニーキャストアドレスに送られたパケットは、そのアドレスを持つ最も近いインタフェースにルーティングされます。

エニーキャストアドレスはルートシーケンスの一部に使用できます。したがって、ノードはトラフィックを搬送するインターネットサービスプロバイダを選択できます。この機能をソース選択ポリシーと呼ぶこともあります。この機能を実装するには、インターネットサービスプロバイダに所属するルーターセットを識別するようにエニーキャストアドレスを構成します。たとえば、インターネットサービスプロバイダごとに 1 つのエニーキャストを構成します。エニーキャストを、IPv6 ルーティングヘッダーで中間アドレスとして使用できます。これにより、特定のプロバイダによってパケットが配信されます。または、一連のプロバイダによって配信されます。また、エニーキャストアドレスは、特定のサブネットに接続されたルーターセットや、特定のルーティングドメインへのエントリを提供するルーターセットの識別にも使用できます。

定義済みのユニキャストアドレスフォーマットを利用すれば、ユニキャストアドレス領域からエニーキャストを指定できます。そのため、エニーキャストアドレスは、構文的にはユニキャストアドレスと区別つきません。複数のインタフェースにユニキャストアドレスを割り当てる場合は、ユニキャストアドレスをエニーキャストアドレスに変換します。ただし、そのアドレスがエニーキャストアドレスであることがわかるように、アドレスを割り当てるノードを明示的に構成する必要があります。

マルチキャストアドレス

IPv6 マルチキャストアドレスは、インタフェースグループの識別子です。1 つのインタフェースが所属できるマルチキャストグループは複数設定できます。表 1-7 は、マルチキャストアドレスフォーマットを示します。

表 1-7 マルチキャストアドレスフォーマット

8 ビット	4 ビット	4 ビット	112 ビット
11111111	FLGS	SCOP	グループ ID

アドレスの先頭の 11111111 は、アドレスがマルチキャストアドレスであることを表します。FLGS は、4 つのフラグ (0、0、0、T) のセットです。

上位 3 つのフラグは、予約されていて、これらのフラグは 0 に初期設定される必要があります。

- **T=0** – 固定的に割り当てられた (既知の) マルチキャストアドレスを識別する。このマルチキャストアドレスは、グローバルインターネット番号指定機関が割り当てます。
- **T=1** – 非固定的に割り当てられた (一時的な) マルチキャストアドレスを識別します。

SCOP は、4 ビットのマルチキャストスコープの値であり、マルチキャストグループの有効範囲を表します。表 1-8 は、SCOP の値です。

表 1-8 SCOP の値

0	予約	8	組織ローカルスコープ
1	ノードローカルスコープ	9	(割り当てなし)
2	リンクローカルスコープ	A	(割り当てなし)
3	(割り当てなし)	B	(割り当てなし)
4	(割り当てなし)	C	(割り当てなし)
5	サイトローカルスコープ	D	(割り当てなし)
6	(割り当てなし)	E	全世界的な配信範囲
7	(割り当てなし)	F	予約

グループ ID は、指定スコープ内で、固定または一時的のどちらかのマルチキャストグループを識別します。

IPv6 のルーティング

IPv6 のルーティングは、CIDR における IPv4 のルーティングとほぼ同じです。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、性能、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格納さ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経路の処理と逆引きをサポートしています。始点経路の処理と逆引きは、プロバイダが新機能を実装するホストを使用するためのポイントです。新機能には、プロバイダの選択や拡張アドレスが含まれます。

IPv6 の近傍検索

IPv6 では、同じリンクに接続されたノード間の対話に関連した問題をまとめて解決しました。そのため、次のような問題を個々に解決する仕組みを定義しています。

- ルーター発見 – 接続されたリンクにあるルーターをホストが探索する
- プレフィックス探索 – どの宛先がリンクに接続されているかを定義するアドレスプレフィックスのセットをホストが探索する (オンリンクということもある)。リンクにある宛先と、ルーターからだけアクセスできる宛先を、ノードではプレフィックスで区別します。
- パラメータ探索 – ノードは、リンク MTU (最大伝送単位) などのリンクパラメータを調べる。また、出力パケットに設定するホップ限界数などのインターネットパラメータを調べる
- アドレス自動設定 – インタフェースのアドレスをノードが自動的に設定する
- アドレス解決 – 宛先の IP アドレスだけを使用してノードが近傍のリンク層アドレスを判定する (オンリンク宛先)
- 次のホップの決定 – アルゴリズムは宛先へのトラフィックが送られるべき近傍の IP アドレスに IP 宛先アドレスのマッピングを決定する。次のホップはルーターまたは宛先になる

- 不到達検出 – 近傍に到達不可能であることをノードが判定する。ルーターに使用される近傍の場合、代替デフォルトルーターを試行できる。ルーターとホストの場合、アドレス解決を再試行できる
- 重複アドレス検出 – あるノードがアドレスを要求したところ、別のノードがそのアドレスを使用していないかを判別する
- リダイレクト – 特定の宛先へのアクセス手段として、最適な最初のホップノードをルーターからホストに知らせる

近傍検索では、ルーター要請メッセージとルーター通知メッセージのペア、近傍要請メッセージと近傍通知メッセージのペア、リダイレクトメッセージという5種類のICMP (インターネット制御メッセージプロトコル) パケットタイプを定義します。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 – インタフェースが使用可能になると、ホストはルーター要請を送信できる。この要請は、次に予定されている時刻ではなく、ただちにルーター通知メッセージを送信するようにルーターに要求する
- ルーター通知 – ルーターはさまざまなリンクパラメータやインターネットパラメータとともにその存在を通知する。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する。ルーター通知には、オンリンク判別またはアドレス設定、あるいはホップ限界数の選択肢などに使用するプレフィックスが含まれる
- 近傍要請 – 近傍のリンク層アドレスを判定するため、および、近傍がキャッシュリンク層アドレスで到達可能かどうかを確認するためにノードによって送信される。近傍要請は重複アドレス検出にも使用する
- 近傍通知 – 近傍要請メッセージに対する応答として、ノードでは未要請の近傍通知も送信してリンク層アドレスの変更を伝える
- リダイレクト – 宛先までの最適な最初のホップ、または宛先がオンリンクであることをルーターからホストに知らせる

ルーター通知

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、ルーターは定期的にルーター通知パケットをマルチキャストして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通知を受け取り、デフォルトルーターのリストを作成します。利用できるルーターをホストが短時間 (2、3分以内) に知ることができるように、ルーターは頻繁にルーター通知を生成します。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。

ルーター通知プレフィックス

ルーター通知には、オンリンク判別に使用するプレフィックスリストが含まれます。このプレフィックスリストは、自動アドレス設定にも使用されます。プレフィックスに付属するフラグは特定のプレフィックスの使用目的を表します。ホストは、通知されたオンリンクプレフィックスからリストを作成し管理します。リストは、パケット

の宛先がいつオンリンクになっているか、あるいはルーターを離れているかを知るために使用します。通知されたオンリンクプレフィックス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。

ルーター通知 (およびプレフィックス別のフラグ) では、ルーターからホストにアドレスの自動設定の方法を伝えることができます。たとえば、ステートフル (DHCPv6) か自動 (ステートレス) のどちらのアドレス設定を使用するかなどがあります。

ルーター通知メッセージ

ルーター通知メッセージには、ホストが出力パケットで使用する必要があるホップ限界数などのインターネットパラメータも組み込むことができます。また、オプションでリンク MTU などのリンクパラメータも組み込むことができます。この機能により、重要なパラメータの集中管理が可能になります。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。

ノードでは、宛先ノードに対してそのリンク層アドレスを戻すよう要求する近傍要請をマルチキャストしてアドレス解決を行います。近傍要請メッセージは、宛先アドレスの要請先のノードマルチキャストアドレスにマルチキャストされます。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。

近傍要請と不到達

近傍要請メッセージでは、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を確認することもできます。

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、その IP 層で正しく処理されたかどうかを確認する肯定確認が必要です。近傍不到達検出では、2つのソースの確認を使用します。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。先に送信されたデータは正しく配信されたということが通知されます。たとえば、最も新しい TCP 肯定を受信したことが通知されます。肯定応答が得られない場合、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。

上記の一般的な問題を解決する以外に、近傍検索では次のような状況にも対応します。

- リンク層アドレスの変更 – リンク層アドレスの変更を認識したノードは、非要請の近傍通知パケットをマルチキャストできる。ノードはすべてのノードにマルチキャストして、無効になったキャッシュリンク層アドレスを更新できる。非要請通知の送信は、性能強化が目的。近傍不到達検出アルゴリズムにより、すべてのノード

ドが確実に新しいアドレスを探索できるが、遅延が多少伸びる可能性がある

- 入力負荷均衡 – 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均衡ができる。このようなノード間では、同じインタフェースに複数のリンク層アドレスが割り当てられる。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、複数のリンク層アドレスを持つ1つの論理インタフェースとして表現できる

負荷均衡は、ルーターがソースリンク層アドレスをルーター通知パケットから省略することを可能にすることで処理する。この場合、近傍では近傍要請メッセージを使用してルーターのリンク層アドレスを確認する。近傍通知メッセージの戻りには、要請元によって異なるリンク層アドレスが組み込まれる

- エニーキャストアドレス – エニーキャストアドレスは、等価サービスを提供するノードセットの1つを識別する。同じリンクの複数のノードは同じ任意を認識するように設定できる。近傍検索では、ノードが同じ宛先に対する複数の近傍通知を受信するようにノードを設定してエニーキャストを処理する。エニーキャストアドレスの通知にはすべて、取り消しできない通知としてのタグが設定される。取り消しできない通知により、複数存在する可能性がある通知の中でどれを使用するかを判定する特定の規則が呼び出される
- プロキシ通知 – 宛先アドレスのかわりにパケットを受信するルーターは、取り消し無効の近傍通知を発行できる。ルーターは、近傍要請に回答できない宛先アドレスのかわりにパケットを受信する。現在はプロキシの使用方法は指定されていないが、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る。ただし、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機構として使用されることはない

IPv4 との比較

IPv6 近傍検索プロトコルは、IPv4 プロトコル ARP (アドレス解決プロトコル)、ICMP ルーター検索、ICMP リダイレクトを組み合わせたようなものです。IPv4 には近傍不到達検出に一般的に対応できるプロトコルや機構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。

近傍検索プロトコルでは、IPv4 プロトコルセットに対するさまざまな強化措置が施されています。

- ルーター検索はベースプロトコルセットの一部であり、ホストがルーティングプロトコルを *snoop* する必要はない
- ルーター通知ではリンク層アドレスが伝達される。ルーターのリンク層アドレスの解決に、これ以外のパケット交換は不要
- ルーター通知ではリンクのプレフィックスが伝達される。ネットマスクを設定する独立した機構は不要
- ルーター通知では、アドレス自動設定が使用可能になる
- ルーターは、ホストがリンクで使用するMTU を通知できる。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 値を使用する

- アドレス解決マルチキャストは、40 億 (2^{32}) マルチキャストアドレスに展開され、宛先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削減した。さらに、IPv6 以外のマシンの割り込みをなくした
- リダイレクトには、新しい最初のホップのリンク層アドレスを保存する。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る
- 同じリンクに複数のプレフィックスを関連付けられる。デフォルトで、ホストはルーター通知からすべてのオンリンクプレフィックスを受け取る。ただし、ルーター通知にあるプレフィックスをすべて、あるいは一部省略するようにルーターを設定できる。その場合、ホストは宛先がオフリンクであるとみなす。その結果、ホストはルーターにトラフィックを送信する。ルーターは適宜リダイレクトを発行する
- IPv4 と異なり、IPv6 リダイレクトメッセージの受信者は新しい次のホップがオンリンクであるとみなす。IPv4 では、ホストはリダイレクトメッセージを無視し、リンクのネットワークマスクに基づいて、リンクにない次ホップを指定する。IPv6 リダイレクト機構は XRedirect 機能に似ている。リダイレクト機構は、非ブロードキャストおよび共有メディアリンクで有効。これらのリンクでは、ノードはオンリンク宛先のすべてのプレフィックスを確認できない
- 近傍不能性検出により、障害ルーターがある場合のケット伝送能力が改善される。また、この機能により、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ケット伝送、あるいはリンク層アドレスが変更されたノードを経由するケット伝送が改善される。たとえば、移動ノードは、頻繁に更新される ARP キャッシュのおかげでオフリンクになっても接続が切れない
- 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する。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ぐ
- IPv4 ルーター検索と異なり、ルーター通知メッセージにはユーザー定義フィールドはない。安定性の異なるルーターの操作にユーザー定義フィールドは不要。近傍不能性検出で、デッドルーターを検出し、アクティブルーターに切り替えることができる
- 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る。ルータを識別する機能は、ルーター通知で必要とされる。また、この機能はリダイレクトメッセージも必要とされる。サイトが新しいグローバルプレフィックスを使用しても、ホストはルーター関連付けを維持する必要がある
- 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ない。これに対し、IPv4 オフリンクノードは、ICMP (インターネット制御メッセージプロトコル) リダイレクトメッセージと、ルータ通知メッセージを送ることができる
- ICMP 層にアドレス解決を配置すると、プロトコルが ARP よりも媒体に依存しなくなる。その結果、標準 IP 認証とセキュリティ機構が使用できるようになる

IPv6 ステートレスアドレス自動設定

ホストでは、IPv6 のインタフェースの自動設定を数ステップかけて実行します。自動設定プロセスでは、リンクローカルアドレスの作成、リンク上の一意性の検査、どのような情報を自動設定するか (アドレス、その他の情報、または両方)、アドレスをステートフル機構またはステートレス機構、あるいはその両方で取得するかの決定が行われます。ここでは、リンクローカルアドレスの生成手順、ステートレスアドレス自動設定によるサイトローカルアドレスとグローバルアドレスの生成手順、そして重複アドレス検出手順について説明します。

ステートレス自動設定の条件

IPv6 では、ステートフルとステートレスのアドレス自動設定機構を定義しています。ステートレス自動設定では、手動によるホストの設定は不要です。ルーターは最小限の設定 (あれば) ですみ、サーバーの追加も不要です。ステートレス機構により、ホストは独自のアドレスを生成できます。ステートレス機構は、アドレスを生成するために、ローカルな情報とルーターが通知する情報を使用します。ルーターはリンクに関連付けられたサブネットを識別するプレフィックスを通知します。ホストはサブネット上で一意にインタフェースを識別するインタフェース識別子を生成します。アドレスはこれらのプレフィックスとインタフェース識別子を組み合わせで作ります。ルーターがない場合、ホストはリンクローカルアドレスだけを生成します。ただし、同じリンクに接続されたノード間の通信では、リンクローカルアドレスで十分です。

ステートフル自動設定モデル

ステートフル自動設定モデルでは、ホストはインタフェースアドレスや設定情報とパラメータをサーバーから取り込みます。サーバーでは、どのホストにどのアドレスが割り当てられたかを保存したデータベースを管理します。ホストは、ステートフル自動設定プロトコルを利用してアドレスやその他の設定情報をサーバーから取り込むことができます。ステートレス自動設定とステートフル自動設定は互いに補完し合います。たとえば、ホストでは、ステートレス自動設定でアドレスを設定し、ステートレス自動設定でその他の情報を取り込みます。

ステートレス方式とステートフル方式をいつ使用するか

ホストが使用するアドレスを厳密に知る必要はない場合に、ステートレス方式を使用します。ただし、アドレスは一意である必要があります。また、アドレスは正しくルートできる必要もあります。正確なアドレス割り当てに対してサイトでさらに厳し

く管理する必要がある場合に、ステートフル方式を使用します。ステートフルとステートレスのどちらのアドレス自動設定も同時に使用できます。サイト管理者は、ルーター通知メッセージのフィールドの設定を通じて、どの方式の自動設定を使用するかを指定します。

IPv6 アドレスは、一定の時間 (場合によっては無限に) インタフェースにリースされず。各アドレスには、アドレスがどれだけの時間、インタフェースに割り当てられるかを示す寿命があります。寿命が尽きると、結合とアドレスが無効になり、そのアドレスを別のインタフェースに割り当てることができます。アドレスの割り当ての終了を正常に行うため、アドレスはインタフェースに割り当てられた状態で 2 つの別々のフェーズを経ます。最初、アドレスには優先権が与えられ、任意に通信ができます。次に、アドレスの現在のインタフェース割り当てが無効になるという前提から、優先順位が下がります。優先順位が低い状態で、アドレスを使用するのは避けるべきですが、使用できないわけではありません。新しい通信 (たとえば、新しい TCP 接続の開始など) ではできるだけ優先順位の高いアドレスを使用します。優先順位の低いアドレスを使用できるのは、そのアドレスを使用中のアプリケーションだけにする必要があります。サービスを打ち切らないと別のアドレスに切り替えるのが困難なアプリケーションは、優先順位の低いアドレスを使用できます。

重複アドレスの検出アルゴリズム

特定のリンク上ですべての設定済みアドレスが一意であることを保証するため、ノードは重複アドレスの検出アルゴリズムを実行します。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。重複アドレスの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。

このマニュアルで指定する自動設定プロセスは、ホストにだけ適用し、ルーターには適用しません。ホストの自動設定では、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設定する必要があります。ただし、このマニュアルで説明した機構を使用して、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の検出処理が正常終了していることが望まれます。

自動設定プロセス

ここでは、自動設定中にインタフェースが実行する通常の手順について概要を説明します。自動設定が行われるのはマルチキャスト対応リンクだけです。たとえばシステム起動時など、マルチキャスト対応インタフェースが使用可能な状態で開始します。ノード (ホストとルーターの両方) では、そのインタフェースのリンクローカルアドレスを生成して自動設定プロセスを開始します。リンクローカルアドレスは、インタフェースの識別子を既知のリンクローカルプレフィックスに追加して作成します。

ノードは、この仮リンクローカルアドレスがリンク上の別のノードで使用されていないことを確認する必要があります。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。特に、ノードは宛先が仮アドレスになっている近傍要請メッセージを送信します。別のノードがそのアドレスを使用中の

場合、そのノードはそのことを伝える内容を含む近傍要請を返信します。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、システム管理で設定できます。

ノードにおいて、仮リンクローカルアドレスが一意でないことがわかると自動設定が打ち切られるため、手動でインタフェースを設定する必要があります。この状態からの回復を簡単にするには、管理者が代替インタフェース識別子を提供してデフォルト識別子を無効にします。これにより、新しい(一意であると考えられる)インタフェース識別子を利用して自動設定機構を実行できます。そうでなければ、リンクローカルアドレスとその他のアドレスは手動で設定します。

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。自動設定手順の残りは、ホストだけで実行されます。

ルーター通知の受信

自動設定の次の手順では、ルーター通知を受信するか、ルーターが存在しないことを確認します。ルーターがあれば、ホストが実行すべき自動設定の種類を指定したルーター通知が送信されます。ルーターがない場合、ステートフル自動設定が呼び出されます。

ルーターはルーター通知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、自動設定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。ルーター通知には2つのフラグがあり、どのようなステートフル自動設定(あれば)を実行すべきかを表します。管理アドレス設定フラグは、アドレスの取得時にホストがステートフル自動設定を使用するかどうかを表します。もう1つのステートフル設定フラグは、その他の情報(アドレスを除く)の取得時にホストがステートフル自動設定を使用するかどうかを表します。

プレフィックス情報

ルーター通知には0個以上のプレフィックス情報オプションも入っており、これらのオプションにはステートレスアドレス自動設定においてサイトローカルアドレスとグローバルアドレスの生成に使用する情報が含まれています。ルーター通知のステートレスアドレス自動設定フィールドとステートフルアドレス自動設定フィールドは個別に処理されます。ホストでは、ステートフルアドレス自動設定とステートレスアドレス自動設定を同時に使用できます。プレフィックス情報オプションフィールドの1つである自動アドレス設定フラグは、オプションがステートレス自動設定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プレフィックスと寿命値が保存されます。これらの値は、プレフィックス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。

ルーターではルーター通知が定期的に生成されるので、ホストでは常に新しい通知を受信します。ホストは各通知に組み込まれた情報を上記の手順で処理し、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更新します。

アドレスの一意性

安全性確保のため、すべてのアドレスについて、インタフェースに対する割り当て前に一意かどうかを確認されます。ただし、ステートレス自動設定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、インタフェース識別子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、他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識別子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。ステートフルアドレスの自動設定で得られるアドレスについても同じです。一部のサイトでは、重複アドレスの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場合があります。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の検出の使用を無効にできます。

自動設定処理を短時間で終了するために、ルーター通知の待機と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、自動設定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。

IPv6 モビリティ (移動性) サポート

ルーティングは、パケットの宛先 IP アドレスのサブネットプレフィックスに基づいて行われます。そのため、モバイルノードを宛先とするパケットは、ホームリンクに関連付けられていないノードには到達できません。ホームリンクは、ノードの IPv6 サブネットプレフィックスが存在するリンクです。通信を継続するために、ノードが新しいリンクに移動するたびに、モバイルノードがその IP アドレスを変更できます。ただし、モバイルノードの位置を変更すると、移動ノードではトランスポート層とその上位層の接続が失われます。以上のことから、将来、インターネットに接続するモバイルコンピュータが増加することを考えると、IPv6 モビリティサポートが大きな意味を持つこととなります。

上記の問題に IPv6 モビリティサポートが対応します。IPv6 モビリティでは、モバイルノードがリンク間を移動してもその IP アドレスは変更されません。モバイルノードに対する IP アドレスの割り当ては、そのノードのホームリンク上のホームサブネットプレフィックスの範囲内で行われます。これをノードのホームアドレスといいます。

これにより、モバイルノードのホームアドレスにルートされたパケットは、宛先アクセスできます。モバイルノードが、現在インターネットのどこに接続していても問題はありませぬ。モバイルノードが新しいリンクに移動しても他のノード(固定またはモバイル)との通信は途切れませぬ。

ホームを離れたモバイルノードと送受信するパケットを透過的にルーティングする問題は IPv6 移動サポートで解決できます。しかし、モバイルコンピュータや無線ネットワークの使用に伴うすべての問題が解決されるわけではありません。特に次の問題には対処できません。

- 通常の無線ネットワークのようにアクセスできるときとできないときがあるリンクの処理。ただし、移動検出手順でいくつかの問題は処理できる
- モバイルノードが接続しているリンクのアクセス制御

IPv6 サービス品質 (QoS) 機能

ホストは、IPv6 ヘッダーのフローラベルフィールドとトラフィッククラスフィールドを使用できます。ホストは、これらのフィールドを使用して、IPv6 ルーターによる特別処理を要求するパケットを識別します。特別処理の例としては、デフォルト以外のサービス品質やリアルタイムサービスがあります。この機能により、ある程度一貫したスループット、遅延、ジッターが必要なアプリケーションをサポートできます。この種のアプリケーションには、マルチメディアアプリケーションまたはリアルタイムアプリケーションがあります。

フローラベル

発信元では、IPv6 ヘッダーの 20 ビットのフローラベルフィールドを使用できます。送信元は、IPv6 ルーターによる特別処理を要求するパケットに、このフィールドを使用してラベルを付けます。特別処理の例としては、デフォルト以外のサービス品質やリアルタイムサービスがあります。この IPv6 の機能はまだ実験段階であり、インターネットのフローサポートの条件が確定すると変更される可能性があります。一部のホストまたはルーターではフローラベルフィールドの機能をサポートしていません。このようなホストまたはルーターでは、パケットの生成時にフローラベルフィールドをゼロに設定する必要があります。パケットが転送される場合は、フローラベルフィールドは変更されないまま転送されます。パケットを受信したホストやルーターはフローラベルフィールドを無視します。

フローとは

フローは、特定の送信元から特定の (ユニキャストまたはマルチキャスト) 宛先に送信されるパケットのシーケンスです。ソースは、ルーターによる特別処理を必要とします。特別処理の特性は、制御プロトコルによってルーターに伝達される場合があります。制御プロトコルとして、リソース予約プロトコルを使用できます。また、ホップバイホップオプションなど、フローのパケット内の情報によって伝達される場合もあります。

ソースから宛先までのアクティブフローが複数ある場合があります。また、どのフローとも関連していないトラフィックを含む場合もあります。フローの一意の識別はソースアドレスとゼロ以外のフローラベルの組み合わせによって行います。フローに所属しないパケットは、フローラベルゼロを運びます。

フローのソースノードでは、フローにフローラベルを割り当てます。新しいフローラベルは、擬似的な方法で、ランダムに選択します。16進数で、1からFFFFFFの範囲から均等に選択します。ランダムに割り当てることにより、ルーターはフローラベルフィールド内の任意のビットセットをハッシュキーとして利用できます。ルーターは、ハッシュキーを使ってフローに関連付けられた状態を調べることができます。

同じフローに所属するパケット

同じフローに所属するパケットは、同じソースアドレス、同じ宛先アドレス、同じゼロ以外のフローラベルで送信します。これらのパケットのどれかにホップバイホップオプションヘッダーが含まれる場合、パケットを同じホップバイホップオプションヘッダーの内容で生成する必要があります。ただし、ホップバイホップオプションヘッダーの次のヘッダーフィールドは除かれます。これらのパケットのどれかにルーティングヘッダーが含まれる場合、パケットの拡張ヘッダーを同じ内容で生成する必要があります。この同じ内容には、ルーティングヘッダーより前のすべての拡張ヘッダーと、ルーティングヘッダーが含まれます。ただし、ルーティングヘッダーの次のヘッダーフィールドは除かれます。ルーターや宛先では、場合によってはこれらの条件が満たされているかを確認できます。違反を検出した場合、そのことを送信元に報告する必要があります。違反を報告するには、ICMPパラメータ問題メッセージ、コード0を使用します。違反は、フローラベルフィールドの上位オクテットで表されます。この上位オクテットは、IPv6パケット内のオフセット1オクテットです。

ルーターは、任意のフローのフロー処理状態を自由にセットアップできます。この場合ルーターは、制御プロトコル、ホップバイホップオプション、その他の手段による、明示的なフロー確立情報を必要としません。たとえば、未知のゼロ以外に設定されたフローラベルを持つパケットを特定のソースから受信した場合、ルーターではそのIPv6ヘッダーを処理できます。ルーターは、フローラベルがゼロに設定されている拡張ヘッダーを処理する場合と同じ方法で、必要な拡張ヘッダーを処理できます。ルーターは、次中継点のインタフェースの判別を行います。場合によってはホップバイホップオプションの更新、ルーティングヘッダーのポインタとアドレスの加算、あるいはパケットのキューイングの方法の決定なども行います。パケットのキューイングの方法の決定は、パケットのトラフィッククラスフィールドに基づいて行われます。ルーターは、この処理手順の結果を記憶することを選択できます。そして、記憶した後でその情報をキャッシュに保存できます。始点アドレスとフローラベルがキャッシュキーとして使用されます。同じ始点アドレスとフローラベルを持つ後続のパケットについては、キャッシュされた情報を参照することにより処理できます。これらのパケットの始点アドレスとフローラベルをすべて調べる必要はありません。ルーターは、フローの最初のパケットは確認しますが、その後はフィールドの内容は変更されないと仮定することができます。

トラフィッククラス

パケットを生成したノードは、IPv6 パケットの異なるクラスまたは優先順位を識別する必要があります。その場合、IPv6 ヘッダーのトラフィッククラスフィールドが使用されます。パケットを転送するルーターも同じ目的でトラフィッククラスフィールドを使用します。

トラフィッククラスフィールドには、以下の一般的な要件が適用されます。

- 1つのノード内の IPv6 サービスへのサービスインタフェースは、上位層プロトコルに対して、トラフィッククラスビットの値を提供する必要があります。上位層プロトコルで生成されたパケットにはトラフィッククラスビットが必要です。デフォルト値は、8 ビットすべてが 0 です。
- 一部またはすべてのトラフィッククラスビットをサポートするノードは、ビットの値を変更することができます。変更できるのは、サポートする特定の使用方法に従ってそのノードが生成、転送、または受信するパケット内のビットの値です。ノードは、特定の使用方法をサポートしないすべてのトラフィッククラスフィールド内のビットを無視し、変更しないようにしなければなりません。
- 受信パケット内のトラフィッククラスビットは、そのパケットの発信元が送信した値とは異なる値である可能性があります。したがって、上位層プロトコルは、トラフィッククラスビットの値が同じであると仮定することはできません。

IPv6 セキュリティの強化

現在のインターネットには多くのセキュリティ問題があります。インターネットでは、アプリケーション層より下の層には有効な機密機構や認証機構がありません。この欠点に対し、IPv6 では、セキュリティサービスを提供する 2 つの統合オプションを設けて対応しています。この 2 つのオプションは、別々に、あるいはまとめて使用してさまざまなユーザーにさまざまなセキュリティレベルを提供できます。ユーザー通信が異なれば、セキュリティのニーズも異なります。

最初のオプションは IPv6 認証ヘッダー (AH) と呼ばれる拡張ヘッダーです。この拡張ヘッダーは、IPv6 データグラムに機密性を持たない認証と完全性を提供します。この拡張機能はアルゴリズムに依存せず、さまざまな認証方式をサポートします。認証ヘッダーは、ワールドワイドなインターネット内の相互運用性の保証を支援するために、それを使用することが提案されています。認証ヘッダーを使用することにより、ホストなりすまし攻撃など、主なネットワーク侵害を回避できます。IPv6 でソースルーティングを使用する場合、IP ソースルーティングに明らかな危険性があるので IPv6 認証ヘッダーが重要になります。上位層プロトコルおよび上位層サービスには、現在有効な保護策はありません。しかし、インターネット層に認証ヘッダーを使用することで、ホスト発信元認証を提供できます。

2 番目のオプションである、IPv6 カプセル化セキュリティペイロード (ESP) と呼ばれる拡張ヘッダーは、IPv6 データグラムに完全性と機密性を提供します。いくつかの同じようなセキュリティプロトコルよりも単純ですが、ESP はフレキシブルなままで、独立したアルゴリズムです。同様のセキュリティプロトコルには、SP3D と ISO NLSP が含まれます。

IPv6 認証ヘッダーと IPv6 カプセル化セキュリティヘッダーは、新しいインターネットプロトコルセキュリティ (IPsec) の機能です。IPsec の概要については、『Solaris のシステム管理 (IP サービス)』の「IPsec (概要)」を参照してください。IPsec の実装の方法については、『Solaris のシステム管理 (IP サービス)』の「IPsec 実装の作業マップ」を参照してください。

IPv4 ネットワーク経由の 6to4 トンネル

IPv4 から IPv6 へのスムーズな移行のため、Solaris オペレーティングシステムでは 6to4 移行機構をサポートしています。この機構は、IPv4 ネットワークを介し、独立した IPv6 サイトから独立した別の IPv6 サイトへパケットのトンネリングを可能にするものです。6to4 機構を使用することで管理者は、1 つのグローバル IPv4 アドレスを使用し、完全な 48 ビット IPv6 グローバルプレフィックスを生成できます。6to4 ルーティングの技術情報が必要な場合は、RFC 3056 の『Connection of IPv6 Domains via IPv4 Clouds』を参照してください。

ご使用の IPv6 サイトが以下のどちらか一方あるいは両方の条件にあう場合は、6to4 の実装を検討してください。

- インターネットに対する IPv6 接続が確立されていない
- 独立した IPv6 サイトが、独立した別の IPv6 サイトと通信を行う必要がある

かつては、独立した IPv6 サイトは、ほかの IPv6 サイトと通信を行えませんでした。6to4 ルーティングを使用すると、独立したサイトから IPv4 ネットワーク経由でトンネルを介してパケットを転送できます。この場合に必要となる条件は、IPv4 ネットワークに接続するインタフェースに一意の (世界に 1 つしかない) IPv4 アドレスを持たせて境界ルーターを使用することだけです。

インタフェースがこの要件を満たす場合には、6to4 サポート用のルーターに複数のインタフェースを設定できます。6to4 サポートのためにホストを手動で設定する必要はありません。6to4 ルーターからプレフィックス通知を受け取ると、IPv6 ホストは 6to4 アドレスを使用して自動的に IPv6 インタフェースを再設定します。

ルーターは、出力用の IPv6 パケットを IPv4 ヘッダーを使用してカプセル化します。続いて、IPv4 ネットワーク経由で 6to4 ルーターと宛先の 6to4 サイト間で自動トンネルが構築されます。リモート側の 6to4 ルーターは、パケットを受けとった時点でそのカプセル化を解除し、標準の IPv6 形式となったこのパケットを適切な IPv6 ノードへ配布します。

6to4 ルーターは、6to4 サイトではないネイティブ IPv6 サイトにパケットをトンネリングすることもできます。この場合、6to4 ルーターは IPv4 ネットワークを介して 6to4 リレールーターとの間のトンネルを構築します。リレールーターは、パケットを IPv6 ネットワークに転送します。

6to4 ルーティングの詳細

6to4 ルーティングの設定はいたって容易です。次に、6to4 設定の作業と技術情報の参照先を示します。

作業または技術情報	参照先
6to4 ルーターの設定	38 ページの「6to4 ルーターの設定方法」
6to4 リレールーターの設定	57 ページの「6to4 リレールーターとの間の 6to4 トンネルの設定方法」
ifconfig に対する 6to4 変更	66 ページの「ifconfig ユーティリティに対する IPv6 拡張機能」
6to4relay コマンドの詳細	68 ページの「6to4relay コマンド」
6to4 サイトトポロジの詳細	93 ページの「6to4 トンネルの構成」
6to4 アドレス指定の詳細	95 ページの「6to4 派生のアドレス指定」
6to4 リレールーターの詳細	98 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」

第 2 章

IPv6 の管理 (手順)

この章では、ホストやルーターで IPv6 を設定する方法について説明します。主な内容を以下に示します。

- 35 ページの「IPv6 ノードを有効にする」
- 36 ページの「IPv6 ノードを有効にする (作業マップ)」
- 43 ページの「IPv6 の監視」
- 43 ページの「IPv6 の監視 (作業マップ)」
- 52 ページの「IP 内 IP トンネルの設定」
- 52 ページの「IP 内 IP トンネルを設定する (作業マップ)」
- 58 ページの「IPv6 ネームサービス情報の表示」
- 59 ページの「IPv6 ネームサービス情報を表示する (作業マップ)」

内容	参照先
IPv6 の概要	第 1 章
この章で説明する IPv6 作業に関連した詳しい概念情報	第 3 章
IPv4 から IPv6 への移行方法の詳細	第 4 章

IPv6 ノードを有効にする

この節では、IPv6 ノードをネットワークで設定するときに必要な手順について説明します。

注 - この節で「ノード」という用語は、Solaris サーバーまたはクライアントホストを指します。

IPv6 ノードを有効にする (作業マップ)

表 2-1 IPv6 ノードを有効にする (作業マップ)

作業	説明	参照先
ノード上の IPv6 を有効にする	hostname6.interface ファイルの操作、アドレスの表示、 /etc/inet/ipnodes ファイルへのアドレスの入力。この表の「注」参照。	36 ページの「ノード上の IPv6 を有効にする方法」
Solaris IPv6 ルーターの設定	indp.conf ファイルへのエントリの追加	37 ページの「Solaris IPv6 ルーターの設定方法」
6to4 ルーターの設定	/etc/hostname6.ip.6to4tun0 ファイルを使用して 6to4 インタフェースを設定する	38 ページの「6to4 ルーターの設定方法」
IPv6 アドレスを NIS と NIS+ に追加	/etc/ipnodes ファイルへのエントリ追加	41 ページの「NIS と NIS+ に対する IPv6 アドレスの追加方法」
IPv6 アドレスを DNS に追加	DNS ゾーンと逆ゾーンファイルに対する AAAA レコードの追加	42 ページの「DNS に対する IPv6 アドレスの追加方法」

注 - IPv6 は、Solaris ソフトウェアをインストールするときにシステムで有効にできません。インストールプロセスで yes と応答して IPv6 を有効にすると、あとの IPv6 を有効にする手順を省略できます。

▼ ノード上の IPv6 を有効にする方法

1. IPv6 を有効にしたいシステム上でスーパーユーザーになります。
2. 各インタフェースに対して次のように入力します。

```
# touch /etc/hostname6.interface
```

```
interface
```

```
1e0, 1e1 などのインタフェース名
```

3. ノードをリブートする

注 - リブートすると、ルーターに検索パケットが送信されます。ルーターがプレフィックスを応答することにより、ノードが IPv6 アドレスでインタフェースを設定できるようになります。リブートすると、主なネットワークデーモンも IPv6 モードで再起動します。

4. 全インタフェースの IPv6 アドレスを表示します。

```
# ifconfig -a6
```

IPv4 アドレスと IPv6 アドレスを表示するには、`-a` オプションだけを使用します。

5. 適切なネームサービスに、IPv6 アドレスを次のように追加します。
 - NIS と NIS+ については、41 ページの「NIS と NIS+ に対する IPv6 アドレスの追加方法」を参照してください。
 - DNS については、42 ページの「DNS に対する IPv6 アドレスの追加方法」を参照してください。

▼ Solaris IPv6 ルーターの設定方法

1. ルーターとして機能するシステム上で、スーパーユーザーになります。
2. `/etc/inet/ndpd.conf` ファイルを編集して、サブネットプレフィックスを使用して次のエントリを 1 つまたは複数追加します。

変数と使用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を参照してください。`ndpd.conf` ファイルについては、`ndpd.conf(4)` のマニュアルページを参照してください。

 - a. すべてのインタフェースについて、ルーター動作を指定するエントリを追加します。

```
ifdefault variable value
```
 - b. プレフィックス通知のデフォルト動作を指定するエントリを追加します。

```
prefixdefault variable value
```
 - c. インタフェースパラメータごとのセットエントリを追加します。

```
if interface variable value
```
 - d. インタフェースプレフィックス情報ごとの通知エントリを追加します。

```
prefix prefix/length interface variable value
```
3. システムをリブートします。

注 - ホストのサブネットアドレスプレフィックスが、近傍検索 (in.ndpd) からホストにリレーされます。また、次世代 RIP ルーティングプロトコル (in.ripngd) が自動的に実行されます。

例 - ndpd.conf ルーター設定ファイル

```
# Send router advertisements out all NICs
ifdefault AdvSendAdvertisements on
# Advertise a global prefix and a
# site local prefix on three interfaces.
# 0x9255 = 146.85
prefix 2:0:0:9255::0/64      hme0
prefix fec0:0:0:9255::0/64  hme0
# 0x9256 = 146.86
prefix 2:0:0:9256::0/64      hme1
prefix fec0:0:0:9256::0/64  hme1
# 0x9259 = 146.89
prefix 2:0:0:9259::0/64      hme2
prefix fec0:0:0:9259::0/64  hme2
```

▼ 6to4 ルーターの設定方法

IPv6 ネットワーク上で 6to4 ルーティングを設定する前に、以下の作業をすませておく必要があります。

- 36 ページの「ノード上の IPv6 を有効にする方法」で説明しているように、予定されている 6to4 サイトですべての必要なノードで IPv6 を設定する
- 6to4 ルーターとして使用するため、IPv4 ネットワークに接続された 1 台以上のルーターを選択する
- IPv4 ネットワークに対して使用する 6to4 ルーターのインタフェースに一意の (全世界に 1 つしかない) IPv4 アドレスを設定する。この IPv4 アドレスは静的なものでなければならない。動的に割り当てられる IPv4 アドレスは使用できない。IPv4 インタフェースの設定方法については、『Solaris のシステム管理 (IP サービス)』の「ネットワーク構成手順」を参照してください。

1. 予定されている 6to4 ルーター上で、スーパーユーザーになります。
2. `/etc/hostname6.ip.6to4tun0` ファイルを作成することにより、そのルーター上で 6to4 擬似インタフェースを設定します。
 - サブネット ID が 0 でホスト ID が 1 の推奨されている規則を使用する場合は、`/etc/hostname6.ip.6to4tun0` では次に示す短い形式を使用してください。

```
tsrc IPv4 address up
```
 - サブネット ID とホスト ID にほかの規則を使用する場合は、`/etc/hostname6.ip.6to4tun0` では次に示す長い形式を使用してください。

```
tsrc IPv4 address 2002:IPv4 address:subnet ID:host ID:/64 up
```

次の表は、この作業で使用される `/etc/hostname6.ip.6to4tun0` のパラメータを説明したものです。

<code>tsrc</code>	このインタフェースがトンネルソースとして使用されることを示す
<code>IPv4 address</code>	6to4 擬似インタフェースとなる物理インタフェース上で設定される IPv4 アドレスをドット付きの 10 進数形式で指定する
<code>2002</code>	(省略可能) 6to4 プレフィックスを指定する
<code>IPv4 address</code>	(省略可能) 擬似インタフェースの IPv4 アドレスを 16 進で指定する
<code>subnet ID</code>	(省略可能) 0 以外のサブネット ID を 16 進表記で指定する
<code>host ID</code>	(省略可能) 1 以外のホスト ID を指定する
<code>/64</code>	アドレスの長さが 64 ビットであることを示す
<code>up</code>	インタフェースを “up” として設定するように <code>ifconfig</code> に指示する

注 - 2 つの IPv6 トンネルに同じソースアドレスと宛先アドレスを指定することはできません。同じアドレスを指定するとパケットは削除されます。このような状況は、6to4 ルーターが `atun` を通してトンネリングも実施する場合に発生する可能性があります。

3. (省略可能) ルーター上でさらに **6to4** 擬似インタフェースを作成します。
予定されている各 6to4 擬似インタフェースには、すでに設定された一意の (全世界に 1 つしかない) IPv4 アドレスが必要です。

4. インタフェースの状態を確認するために `ifconfig` を実行します。

```
# ifconfig ip.6to4tun0 inet6
```

インタフェースが正しく設定されている場合は、次のようなメッセージが表示されます。

```
ip.6to4tun0: flags=2200041<UP,RUNNING,NUD,IPv6>mtu 1480 index 11
    inet tunnel src 129.146.87.188
    tunnel hop limit 60
    inet6 2002:8192:56bb:56bb::1/64
```

5. **6to4** ルーティングを通知するために `/etc/inet/ndpd.conf` を編集します。

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

- a. 最初の行で、通知を受け取るサブネットを指定します。

`if` エントリを次の書式で作成してください。

```
if subnet-interface AdvSendAdvertisements 1
```

たとえば、インタフェース `hme0` に接続しているサブネットに対して `6to4` ルーティングを通知するには、`subnet-interface` を `hme0` に置き換えます。

```
if hme0 AdvSendAdvertisements 1
```

- b. 通知の 2 行目として **6to4** プレフィックスを追加します。

`prefix` エントリを次の書式で作成してください。

```
prefix 2002:IPv4-address:subnet ID::/64 subnet-interface
```

<code>prefix 2002</code>	6to4 プレフィックスを示す
<code>IPv4-address</code>	6to4 擬似インタフェースの一意の IPv4 アドレスを 16 進表記で指定する
<code>subnet ID</code>	サブネット ID を 16 進で指定する
<code>subnet-interface</code>	サブネットが接続しているルーターインタフェース

6. ルーターをリポートします。

リポートするかわりに、`/etc/inet/in.ndpd` に対して `sighup` を発行してルーター通知の送信を開始することもできます。これにより、`6to4` プレフィックスを受け取る各サブネット上の IPv6 ノードは、`6to4` から派生した新しいアドレスを使用して自動設定を行います。

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、42 ページの「DNS に対する IPv6 アドレスの追加方法」または 41 ページの「NIS と NIS+ に対する IPv6 アドレスの追加方法」を参照してください。

6to4 サイトにおける複数のルーターの設定

サイトに複数のルーターが存在する場合は、`6to4` をサポートするために `6to4` ルーターの背後のルーターをさらに設定しなければならないことがあります。サイトで RIP を使用している場合は、`6to4` 以外の各ルーター上で `6to4` ルーターとの間の静的な経路を設定する必要があります。市販のルーティングプロトコルを使用する場合は、`6to4` ルーターとの間の静的な経路を構築する必要はありません。

例—6to4 ルーターの設定

次に、`/etc/hostname6.ip.6to4tun0` の短い形式の例を示します。

```
# cat /etc/hostname6.ip.6to4tun0
tsrc 129.146.87.188 up
```

次に、`/etc/hostname6.ip.6to4tun0` の長い形式の例を示します。

```
# cat /etc/hostname6.ip.6to4tun0
tsrc 129.146.87.188 2002:8192:56bb::1/64 up
```

次に、6to4 擬似インタフェースの `ifconfig` の出力例を示します。

```
# ifconfig ip.6to4tun0 inet6
ip.6to4tun0: flags=2200041<UP,RUNNING,NOUD,IPv6> mtu 1480 index 11
    inet tunnel src 129.146.87.188
    tunnel hop limit 60
    inet6 2002:8192:56bb::1/64
```

次の `/etc/inet/ndpd.conf` ファイル例は、2つのサブネット上の 6to4 ルーティングを通知します。

```
if qfe0 AdvSendAdvertisements 1
prefix 2002:8192:56bb:1::/64 qfe0

if qfe1 AdvSendAdvertisements 1
prefix 2002:8192:56bb:2::/64 qfe1
```

▼ NIS と NIS+ に対する IPv6 アドレスの追加方法

NIS+ 用に `ipnodes.org_dir` という新しいテーブルが追加されました。このテーブルには、ホスト用の IPv4 アドレスと IPv6 アドレスの両方が保存されています。既存の IPv4 情報だけを保持している `hosts.org_dir` テーブルは、既存のアプリケーションを円滑に実行するために残しておきます。`hosts.org_dir` テーブルと `ipnodes.org_dir` テーブルはどちらも IPv4 アドレスと整合させておく必要があります。概要については、79 ページの「Solaris ネームサービスに対する IPv6 拡張機能」を参照してください。

新しい `ipnodes.org_dir` テーブルの管理方法は、`hosts.org_dir` の管理方法と似ています。従来の NIS+ テーブルの管理に使用したのと同じツールとユーティリティは、`ipnodes.org_dir` にも有効です。NIS+ テーブルの操作についての詳細は、『Solaris のシステム管理 (ネーミングとディレクトリサービス: DNS、NIS、LDAP 編)』を参照してください。

次のコマンドは、`/etc/inet/ipnodes` のエントリを `ipnodes.org_dir` テーブルに冗長モードでマージします。NIS+ テーブルは、`nistbladm(1)`、`nissetup(1M)`、または `nissserver(1M)` のどれかで作成されたものとします。

- コマンド行で、次のコマンドを入力します。

```
% nisaddent -mv -f /etc/inet/ipnodes ipnodes
```

ipnodes.org_dir テーブルを表示するには、次のように操作します。

- コマンド行で、次のコマンドを入力します。

```
% nisaddent -d ipnodes
```

NIS 用に、新しいマップ ipnodes.byname と ipnodes.byaddr の 2 つが追加されました。これらのマップは、いずれも IPv4 と IPv6 のホスト名とアドレスの関連付けを保存しています。hosts.byname マップと hosts.byaddr マップは、IPv4 のホスト名とアドレスの関連情報だけを保存していますが、既存のアプリケーションが動作できるように変更されていません。しいマップの管理は、以前の hosts.byname マップと hosts.byaddr マップの管理方法と同様です。hosts マップを IPv4 アドレスで更新すると、新しい ipnode マップも同じ情報で更新されることに注意してください。

注 - IPv6 対応ツールは、新しい NIS マップと新しい NIS+ テーブルを使用します。

▼ DNS に対する IPv6 アドレスの追加方法

1. **IPv6** ネットワークのプライマリまたはセカンダリ **DNS** サーバー上でスーパーユーザーになります。
2. **DNS** ゾーンファイルに、各 **IPv6** 有効化ホストの **AAAA** レコードを次のフォーマットで追加して編集します。

```
host-name IN AAAA host-address
```

3. **DNS** 逆ゾーンファイルを編集し、次のフォーマットで **PTR** レコードを追加します。

```
host-address IN PTR host-name
```

DNS 管理の詳細については、『Solaris のシステム管理 (ネーミングとディレクトリサービス: DNS、NIS、LDAP 編)』の「DNS の管理 (参照情報)」を参照してください。

例 - DNS ゾーンファイル

```
vallejo IN AAAA 2::9256:a00:20ff:fe12
IN AAAA fec0::9256:a00:20ff:fe12:528
```

例 - DNS 逆ゾーンファイル

```
$ORIGIN ip6.int.
8.2.5.0.2.1.e.f.f.f.9.2.0.0.a.0.6.5.2.9.0.0.0.0.0.0.2.0.0.0 \
IN PTR vallejo.Eng.apex.COM.
```

IPv6 の監視

次のコマンドは IPv6 の Solaris 実装に対応するように変更されています。

- ifconfig (1M)
- netstat (1M)
- snoop (1M)
- ping (1M)
- traceroute (1M)

追加コマンドを使用すると診断を実行できます。これらのコマンドの考え方については、66 ページの「ifconfig ユーティリティに対する IPv6 拡張機能」と 75 ページの「既存のユーティリティに対する IPv6 拡張機能」を参照してください。

IPv6 の監視 (作業マップ)

表 2-2 IPv6 の監視 (作業マップ)

作業	説明	参照先
インタフェースアドレス割り当ての表示	ifconfig コマンドで、すべてのアドレス割り当て、あるいは IPv4、IPv6、または 6to4 派生アドレス割り当てだけを表示	44 ページの「インタフェースアドレス割り当ての表示方法」
ネットワーク状態の表示	すべてのソケットとルーティングテーブルエントリ、IPv4 用の inet アドレスファミリー、IPv6 用の inet6 アドレスファミリー、netstat コマンドによるインタフェースの IPv6 または ICMPv6 カウンタの統計を表示	45 ページの「ネットワーク状態の表示方法」
IPv6 関連コマンドの出力表示の制御	ping コマンド、netstat コマンド、ifconfig コマンド、traceroute コマンドの出力の制御。inet_type という名前のファイルの作成と、そのファイル内の DEFAULT_IP 変数の設定	49 ページの「IPv6 関連コマンドの出力表示の制御方法」

表 2-2 IPv6 の監視 (作業マップ) (続き)

作業	説明	参照先
IPv6 ネットワークトラフィックだけの監視	snoop コマンドによるすべての IPv6 パケットの表示	50 ページの「IPv6 ネットワークトラフィックの監視方法」
すべてのマルチホームホストアドレスの探査	ping コマンドによるすべてのアドレスの確認	50 ページの「すべてのマルチホームホストアドレスの探査方法」
すべてのルートのトレース	traceroute コマンドの使用	51 ページの「すべてのルーターのトレース方法」

▼ インタフェースアドレス割り当ての表示方法

IPv4 や IPv6 のアドレス割り当ての場合だけでなく、すべてのアドレス割り当てを表示する場合も `ifconfig` コマンドを使用します。

- コマンド行で次のコマンドを入力します。

```
% ifconfig [option]
```

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

例 – すべてのインタフェースについてアドレス指定情報を表示

```
% ifconfig -a
lo0: flags=1000849 mtu 8232 index 1
    inet 120.10.0.1 netmask ff000000
le0: flags=1000843 mtu 1500 index 2
    inet 120.46.86.54 netmask ffffffff00 broadcast 120.146.86.255
    ether 8:0:73:56:a8
lo0: flags=2000849 mtu 8252 index 1
    inet6 ::1/128
le0: flags=2000841 mtu 1500 index 2
    ether 8:0:20:56:a8
    inet6 fe80::a00:fe73:56a8/10
le0:1: flags=2080841 mtu 1500 index 2
    inet6 fec0::56:20ff:fe73:56a8/64
le0:2: flags=2080841 mtu 1500 index 2
    inet6 2::56:a00:fe73:56a8/64
```

例 – すべての IPv4 インタフェースについてアドレス指定情報を表示

```
% ifconfig -a4
lo0: flags=1000849 mtu 8232 index 1
    inet 120.10.0.1 netmask ff000000
```

```
le0: flags=1000843 mtu 1500 index 2
      inet 120.46.86.54 netmask ffffffff0 broadcast 120.46.86.255
      ether 8:0:20:56:a8
```

例 – すべての IPv6 インタフェースについてアドレス指定情報を表示

```
% ifconfig -a6
lo0: flags=2000849 mtu 8252 index 1
      inet6 ::1/128
le0: flags=2000841 mtu 1500 index 2
      ether 8:0:20:56:a8
      inet6 fe80::a00:fe73:56a8/10
le0:1: flags=2080841 mtu 1500 index 2
      inet6 fec0::56:20ff:fe73:56a8/64
le0:2: flags=2080841 mtu 1500 index 2
      inet6 2::56:a00:fe73:56a8/64
ip.6to4tun0: flags=2200041<UP,RUNNING,NUD,IPv6> mtu 1480 index 11
      inet tunnel src 129.146.86.187
      tunnel hop limit 60
      inet6 2002:8192:56bb::1/64
```

例 – 6to4 擬似インタフェースについてアドレス指定情報を表示

```
% ifconfig ip.6to4tun0 inet6
6to4tun0: flags=2200041<UP,RUNNING,NUD,IPv6> mtu 1480 index 11
      inet tunnel src 172.16.213.81
      tunnel hop limit 60
      inet6 2002:ac10:d551::1/64
```

▼ ネットワーク状態の表示方法

次の手順では、`netstat` コマンドで、次に示すネットワークデータ構造フォーマットを表示できます。

- すべてのソケットとルーティングテーブルのエントリ
 - IPv4 用の `inet` アドレスファミリ
 - IPv6 用の `inet6` アドレスファミリ
 - インタフェース別統計 – IPv6/ICMPv6 カウンタ
- コマンド行で次のコマンドを入力します。

```
% netstat [option]
```

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

例 - 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
UDP: IPv4
  Local Address          Remote Address      State
-----
  *.*                   Unbound
  *.apexrpc              Idle
  *.*                   Unbound
  .
  .

UDP: IPv6
  Local Address          Remote Address      State
If -----
  *.*                   Unbound
  *.time                 Idle
  *.echo                 Idle
  *.discard              Idle
  *.daytime              Idle
  *.chargen              Idle

TCP: IPv4
  Local Address          Remote Address      Swind Send-Q Rwind Recv-Q  State
-----
  *.*                   *.*                0     0     0     0  IDLE
  *.apexrpc             *.*                0     0     0     0  LISTEN
  *.*                   *.*                0     0     0     0  IDLE
  *.ftp                 *.*                0     0     0     0  LISTEN
localhost.427          *.*                0     0     0     0  LISTEN
  *.telnet               *.*                0     0     0     0  LISTEN
tn.apex.COM.telnet is.Eng.apex.COM  8760    0   8760    0  ESTABLISHED
tn.apex.COM.33528 np.apex.COM.46637  8760    0   8760    0  TIME_WAIT
tn.apex.COM.33529 np.apex.COM.apexrpc 8760    0   8760    0  TIME_WAIT

TCP: IPv6
  Local Address          Remote Address      Swind Send-Q Rwind Recv-Q  State  If
-----
  *.*                   *.*                0     0     0     0  IDLE
  *.ftp                 *.*                0     0     0     0  LISTEN
  *.telnet               *.*                0     0     0     0  LISTEN
  *.shell                *.*                0     0     0     0  LISTEN
  *.smtp                 *.*                0     0     0     0  LISTEN
  .
  .
2::56:8.login          something.1023      8640    0  8640    0  ESTABLISHED
fe80::a:a8.echo        fe80::a:89         8640    0  8640    0  ESTABLISHED
fe80::a:a8.ftp         fe80::a:90         8640    0  8640    0  ESTABLISHED
```

例 - IPv4 用の inet アドレスファミリーを表示

```
% netstat -f inet
TCP: IPv4
```

Local Address	Remote Address	Swind	Send-Q	Rwind	Recv-Q	State
tn.apex.COM.telnet	is.apex.COM.35388	8760	0	8760	0	ESTABLISHED
tn.apex.COM.1022	alive-v4.nfsd	8760	0	8760	0	ESTABLISHED
tn.apex.COM.1021	sl.apex.COM.nfsd	8760	0	8760	0	ESTABLISHED
.						
tn.apex.COM.33539	np.apex.COM.apexrpc	8760	0	8760	0	TIME_WAIT

例 - IPv6 用の inet6 アドレスファミリーを表示

```
% netstat -f inet6
```

TCP: IPv6

Local Address	Remote Address	Swind	Send-Q	Rwind	Recv-Q	State	If
2::56:a8.login	something.1023	8640	0	8640	0	ESTABLISHED	
fe80::a0:a8.echo	fe80::a0:de.35389	8640	0	8640	0	ESTABLISHED	
.							
fe80::a0:a8.ftp-data	fe80::a0:de.35394	25920	0	25920	0	TIME_WAIT	

例 - インタフェース別統計を表示 - IPv6 / ICMPv6 カウンタ

```
% netstat -sa
```

RAWIP

rawipInDatagrams	=	1407	rawipInErrors	=	0
rawipInCksumErrs	=	0	rawipOutDatagrams	=	5
rawipOutErrors	=	0			

UDP

udpInDatagrams	=	7900	udpInErrors	=	0
udpOutDatagrams	=	7725	udpOutErrors	=	0

TCP

tcpRtoAlgorithm	=	4	tcpRtoMin	=	200
tcpRtoMax	=	60000	tcpMaxConn	=	-1

.

IPv4

ipForwarding	=	2	ipDefaultTTL	=	255
ipInReceives	=	406345	ipInHdrErrors	=	0
ipInAddrErrors	=	0	ipInCksumErrs	=	0

.

IPv6 for lo0

ipv6Forwarding	=	2	ipv6DefaultHopLimit	=	0
ipv6InReceives	=	0	ipv6InHdrErrors	=	0

.

IPv6 for le0

ipv6Forwarding	=	2	ipv6DefaultHopLimit	=	255
----------------	---	---	---------------------	---	-----

```

        ipv6InReceives      = 885      ipv6InHdrErrors      = 0
        .
IPv6    ipv6Forwarding      = 2      ipv6DefaultHopLimit  = 255
        ipv6InReceives      = 885      ipv6InHdrErrors      = 0
        .
ICMPv4  icmpInMsgs          = 618      icmpInErrors          = 0
        icmpInCksumErrs     = 0      icmpInUnknowns        = 0
        icmpInDestUnreachs  = 5      icmpInTimeExcds       = 0
        .
ICMPv6  for lo0
        icmp6InMsgs         = 0      icmp6InErrors          = 0
        icmp6InDestUnreachs = 0      icmp6InAdminProhibs   = 0
        .
ICMPv6  for le0
        icmp6InMsgs         = 796      icmp6InErrors          = 0
        icmp6InDestUnreachs = 0      icmp6InAdminProhibs   = 0
        icmp6InTimeExcds   = 0      icmp6InParmProblems   = 0
        .
ICMPv6  icmp6InMsgs         = 796      icmp6InErrors          = 0
        icmp6InDestUnreachs = 0      icmp6InAdminProhibs   = 0
        .
ICMPv6  for ip.6to4tun0
        icmp6InMsgs         = 57      icmp6InErrors          = 0
        icmp6InDestUnreachs = 57      icmp6InAdminProhibs   = 0
        icmp6InTimeExcds   = 0      icmp6InParmProblems   = 0
        icmp6InPktTooBigs  = 0      icmp6InEchos           = 0
        icmp6InEchoReplies = 0      icmp6InRouterSols     = 0
        icmp6InRouterAds   = 0      icmp6InNeighborSols   = 0
        icmp6InNeighborAds = 0      icmp6InRedirects      = 0
        icmp6InBadRedirects = 0      icmp6InGroupQueries   = 0
        icmp6InGroupResps  = 0      icmp6InGroupReds      = 0
        icmp6InOverflows   = 0
        icmp6OutMsgs        = 57      icmp6OutErrors         = 0
        icmp6OutDestUnreachs = 57      icmp6OutAdminProhibs  = 0
        icmp6OutTimeExcds  = 0      icmp6OutParmProblems  = 0
        icmp6OutPktTooBigs = 0      icmp6OutEchos         = 0
        icmp6OutEchoReplies = 0      icmp6OutRouterSols    = 0
        icmp6OutRouterAds   = 0      icmp6OutNeighborSols  = 0
        icmp6OutNeighborAds = 0      icmp6OutRedirects     = 0
        icmp6OutGroupQueries = 0      icmp6OutGroupResps    = 0
        icmp6OutGroupReds  = 0

IGMP:
2542 messages received
    0 messages received with too few bytes
    0 messages received with bad checksum
2542 membership queries received
        .

```


▼ IPv6 関連コマンドの出力表示の制御方法

netstat コマンドと ifconfig コマンドの出力は制御できます。まず、`/etc/default` ディレクトリで `inet_type` という名のファイルを作成します。次に、`DEFAULT_IP` 変数の値を指定します。`inet_type` 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。

1. `/etc/default/inet_type` ファイルを作成します。
2. 必要に応じて、次のどれかのエントリを作成します。
 - 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
```

注 - `ifconfig` の `-4` フラグと `-6` フラグの設定は、`inet_type` ファイルに設定された値より優先します。また、`netstat` の `-f` フラグの設定も、`inet_type` ファイルに設定された値より優先します。

例 - IPv4 情報と IPv6 情報を選択する出力の制御

- `DEFAULT_IP=BOTH` または `DEFAULT_IP=IP_VERSION6` 変数を `inet_type` ファイルで設定する場合、次の結果が得られます。

```
% ifconfig -a
lo0: flags=1000849 mtu 8232 index 1
    inet 120.10.0.1 netmask ff000000
le0: flags=1000843 mtu 1500 index 2
    inet 120.46.86.54 netmask ffffffff00 broadcast 120.46.86.255
    ether 8:0:20:56:a8
lo0: flags=2000849 mtu 8252 index 1
    inet6 ::1/128
le0: flags=2000841 mtu 1500 index 2
    ether 8:0:20:56:a8
    inet6 fe80::a00:fe73:56a8/10
le0:1: flags=2080841 mtu 1500 index 2
    inet6 fec0::56:a00:fe73:56a8/64
le0:2: flags=2080841 mtu 1500 index 2
    inet6 2::56:a00:fe73:56a8/64
```

- `DEFAULT_IP=IP_VERSION4` 変数を `inet_type` ファイルで設定する場合、次の結果が得られます。

```
% ifconfig -a
lo0: flags=849 mtu 8232
    inet 120.10.0.1 netmask ff000000
le0: flags=843 mtu 1500
    inet 120.46.86.54 netmask ffffffff broadcast 120.46.86.255
    ether 8:0:20:56:a8
```

▼ IPv6 ネットワークトラフィックの監視方法

すべてのIPv6 パケットを表示するためには、次のように snoop コマンドを実行します。

1. スーパーユーザーになります。
2. コマンド行で次のコマンドを入力します。

```
# 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 – IPv6 ネットワークトラフィックだけの表示

```
# snoop ip6
Using device /dev/le (promiscuous mode)
fe80::a0:a1 -> ff02::9 IPv6 S=fe80::a0:a1 D=ff02::9 LEN=892
fe80::a0:de -> fe80::a0:a8 IPv6 S=fe80::a0:de D=fe80::a0:a8 LEN=104
fe80::a0:a8 -> fe80::a0:de IPv6 S=fe80::a0:a8 D=fe80::a0:de LEN=104
fe80::a0:a1 -> ff02::9 IPv6 S=fe80::a0:a1 D=ff02::9 LEN=892
fe80::a0:de -> fe80::a0:a8 IPv6 S=fe80::a0:de D=fe80::a0:a8 LEN=104
ping -afe80::a0:a8 -> fe80::a0:de IPv6 S=fe80::a0:a8 D=fe80::a0:de LEN=152
fe80::a0:a1 -> ff02::9 IPv6 S=fe80::a0:a1 D=ff02::9 LEN=892
fe80::a0:de -> fe80::a0:a8 IPv6 S=fe80::a0:de D=fe80::a0:a8 LEN=72
fe80::a0:a8 -> fe80::a0:de IPv6 S=fe80::a0:a8 D=fe80::a0:de LEN=72
fe80::a0:a8 -> fe80::a0:de IPv6 S=fe80::a0:a8 D=fe80::a0:de LEN=72
fe80::a0:de -> fe80::a0:a8 IPv6 S=fe80::a0:de D=fe80::a0:a8 LEN=72
```

▼ すべてのマルチホームホストアドレスの探査方法

この操作では、ping コマンドですべてのアドレスを調べます。

- コマンド行で次のコマンドを入力します。

```
% ping -a ipng11
ipng11 (2::102:a00:fe79:19b0) is alive
ipng11 (fec0::102:a00:fe79:19b0) is alive
ipng11 (190.68.10.75) is alive
```

ping コマンドについての詳細は、ping(1M) のマニュアルページを参照してください。

▼ すべてのルーターのトレース方法

この操作では、traceroute コマンドですべてのルーターを調べます。

- コマンド行で次のコマンドを入力します。

```
% traceroute -a <hostname>
```

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

例 – すべてのルーターのトレース

```
% traceroute -a ipng11
traceroute: Warning: Multiple interfaces found; using 2::56:a0:a8 @ le0:2
traceroute to ipng11 (2::102:a00:fe79:19b0), 30 hops max, 60 byte packets
 1 ipng-rout86 (2::56:a00:fe1f:59a1) 35.534 ms 56.998 ms *
 2 2::255:0:c0a8:717 32.659 ms 39.444 ms *
 3 ipng61.Eng.apex.COM (2::103:a00:fe9a:ce7b) 401.518 ms 7.143 ms *
 4 ipng12-00 (2::100:a00:fe7c:cf35) 113.034 ms 7.949 ms *
 5 ipng11 (2::102:a00:fe79:19b0) 66.111 ms * 36.965 ms

traceroute: Warning: Multiple interfaces found; using fec0::56:a8 @ le0:1
traceroute to ipng11 (fec0::10:b0), 30 hops max, 60 byte packets
 1 ipng-rout86 (fec0::56:a00:fe1f:59a1) 96.342 ms 78.282 ms 88.327 ms
 2 ipng8-tun1 (fec0::25:0:0:c0a8:717) 268.614 ms 508.416 ms 438.774 ms
 3 ipng61.Eng.apex.COM (fec0::103:a00:fe9a:ce7b) 6.356 ms * 713.166 ms
 4 ipng12-00 (fec0::100:a00:fe7c:cf35) 7.409 ms * 122.094 ms
 5 ipng11 (fec0::102:a00:fe79:19b0) 10.620 ms * *

traceroute to ipng11.eng.apex.com (190.68.10.75), 30 hops max, 40 byte packets
 1 rmpj17c-086.Eng.apex.COM (120.46.86.1) 4.360 ms 3.452 ms 3.479 ms
 2 flrmpj17u.Eng.apex.COM (120.46.17.131) 4.062 ms 3.848 ms 3.505 ms
 3 ipng8.Eng.apex.COM (120.68.7.23) 4.773 ms * 4.294 ms
 4 ipng61.Eng.apex.COM (120.68.10.104) 5.128 ms 5.362 ms *
 5 ipng12-20.Eng.apex.COM (120.68.10.62) 7.298 ms 5.444 ms *
 6 ipng11.Eng.apex.COM (120.68.10.75) 8.053 ms 6.394 ms *
```

IP 内 IP トンネルの設定

ここでは、IP 内 IP トンネルを手動で設定する方法について説明します。自動 6to4 トンネルを設定する方法については、38 ページの「6to4 ルーターの設定方法」を参照してください。

IPv6 は、以下に示すトンネルのカプセル化をサポートします。

- IPv4 トンネル経由の IPv4
- IPv4 トンネル経由の IPv6 (ネイティブ IPv6 と 6to4)
- IPv6 トンネル経由の IPv6
- IPv6 トンネル経由の IPv4
- 6to4 トンネル経由で IPv6 から IPv6 へ
- 6to4 サイトトンネルから 6to4 リレールーターを経由してネイティブ IPv6 サイトへ

トンネルの概念については、77 ページの「IPv6 の Solaris トンネルインタフェース」と 89 ページの「トンネル機構」を参照してください。

IP 内 IP トンネルを設定する (作業マップ)

表 2-3 IP 内 IP トンネルを設定する (作業マップ)

作業	説明	参照先
IPv4 トンネル経由の IPv6 の手動設定	hostname6.ip.tun <i>n</i> ファイルに必要なエントリの説明	53 ページの「IPv4 トンネルによる IPv6 の設定方法」
IPv4 トンネル (6to4 トンネル) 経由の IPv6 の自動設定	hostname6.ip.6to4tun ファイルに必要なエントリの説明	38 ページの「6to4 ルーターの設定方法」
IPv6 経由の IPv6 トンネルの設定	hostname6.ip.6.tun <i>n</i> ファイルに必要なエントリの説明	54 ページの「IPv6 トンネル経由の IPv6 の設定方法」
IPv6 経由の IPv4 トンネルの設定	hostname.ip6.tun <i>n</i> ファイルに必要なエントリの説明	55 ページの「IPv6 トンネル経由の IPv4 の設定方法」
IPv4 経由の IPv4 トンネルの設定	hostname.ip.tun <i>n</i> ファイルに必要なエントリの説明	55 ページの「IPv4 トンネル経由の IPv4 の設定方法」

表 2-3 IP 内 IP トンネルを設定する (作業マップ) (続き)

作業	説明	参照先
6to4 ルーターと 6to4 リレー ルーター間のトンネルの設定	6to4relay コマンドを使用して 6to4 リレー ルーターとの間のトンネルを有効にする方法の説明	57 ページの「6to4 リレー ルーターとの間の 6to4 トンネルの設定方法」
トンネルインタフェースに通知するルーターの設定	/etc/inet/ndpd.conf ファイルに必要なエントリの説明	56 ページの「トンネルインタフェースで通知するためのルーターの設定方法」

▼ IPv4 トンネルによる IPv6 の設定方法

注 - IPv4 ネットワーク経由で IPv6 パケットを引き渡す方法としてもっとも望ましいのは、自動 6to4 トンネルを使用することです。サイトで 6to4 ルーティングを実装する手順は、38 ページの「6to4 ルーターの設定方法」を参照してください。

1. スーパーユーザーになります。
2. `/etc/hostname6.ip.tun n` ファイルを作成します。n には 0、1、2 などの値を使用します。次に、以下の手順に従って、エントリを追加します。

- a. トンネルソースアドレスとトンネル宛先アドレスを追加します。

```
tsrc IPv4-source-addr tdst IPv4-destination-addr up
```

- b. (省略可能) ソース IPv6 アドレスと宛先 IPv6 アドレスの論理インタフェースを追加します。

```
addif IPv6-source-address IPv6-destination-address up
```

このインタフェースに対してアドレスを自動設定したい場合は、この手順を省きます。各トンネルに対するリンクローカルアドレスを設定する必要はありません。リンクローカルアドレスは自動的に設定されます。

トンネルを設定したあと、リポートしてください。

注 - 双方向通信を実現するには、トンネルのもう一方の端についても同じ手順を行う必要があります。

使用するシステムをルーターとして設定する場合、リポートする前にトンネルインタフェースに通知するようにルーターを設定する必要があります。56 ページの「トンネルインタフェースで通知するためのルーターの設定方法」を参照してください。

例 — IPv6 アドレスを自動設定するための IPv6 設定ファイルのエントリ

次に、すべての IPv6 アドレスが自動設定されるトンネルの例を示します。

```
tsrc 129.146.86.138 tdst 192.168.7.19 up
```

例 — 手動で設定されたアドレスの IPv6 設定ファイルのエントリ

次に、グローバルソースアドレスとグローバル宛先アドレスが手動で設定されるトンネルの例を示します。サイトローカルソースアドレスとサイトローカル宛先アドレスも手動で設定されます。

```
tsrc 120.46.86.138 tdst 190.68.7.19 up
addif fec0::1234:a00:fe12:528 fec0::5678:a00:20ff:fe12:1234 up
addif 2::1234:a00:fe12:528 2::5678:a00:20ff:fe12:1234 up
```

▼ IPv6 トンネル経由の IPv6 の設定方法

1. スーパーユーザーになります。
2. `/etc/hostname6.ip6.tun n` ファイルを作成します。*n* には **0**、**1**、**2** などの値を使用します。次に、以下の手順に従って、エントリを追加します。
 - a. トンネルソースアドレスとトンネル宛先アドレスを追加します。

```
tsrc IPv6-source-address tdst IPv6-destination-address
IPv6-packet-source-address IPv6-packet-destination-address up
```

- b. (省略可能) ソース **IPv6** アドレスと宛先 **IPv6** アドレスの論理インタフェースを追加します。

```
addif IPv6-source-address IPv6-destination-address up
```

このインタフェースに対してアドレスを自動設定したい場合は、この手順を省きません。各トンネルに対するリンクローカルアドレスを設定する必要はありません。リンクローカルアドレスは自動的に設定されます。

トンネルを設定したあと、リポートしてください。

注 – 双方向通信を実現するには、トンネルのもう一方の端についても同じ手順を行う必要があります。

使用するシステムをルーターとして設定する場合、リポートする前にトンネルインタフェースに通知するようにルーターを設定する必要があります。56 ページの「トンネルインタフェースで通知するためのルーターの設定方法」を参照してください。

例 — IPv6 トンネル経由で IPv6 を作成するための IPv6 設定ファイルのエントリ

次に、IPv6 トンネルによる IPv6 のエントリの例を示します。

```
tsrc 2000::114:a00:20ff:fe72:668c tdst 2000::103:a00:20ff:fe9b:a1c3
fe80::4 fe80::61 up
```

▼ IPv6 トンネル経由の IPv4 の設定方法

1. スーパーユーザーになります。
2. `/etc/hostname.ip6.tun n` ファイルを作成します。n には 0、1、2 などの値を使用します。次に、以下の手順に従って、エントリを追加します。
 - a. トンネルソースアドレスとトンネル宛先アドレスを追加します。

```
tsrc IPv6-source-address tdst IPv6-destination-address
tunnel-IPv4-source-address tunnel-IPv4-destination-address up
```

- b. (省略可能) ソース IPv6 アドレスと宛先 IPv6 アドレスの論理インタフェースを追加します。

```
addif IPv6-source-address IPv6-destination-address up
```

トンネルを設定したあと、リポートしてください。

注 – 双方向通信を実現するには、トンネルのもう一方の端についても同じ手順を行う必要があります。

使用するシステムをルーターとして設定する場合、リポートする前にトンネルインタフェースに通知するようにルーターを設定する必要があります。56 ページの「トンネルインタフェースで通知するためのルーターの設定方法」を参照してください。

例 — IPv6 トンネル経由で IPv4 を作成するための IPv4 設定ファイルのエントリ

次に、IPv6 トンネル経由の IPv4 のエントリの例を示します。

```
tsrc 2000::114:a00:20ff:fe72:668c tdst 2000::103:a00:20ff:fe9b:a1c3
10.0.0.4 10.0.0.61 up
```

▼ IPv4 トンネル経由の IPv4 の設定方法

1. スーパーユーザーになります。

2. `/etc/hostname.ip.tun n` ファイルを作成します。*n* には 0、1、2 などの値を使用します。次に、以下の手順に従って、エントリを追加します。

a. トンネルソースアドレスとトンネル宛先アドレスを追加します。

```
tsrc IPv4-source-address tdst IPv4-destination-address  
tunnel-IPv4-source-address tunnel-IPv4-destination-address up
```

b. (省略可能) ソース IPv4 アドレスと宛先 IPv4 アドレスの論理インタフェースを追加します。

```
addif IPv4-source-address IPv4-destination-address up
```

トンネルを設定したあと、リブートしてください。

注 - 双方向通信を実現するには、トンネルのもう一方の端についても同じ手順を行う必要があります。

使用するシステムをルーターとして設定する場合、リブートする前にトンネルインタフェースに通知するようにルーターを設定する必要もあります。56 ページの「トンネルインタフェースで通知するためのルーターの設定方法」を参照してください。

例 — IPv4 トンネル経由で IPv4 を作成するための IPv4 設定ファイルのエントリ

次に、IPv4 トンネル経由の IPv4 のエントリの例を示します。

```
tsrc 120.46.86.158 tdst 120.46.86.122  
10.0.0.4 10.0.0.61 up
```

▼ トンネルインタフェースで通知するためのルーターの設定方法

トンネルごとに次の操作をします。

1. スーパーユーザーになります。
2. `/etc/inet/ndpd.conf` ファイルを編集します。次の手順に従って、エントリを追加します。

a. トンネルインタフェース経由のルーター通知を有効にします。

```
if ip.tunn AdvSendAdvertisements 1
```

b. 必要に応じてプレフィックスを追加します。


```
prefix interface-address ip.tunn
```

3. リポートします。

▼ 6to4 リレールーターとの間の 6to4 トンネルの設定方法



注意 – 主要なセキュリティ問題を考慮して、Solaris オペレーティングシステムのデフォルト設定では 6to4 リレールーターサポートは無効になっています。98 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」とインターネット文書『*Security Considerations for 6to4*』を参照してください。

6to4 リレールーターとの間のトンネルを有効にする前に、以下の作業をすませておく必要があります。

- 38 ページの「6to4 ルーターの設定方法」の説明に従い、サイトで 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問題を検討する

1. 6to4 ルーター上でスーパーユーザーになります。
2. 以下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 リレールーターには、IPv4 アドレス 192.88.99.1 が割り当てられています。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は、RFC 3068、『*An Anycast Prefix for 6to4 Relay Routers*』を参照してください。

- 特定の 6to4 リレールーターとの間のトンネルを有効にします。

```
# /usr/sbin/6to4relay -e -a relay-router-address
```

-a オプションは、特定のルーターアドレスが続くことを示します。relay-router-address には、トンネルを有効にするために使用する特定の 6to4 リレールーターの IPv4 アドレスを指定してください。

6to4 リレールーターとの間のトンネルは、6to4 トンネル擬似インタフェースが削除されるまでアクティブな状態を維持します。

3. このトンネルは、不要になった時点で次のように入力して削除してください。

```
# /usr/sbin/6to4relay -d
```

4. (省略可能) 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ります。このような状況に対応するためには、以下の作業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. (省略可能) 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—6to4 リレールーターサポートの状態情報の表示

`/usr/bin/6to4relay` 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 Solaris オペレーティングシステムのデフォルト) 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 destination address of Relay Router=192.88.99.1
```

IPv6 ネームサービス情報の表示

ここでは、IPv6 ネームサービス情報を表示する手順について説明します。

IPv6 ネームサービス情報を表示する (作業マップ)

表 2-4 IPv6 ネームサービス情報を表示する (作業マップ)

作業	説明	参照先
IPv6 ネームサービス情報の表示	nslookup コマンドで、IPv6 ネームサービス情報を表示する	59 ページの「IPv6 ネームサービス情報の表示方法」
DNS IPv6 PTR レコードの正確な更新の確認	nslookup コマンドで DNS IPv6 PTR レコードを表示する。また、set q=PTR パラメータを使用する	60 ページの「DNS IPv6 PTR レコードの正確な更新の確認方法」
NIS+ による IPv6 情報の表示	ypmatch コマンドで、IPv6 情報を NIS から表示する	60 ページの「NIS による IPv6 情報の表示方法」
NIS+ による IPv6 情報の表示	nismatch コマンドを実行して NIS+ で IPv6 情報を表示する	61 ページの「NIS+ による IPv6 情報の表示方法」
ネームサービスに依存しない IPv6 情報の表示	getent コマンドで IPv6 情報を表示する	61 ページの「ネームサービスに依存しない IPv6 情報の表示方法」

▼ IPv6 ネームサービス情報の表示方法

nslookup コマンドで IPv6 ネームサービス情報を表示するには、次のように操作します。

1. コマンド行で次のコマンドを入力します。

```
% /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  
>host-name
```

3. AAAA レコードだけを表示するには、山括弧プロンプトに次のコマンドを入力します。

```
>set q=AAAA
```

4. exit を入力して、コマンドを終了します。

例 – nslookup による IPv6 情報の表示

```
% /usr/sbin/nslookup  
Default Server: space1999.Eng.apex.COM  
Address: 120.46.168.78
```

```

> set q=any
> vallejo
Server: space1999.Eng.apex.COM
Address: 120.46.168.78

vallejo.ipv6.eng.apex.com IPv6 address = fec0::9256:a00:fe12:528
vallejo.ipv6.eng.apex.com IPv6 address = 2::9256:a00:fe12:528
> exit

```

▼ DNS IPv6 PTR レコードの正確な更新の確認方法

nslookup コマンドを使用して DNS IPv6 PTR レコードを表示します。

1. コマンド行で次のコマンドを入力します。

```
% /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 – nslookup による PTR レコードの表示

```

% /usr/sbin/nslookup
Default Server: space1999.Eng.apex.COM
Address: 120.46.168.78
> set q=PTR
> 8.2.5.0.2.1.e.f.f.f.0.2.0.0.a.0.6.5.2.9.0.0.0.0.0.0.0.2.0.0.0.ip6.int

8.2.5.0.2.1.e.f.f.f.0.2.0.0.a.0.6.5.2.9.0.0.0.0.0.0.0.2.0.0.0.ip6.int name =
vallejo.ipv6.Eng.apex.COM
ip6.int nameserver = space1999.Eng.apex.COM
> exit

```

▼ NIS による IPv6 情報の表示方法

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。

- コマンド行で次のコマンドを入力します。

```
% ypmatch host-name ipnodes.byname
```

host-name に関する情報が表示されます。

例 2-1 例 — ypmatch を使用して NIS で IPv6 情報を表示する

```
% ypmatch vallejo ipnodes.byname
fec0::9256:a00:20ff:fe12:528    vallejo
2::9256:a00:20ff:fe12:528      vallejo
```

▼ NIS+ による IPv6 情報の表示方法

nismatch 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。

- コマンド行で次のコマンドを入力します。

```
% nismatch host-name ipnodes.org-dir
host-name に関する情報が表示されます。
```

例 2-2 例 — nismatch を使用して NIS+ で IPv6 情報を表示する

```
% nismatch vallejo ipnodes.org_dir
vallejo vallejo fec0::9256:a00:20ff:fe12:528
vallejo vallejo 2::9256:a00:20ff:fe12:528
```

▼ ネームサービスに依存しない IPv6 情報の表示方法

- コマンド行で次のコマンドを入力します。

```
% getent ipnodes host-name
host-name に関する情報が表示されます。
```

例 2-3 例 — getent を使用したネームサービスに依存しない IPv6 情報の表示

```
% getent ipnodes vallejo
2::56:a00:fe87:9aba    vallejo vallejo
fec0::56:a00:fe87:9aba  vallejo vallejo
```


第 3 章

IPv6 のファイルおよびコマンド (リファレンス)

Solaris の IPv6 の実装は、主にカーネルレベルとユーザーレベルの両方の TCP/IP スタックへの変更から構成されます。新しい IPv6 モジュールにより、トンネル、ルーター検索、ステートレスアドレス自動設定を使用できます。この章では、IPv6 の Solaris 実装に伴う概念について説明します。

この章では、以下の内容について説明します。

- 63 ページの「Solaris IPv6 実装の概要」
- 64 ページの「IPv6 ネットワークインタフェース設定ファイル」
- 69 ページの「複数のネットワークインタフェースがあるノード」
- 70 ページの「IPv6 デーモン」
- 75 ページの「既存のユーティリティに対する IPv6 拡張機能」
- 76 ページの「表示出力の制御」
- 77 ページの「IPv6 の Solaris トンネルインタフェース」
- 79 ページの「Solaris ネームサービスに対する IPv6 拡張機能」
- 83 ページの「NFS と RPC による IPv6 のサポート」
- 83 ページの「ATM 経由の IPv6 サポート」

Solaris IPv6 実装の概要

IPv4 から IPv6 への移行の一部として、IPv6 では IPv6 パケットを IPv4 パケット内にカプセル化する方式が指定されます。また、IPv6 では、IPv6 パケット内にカプセル化された IPv6 パケットも指定します。

IPv6 には、パケットの実際のカプセル化を行うモジュール `tun` (7M) が含まれます。このモジュールは、トンネルモジュールと呼ばれ、結合されます。このモジュールは、物理インタフェースと同様に `ifconfig` ユーティリティを使用して設定されます。このモジュールによってトンネルモジュールが IP デバイスと IP モジュール間に配置されます。トンネルデバイスにもシステムインタフェースリストにエントリがあります。

自動トンネリングの場合、IPv6 は 6to4tun.c モジュールを含みます。6to4 も、6to4 トンネルの作成時に一般的な tun.c モジュールを使用します。

IPv6 では、ifconfig(1M) ユーティリティも変更されます。IPv6 スタックは、ifconfig を使用して作成します。ifconfig は、この章で説明されている新しいパラメータもサポートするようになりました。

ルーター検索とステートレスアドレス自動設定を行うため、in.ndpd(1M) デーモンが追加されました。

IPv6 ネットワークインタフェース設定 ファイル

IPv4 では起動時に /etc/hostname.interface を使用しましたが、IPv6 でも起動時にファイル /etc/hostname6.interface を使用してネットワークインタフェースを自動的に定義します。このとき、少なくとも /etc/hostname.* ファイルまたは、/etc/hostname6.* ファイルがローカルマシンに存在する必要があります。これらのファイルは、Solaris インストールプログラムで生成されます。IPv6 インタフェースは、Solaris インストールのあとで手動で追加することもできます。

ファイル名については、「interface」を主ネットワークインタフェースのデバイス名で置き換えます。

ファイル名の構文は、次のとおりです。

```
hostname.interface  
hostname6.interface
```

interface の構文は、次のとおりです。

```
dev[.Module[.Module ...]]PPA
```

Dev ネットワークインタフェースデバイス。デバイスは 1e、qe など物理ネットワークインタフェースか、トンネルなどの論理インタフェース。詳細については、77 ページの「IPv6 の Solaris トンネルインタフェース」を参照してください。

Module 結合される際にデバイスにプッシュされるストリームモジュールのリスト

PPA 物理的な接続ポイント

構文 [.[.]] も可能です。

有効なファイル名は、次のとおりです。


```
hostname6.le0
hostname.ip.tun0
hostname.ip6.tun0
hostname6.ip6to4tun0
hostname6.ip.tun0
hostname6.ip6.tun0
```

IPv6 インタフェース設定ファイルのエントリ

IPv6 インタフェースの自動設定では、ノード側でそのリンクローカルアドレスを計算できます。このアドレスは、ノードのリンク層アドレス (つまりその Ethernet (MAC) アドレス) にもとづいたものです。そのため、IPv6 インタフェース構成ファイルにはエントリがないことがあります。その場合、起動スクリプトによってインタフェースが設定されます。ノードは近傍検索デーモン `in.ndpd` で他のアドレスやプレフィックスの情報を取り出します。

インタフェースに静的アドレスが必要な場合、`ifconfig` ユーティリティを使用します。その結果、アドレスまたはホスト名が `/etc/hostname6.interface` (または `/etc/hostname.interface`) に保存されます。インタフェースが設定されるときに、その内容が `ifconfig` に渡されます。

この場合、ファイルに含まれるエントリは1つだけです。このエントリは、ネットワークインタフェースに関連付けられたホスト名または IP アドレスです。たとえば、`ahaggar` というマシンの一次ネットワークインタフェースが `smc0` であるとします。その `/etc/hostname6.*` ファイル名は `/etc/hostname6.smc0` となります。そのエントリは `ahaggar` です。

ネットワークの起動スクリプトでは、ルーティングデーモンとパケット転送を開始するために、インタフェース数と、`/etc/inet/ndpd.conf` ファイルの有無を調べます。37 ページの「Solaris IPv6 ルーターの設定方法」を参照してください。

IPv6 向けに追加または変更されたユーティリティ

IPv6 実装向けに以下のユーティリティの追加または変更が行われました。

- `ifconfig` - このコマンドは、IPv6 をサポートするためにいくつかの変更が加えられました。
- `6to4relay` - このコマンドは、6to4 ルーターとリレールーター間にトンネルを設定します。

ifconfig ユーティリティに対する IPv6 拡張機能

ifconfig ユーティリティにより、IPv6 インタフェースとトンネルモジュールを結合できるようになりました。ifconfig(1M) ユーティリティでは、ioctl の拡張セットで IPv4 ネットワークインタフェースと IPv6 ネットワークインタフェースの両方を設定します。次に、ifconfig に追加されたオプションを示します。このユーティリティによる診断手順については、44 ページの「インタフェースアドレス割り当ての表示方法」を参照してください。

表 3-1 新しい ifconfig ユーティリティオプション

オプション	説明
index	インタフェースインデックスを設定する
tsrc/ tdst	トンネルソース / 宛先を設定する
addif	論理インタフェースの次の候補を作成する
removeif	指定された IP アドレスの論理インタフェースを削除する
destination	インタフェースにポイントツーポイント宛先アドレスを設定する
set	インタフェースにアドレスとネットマスクのどちらか、または両方を設定する
subnet	インタフェースのサブネットアドレスを設定する
xmit/ -xmit	インタフェースにおけるバケット伝送を使用可能または使用不能する

IPv6 設定手順については、35 ページの「IPv6 ノードを有効にする」を参照してください。

例 - 新しい ifconfig コマンドオプション

次に示す ifconfig コマンドは、まず hme0:3 論理インタフェースを 1234::5678/64 IPv6 アドレスに作成します。次に up オプションでインタフェースを使用可能にし、状態を報告し、インタフェースを使用不可にします。最後に、インタフェースを削除します。

例 3-1 例 - addif と removeif の使用

```
# ifconfig hme0 inet6 addif 1234::5678/64 up
Created new logical interface hme0:3

# ifconfig hme0:3 inet6
hme0:3: flags=2000841<UP,RUNNING,MULTICAST,IPv6> mtu 1500 index 2
        inet6 1234::5678/64
```

例 3-1 例 - addif と removeif の使用 (続き)

```
# ifconfig hme0:3 inet6 down
# ifconfig hme0 inet6 removeif 1234::5678
```

次に示す ifconfig コマンドは、まず物理インタフェース名に関連付けられたデバイスを開きます。次に TCP/IP がデバイスを使用するために必要なストリームを構成し、デバイスの状態を報告し、トンネルのソースアドレスと宛先アドレスを構成します。最後に、構成後のデバイスの最新状態を報告します。

例 3-2 例 - tsrc/tdst と index

```
# ifconfig ip.tun0 inet6 plumb index 13

# ifconfig ip.tun0 inet6
ip.tun0: flags=2200850<POINTOPOINT,RUNNING,MULTICAST,NUD,
#IPv6> mtu 1480 index 13
        inet tunnel src 0.0.0.0
        inet6 fe80::/10 --> ::

# ifconfig ip.tun0 inet6 tsrc 120.46.86.158 tdst 120.46.86.122

# ifconfig ip.tun0 inet6
ip.tun0: flags=2200850<POINTOPOINT,RUNNING,MULTICAST,NUD,
IPv6> mtu 1480 index 13
        inet tunnel src 120.46.86.158 tunnel dst 120.46.86.122
        inet6 fe80::8192:569e/10 --> fe80::8192:567a
```

例 3-3 ifconfig による 6to4 の設定—長い形式

```
# ifconfig ip.6to4tun0 inet6 plumb
# ifconfig ip.6to4tun0 inet tsrc 129.146.86.187 \
2002:8192:56bb:1::8192:56bb/64 up

# ifconfig ip.6to4tun0 inet6
ip.6to4tun0: flags=2200041<UP,RUNNING,NUD,IPv6>mtu 1480 index 11
        inet tunnel src 129.146.86.187
        tunnel hop limit 60
        inet6 2002:8192:56bb:1::8192:56bb/64
```

この 6to4 擬似インタフェース設定例は、サブネット ID として 1 を使用し、ホスト ID を 16 進形式で指定しています。

例 3-4 ifconfig による 6to4 の設定—短い形式

```
# ifconfig ip.6to4tun0 inet6 plumb
# ifconfig ip.6to4tun0 inet tsrc 129.146.86.187 up

# ifconfig ip.6to4tun0 inet6
ip.6to4tun0: flags=2200041<UP,RUNNING,NUD,IPv6>mtu 1480 index 11
        inet tunnel src 129.146.86.187
        tunnel hop limit 60
```

例 3-4 ifconfig による 6to4 の設定—短い形式 (続き)

```
inet6 2002:8192:56bb::1/64
```

6to4relay コマンド

6to4 トンネリングを使用すると、独立した 6to4 サイト間の通信が行えます。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 サイトがネイティブ IPv6 サイトとデータを交換しなければならない場合は、6to4relay コマンドを使用して適切なトンネルを設定します。

リレールーターの使用は安全とは言えないため、Solaris オペレーティングシステム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は、98 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の関連作業については、57 ページの「6to4 リレールーターとの間の 6to4 トンネルの設定方法」に示されています。

6to4relay の構文

6to4relay コマンドの書式は次のとおりです。

```
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を確立します。
- 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、Solaris オペレーティングシステムのデフォルトの設定です。
- h 6to4relay の使用方法に関するヘルプ情報を表示します。

詳細は、6to4relay(1M) のマニュアルページを参照してください。

例—6to4 リレールーターサポートの状態の表示

引数を指定せずに `6to4relay` を実行すると、6to4 リレールーターサポートの現在の状態が表示されます。最初の例はデフォルトの状態を示しています。

```
# /usr/sbin/6to4relay
6to4relay:6to4 Relay Router communication support is disabled
```

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
```

`6to4relay` に `-a` オプションと IPv4 アドレスを指定すると、192.88.99.1 ではなく、`-a` で指定した IPv4 アドレスが表示されます。

`-d`、`-e`、および `-a IPv4 address` オプションの実行が正常に完了した場合、`6to4relay` はそのことを報告しません。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。

複数のネットワークインタフェースがあるノード

ノードに複数のネットワークインタフェースがある場合、追加インタフェース用に `/etc/hostname.interface` ファイルを作成する必要があります。

IPv4 の動作

たとえば、マシン `timbuktu` について考えてみます。このシステムは、『Solaris のシステム管理 (IP サービス)』の「相互接続された 3 つのネットワーク」に示されています。このシステムには、2 つのネットワークインタフェースがあり、ルーターとして機能します。プライマリネットワークインタフェース `le0` は、ネットワーク 192.9.200 に接続されています。システムの IP アドレスは 192.9.200.70、ホスト名は `timbuktu` です。Solaris インストールプログラムによって、一次ネットワークインタフェースにファイル `/etc/hostname.le0` が作成され、ホスト名 `timbuktu` がファイルに入力されます。

2 番目のネットワークインタフェースは `le1` で、`192.9.201` に接続されています。このインタフェースは `timbuktu` に物理的にインストールされていますが、独自の IP アドレスが必要です。そのため、このインタフェースに対して `/etc/hostname.le1` ファイルを手動で作成する必要があります。このファイルのエントリはルーター名 `timbuktu-201` です。

IPv6 の動作

IPv6 を設定する場合、`/etc/hostname6.le0` と `/etc/hostname6.le1` のインタフェースだけが必要です。各インタフェースアドレスは、システムの起動時に自動的に設定されます。

IPv6 デーモン

ここでは、次の IPv6 デーモンについて説明します。

- `in.ndpd` - IPv6 自動設定用のデーモン
- `in.ripngd` - IPv6 のネットワークルーティングデーモン
- `inetd` - インターネットサービスデーモン

`in.ndpd` デーモン

このデーモンはルータ発見を実装します。デーモンは、IPv6 の自動アドレス設定も実装します。表 3-4 は、サポートされているオプションを示します。

表 3-2 `in.ndpd` デーモンのオプション

オプション	説明
<code>-d</code>	すべてのイベントのデバッグをオンにする
<code>-D</code>	特定のイベントのデバッグをオンにする
<code>-f</code>	設定を読み出す元のファイル (デフォルトファイルのかわり)
<code>-I</code>	インタフェースごとに関連情報を印刷する
<code>-n</code>	ルーター通知をループバックしない
<code>-r</code>	受信パケットを無視する
<code>-v</code>	冗長モード (さまざまな種類の診断メッセージを報告する)

表 3-2 in.ndpd デーモンのオプション (続き)

オプション	説明
-t	パケット追跡をオンにする

パラメータは、in.ndpd の動作を制御します。これらのパラメータは /etc/inet/ndpd.conf 設定ファイルと /var/inet/ndpd_state.interface 設定ファイル (存在する場合) に設定されます。

/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設定が行われます。表 3-3 に、このファイルに出現する可能性がある各種キーワードをまとめます。ホストを起動しても、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設定できます。このファイルにはインタフェースアドレス、更新時間、有効期間などの情報が保存されています。また、先のルーター通知で得られた情報も保存されています。

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。

表 3-3 /etc/inet/ndpd.conf キーワード

キーワード	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する。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する ifdefault [variable value]
prefixdefault	プレフィックス通知のデフォルトの動作を指定する。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する prefixdefault [variable value]
if	インタフェース別パラメータを設定する。構文は次のとおり if interface [variable value]
prefix	インタフェース別プレフィックス情報を通知する。構文は次のとおり prefix prefix /length interface [variable value]

注 - ifdefault と prefixdefault エントリは、設定ファイルの if エントリと prefix エントリの前に置く必要があります。

設定変数と設定できる値については、in.ndpd(1M) と ndpd.conf(4) のマニュアルページを参照してください。

例 - /etc/inet/ndpd.conf ファイル

次の例は、コメント行のテンプレートと、キーワードと設定変数の使用方法を示します。

```
# 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
#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
```


in.ripngd デーモン

in.ripngd デーモンは、IPv6 ルーターの RIP 次世代ルーティングプロトコルを実装します。このプロトコルは、IPv6 用の RIP に相当する内容を定義します。RIP は、広く使用されている IPv4 ルーティングプロトコルで、Bellman-Ford 距離ベクトルアルゴリズムに基づいています。表 3-4 は、サポートされているオプションを示します。

表 3-4 in.ripngd デーモンのオプション

オプション	説明
-p <i>n</i>	<i>n</i> は RIPNG パケットの送受信に使用する代替ポート番号を指定する
-q	ルーティング情報を打ち切る
-s	デーモンがルーターとして動作している場合でもルーティング情報の提供を強制する
-P	ポイズンリバースを打ち切る
-S	in.ripngd がルーターとして機能しない場合、各ルーターにはデフォルトのルートだけが指定される

inetd インターネットサービスデーモン

IPv6 有効化サーバーは、IPv4 アドレスか IPv6 アドレスを処理できるサーバーです。IPv6 有効化サーバーは、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。/etc/inet/inetd.conf ファイルには、inetd (1M) がソケット経由でインターネット要求を受信したときに呼び出すサーバーリストが保存されています。ソケットベースのインターネットサーバーエントリはそれぞれ、次の構文を使用する 1 行です。

```
service_name socket_type proto flags user server_pathname args
```

各フィールドに指定できる値については、inetd.conf (4) のマニュアルページを参照してください。Solaris オペレーティング環境の場合、IPv6 有効化としてサービスを /etc/inet/inetd.conf ファイルに指定するには、proto フィールドに tcp6 または udp6 を指定します。サービスが IPv4 専用の場合、proto フィールドは tcp または udp として指定します。サービスの proto 値に tcp6 または udp6 を指定すると、inetd は所定のデーモンに AF_INET6 ソケットを渡します。

inetd.conf ファイルの次のエントリは、IPv4 クライアントアプリケーションと IPv6 クライアントアプリケーションの両方と通信できる udp サーバー (myserver) を表します。

例 3-5 IPv4 クライアントアプリケーションと IPv6 クライアントアプリケーションの両方と通信するサーバー

```
myserver dgram udp6 wait root /usr/sbin/myserver mysERVER
```

IPv6 有効化サーバーは、AF_INET (IPv4 専用) ソケットまたは AF_INET6 (IPv6 と IPv4) ソケットを inetd から継承できます。サービスに対して *proto* 値は、tcp6、udp6、tcp または udp として指定されています。この種のサーバーでは、2 つの inetd.conf エントリを指定できます。1 つは *proto* を tcp として、もう1つは *proto* を tcp6 として指定できます。

注 – AF_INET6 ソケットは、IPv4 プロトコルと IPv6 プロトコルのどちらでも使用できるため、*proto* 値 tcp6 (udp6) を指定すれば充分です。

各種 IPv6 有効化サーバーの記述方法については、『プログラミングインタフェース』を参照してください。

Solaris ソフトウェアとともに提供されるサーバーはすべて、*proto* 値を tcp6 または udp6 と指定する inetd エントリが1つあれば十分です。ただし、リモートシェルサーバー (shell) とリモート実行サーバー (exec) のエントリには、tcp と tcp6 の両方の *proto* 値を指定する必要があります。例 3-6 は、rlogin、telnet、shell、exec 用の inetd エントリです。

例 3-6 Solaris ソフトウェアで提供されるサーバー用の inetd.conf エントリ

```
login stream      tcp6  nowait  root    /usr/sbin/in.rlogind  in.rlogind
telnet stream      tcp6  nowait  root    /usr/sbin/in.telnetd  in.telnetd
shell  stream      tcp    nowait  root    /usr/sbin/in.rshd     in.rshd
shell  stream      tcp6   nowait  root    /usr/sbin/in.rshd     in.rshd
exec   stream      tcp    nowait  root    /usr/sbin/in.rexecd   in.rexecd
exec   stream      tcp6   nowait  root    /usr/sbin/in.rexecd   in.rexecd
```

TCP ラッパーは、telnet などさまざまなネットワークサービスで入力要求を監視、フィルタ処理するためのパブリックドメインユーティリティです。以上のユーティリティの *server_pathname* として TCP ラッパーを指定する場合、TCP ラッパーが IPv6 対応である必要があります。対応していない場合、TCP ラッパーで使用するサービスの *proto* を tcp か udp に指定する必要があります。

また、Solaris ユーティリティを別の実装と入れ替える場合、そのサービスの実装が IPv6 をサポートしていることを確認する必要があります。サポートしていない場合、その実装の *proto* を tcp か udp に指定する必要があります。

注 – *proto* 値を tcp か udp のどちらか一方に指定すると、サービスでは IPv4 だけが使用されます。IPv4 接続と IPv6 接続の両方を有効にするには、*proto* 値を tcp6 か udp6 に指定する必要があります。サービスで IPv6 をサポートしていない場合、tcp6 や udp6 は指定しないでください。

ソケットを使用する IPv6 有効化サーバーについては、『プログラミングインタフェース』のソケット API への IPv6 拡張機能についての説明を参照してください。

既存のユーティリティに対する IPv6 拡張機能

ユーザーレベルインタフェースでは、次のユーティリティの組み込み拡張機能も変更されました。

- netstat (1M)
- snoop (1M)
- route (1M)
- ping (1M)
- traceroute (1M)

ifconfig (1M) ユーティリティも変更されました。詳細については、66 ページの「ifconfig ユーティリティに対する IPv6 拡張機能」を参照してください。

netstat (1M)

IPv4 ネットワーク状態の表示の他、netstat では IPv6 ネットワーク状態も表示できます。/etc/default/inet_type ファイルと -f コマンド行オプションで DEFAULT_IP 値を設定して、表示するプロトコル情報を選択できます。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 オプションで無効にできます。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。

新しい -p オプションでは、net-to-media テーブルが表示されます。これは、IPv4 用の ARP テーブルであり、IPv6 用の近傍キャッシュです。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。このコマンドの使用方法については、45 ページの「ネットワーク状態の表示方法」を参照してください。

snoop (1M)

snoop コマンドは、IPv4 パケットと IPv6 パケットの両方を取り込んで、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 プロトコルキーワードか ip6 プロトコルキーワードを指定すると、snoop コマンドは IPv4 パケットか IPv6 パケットのどちらかだけを表示します。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。このコマンドの使用方法については、50 ページの「IPv6 ネットワークトラフィックの監視方法」を参照してください。

route (1M)

このユーティリティは、IPv4 ルーターと IPv6 ルーターの両方で実行できます。デフォルトで、route は IPv4 ルートで実行します。コマンド行で route コマンドの直後にオプション `-inet6` を指定すると、操作が IPv6 ルートで実行されます。詳細については、route (1M) のマニュアルページを参照してください。

ping (1M)

ping コマンドは、IPv4 プロトコルと IPv6 プロトコルの両方で、宛先ホストを調べることができます。プロトコル選択は、指定の宛先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が、宛先ホストの IPv6 アドレスを戻すと、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、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、この動作を無効にできます。

そのほか、`-a` コマンド行オプションを指定すれば、マルチホーム宛先ホストのアドレスをすべて ping 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。このコマンドの使用方法については、50 ページの「すべてのマルチホームホストアドレスの探査方法」を参照してください。

traceroute (1M)

traceroute コマンドを使用して、指定ホストまでの IPv4 ルートと IPv6 ルートの両方をトレースできます。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。選択を無効にするには、`-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。traceroute (1M) のマニュアルページを参照してください。

表示出力の制御

netstat コマンドと ifconfig コマンドによる出力表示の方法を制御できます。

- コマンド行に追加したキーワードで、inet アドレスまたは inet6 アドレスを指定する
- `/etc/default/inet_type` ファイルの設定変数 `DEFAULT_IP` を設定する

`DEFAULT_IP` の値は、`IP_VERSION4`、`IP_VERSION6`、`BOTH` のどれかに設定できます。`DEFAULT_IP` を指定してこのファイルを作成しない場合、netstat と ifconfig では、両方のバージョンが表示されます。

注 – `inet` キーワードオプションと `inet6` キーワードオプションは、`netstat` コマンドと `ifconfig` コマンドの使用時に `inet_type` ファイルで設定した値を無効にします。

操作については、49 ページの「IPv6 関連コマンドの出力表示の制御方法」を参照してください。

IPv6 の Solaris トンネルインタフェース

注 – IPv4 ネットワーク経由で IPv6 パケットをカプセル化する方法としてもっとも望ましいのは、6to4 ルーティングを使用することです。6to4 は、`tun` を使用して自動的にトンネルを作成します。6to4 ルーティングの詳細は、93 ページの「移行機構としての 6to4」を参照してください。6to4 トンネルの設定作業については、38 ページの「6to4 ルーターの設定方法」を参照してください。

トンネルインタフェースのフォーマットは次のとおりです。

```
ip.tun ppa
```

`ppa` は物理的な接続ポイントです。

システム起動時に、トンネルモジュール (`tun`) は、(`ifconfig` によって) IP の最上位にプッシュされ、仮想インタフェースが作成されます。このプッシュは、`hostname6.*` ファイルを作成することによって行われます。

たとえば、IPv4 ネットワーク経由で IPv6 パケットをカプセル化するためのトンネルを作成するには、次のファイルを作成します。

```
/etc/hostname6.ip.tun0
```

このファイルの内容は、インタフェースが結合された後に `ifconfig(1M)` に渡されます。ポイントツーポイントトンネルの設定に必要なパラメータになります。

次のリストは、`hostname6.ip.tun0` ファイルのエントリの例です。

例 3-7 `hostname6.interface` エントリ

```
tsrc 120.68.100.23 tdst 120.68.7.19 up
addif 1234:1234::1 5678:5678::2 up
```

この例では、IPv4 ソースと宛先アドレスは、IPv6 リンクローカルアドレスを自動設定するためのトークンとして使用されます。これらのアドレスは、ip.tun0 インタフェースのソースと宛先アドレスです。次の2つのインタフェース ip.tun0 インタフェースと、論理インタフェース ip.tun0:1 が設定されます。論理インタフェースには、addif コマンドによって指定されたソースと宛先 IPv6 アドレスがあります。

すでに述べたとおり、システムをマルチユーザーとして起動すると、これらの設定ファイルの内容が変更されずに ifconfig に渡されます。上の例は次の内容と同じです。

```
# ifconfig ip.tun0 inet6 plumb
# ifconfig ip.tun0 inet6 tsrc 120.68.100.23 tdst 120.68.7.19 up
# ifconfig ip.tun0 inet6 addif 1234:1234::1 5678:5678::2 up
```

このトンネルにおける ifconfig -a の出力は次のとおりです。

```
ip.tun0: flags=2200850<UP,POINTOPOINT,RUNNING,MULTICAST,
NONUD,IPv6> mtu 1480 index 6
    inet tunnel src 120.68.100.23  tunnel dst 120.68.7.19
    inet6 fe80::c0a8:6417/10 --> fe80::c0a8:713
ip.tun0:1: flags=2200850<UP,POINTOPOINT,RUNNING,MULTICAST,NONUD,IPv6> mtu 1480
index 5
    inet6 1234:1234::1/128 --> 5678:5678::2
```

次の構文で設定ファイルに行を追加すれば、さらに論理インタフェースを設定できます。

```
addif IPv6-source IPv6-destination up
```

注 - トンネルのどちらかの端がトンネル経由で1つ以上のプレフィックスを通知する IPv6 ルータである場合、トンネル設定ファイルには addif コマンドは必要ありません。他のアドレスは自動設定されるため、必要とされる可能性があるのは tsrc と tdst だけです。

場合によっては、特定のトンネルについて、固有のソースリンクローカルアドレスと宛先リンクローカルアドレスを手動で設定する必要があることもあります。その場合、設定ファイルの最初の行を変更して、これらのリンクローカルアドレスを組み込みます。次に例を示します。

```
tsrc 120.68.100.23 tdst 120.68.7.19 fe80::1/10 fe80::2 up
```

ソースリンクローカルアドレスには、長さが10のプレフィックスがあります。この例では、ip.tun0 インタフェースは次のようになります。

```
ip.tun0: flags=2200850<UP,POINTOPOINT,RUNNING,MULTICAST,NONUD,IPv6> mtu 1480
index 6
    inet tunnel src 120.68.100.23  tunnel dst 120.68.7.19
    inet6 fe80::1/10 --> fe80::2
```

IPv6 ネットワーク (IPv6 経由の IPv6) 経由で IPv6 パケットをカプセル化するためのトンネルを作成するには、次のファイルを作成します。

```
/etc/hostname6.ip6.tun0
```

次のリストは、hostname6.ip6.tun0 ファイルのエントリの例です。

例 3-8 hostname6.interface エントリ

```
tsrc 2000::114:a00:20ff:fe72:668c tdst 2000::103:a00:20ff:fe9b:a1c3
fe80::4 fe80::61 up
```

IPv6 ネットワーク (IPv6 経由の IPv4) 経由で IPv4 パケットをカプセル化するためのトンネルを作成するには、次のファイルを作成します。

```
/etc/hostname.ip6.tun0
```

次のリストは、hostname.ip6.tun0 ファイルのエントリの例です。

例 3-9 hostname.interface エントリ

```
tsrc 2000::114:a00:20ff:fe72:668c tdst 2000::103:a00:20ff:fe9b:a1c3
10.0.0.4 10.0.0.61 up
```

IPv4 ネットワーク (IPv4 経由の IPv4) 経由で IPv4 パケットをカプセル化するためのトンネルを作成するには、次のファイルを作成します。

```
/etc/hostname.ip.tun0
```

次のリストは、hostname.ip.tun0 ファイルのエントリの例です。

例 3-10 hostname.interface エントリ

```
tsrc 120.46.86.158 tdst 120.46.86.122
10.0.0.4 10.0.0.61 up
```

tun の固有の情報については、tun(7M) のマニュアルページを参照してください。IPv6 への移行時のトンネルの概念の一般的な説明については、89 ページの「トンネル機構」を参照してください。トンネルの設定方法については、53 ページの「IPv4 トンネルによる IPv6 の設定方法」を参照してください。

Solaris ネームサービスに対する IPv6 拡張機能

ここでは、IPv6 の実装により導入されたネーミングの変更について説明します。IPv6 アドレスは Solaris ネームサービス (NIS、NIS+、DNS およびファイル) のどれでも保存できます。また、IPv6 RPC トランスポートで NIS と NIS+ を使用して NIS データまたは NIS+ データを検出することもできます。

/etc/inet/ipnodes ファイル

/etc/inet/ipnodes ファイルには、IPv4 と IPv6 のアドレスが格納されています。このファイルはローカルデータベースとして、ホスト名を IPv4 アドレスや IPv6 アドレスに関連付けます。ホスト名やそのアドレスは、/etc/inet/ipnodes などの静的ファイルには保存しないでください。ただし、テスト目的の場合は IPv4 アドレスを /etc/inet/hosts に保存するのと同じ方法で IPv6 アドレスを保存します。ipnodes ファイルでは、hosts ファイルと同じフォーマット変換を使用します。hosts ファイルの説明は、『Solaris のシステム管理 (IP サービス)』の「ネットワーク構成手順」を参照してください。ipnodes ファイルについては、ipnodes(4) のマニュアルページを参照してください。

IPv6-aware (IPv6 が利用可能な) ユーティリティでは、新しい /etc/inet/ipnodes データベースを使用します。既存の /etc/hosts データベースには、IPv4 アドレスだけを保存していますが、既存のアプリケーションの便宜上、このデータベースは変更されません。ipnodes データベースがない場合、IPv6-aware ユーティリティでは既存の hosts データベースを使用します。

注 - アドレスを追加する必要がある場合、IPv4 アドレスは hosts ファイルと ipnodes ファイルの両方に追加しなければなりません。IPv6 アドレスは ipnodes ファイルにだけ追加します。

例 - /etc/inet/ipnodes ファイル

```
#
# Internet IPv6 host table
# with both IPv4 and IPv6 addresses
#
::1      localhost
2::9255:a00:20ff:fe78:f37c  fripp.guitars.com fripp fripp-v6
fe80::a00:20ff:fe78:f37c   fripp-11.guitars.com fripp11
120.46.85.87              fripp.guitars.com fripp fripp-v4
2::9255:a00:20ff:fe87:9aba  strat.guitars.com strat strat-v6
fe80::a00:20ff:fe87:9aba   strat-11.guitars.com strat11
120.46.85.177             strat.guitars.com strat strat-v4 loghost
```

注 - 上記の例のように、ホスト名アドレスは、ホスト名でグループにまとめる必要があります。

IPv6 の NIS 拡張機能

NIS 用に 2 つの新しいマップが追加されました。ipnodes.byname と ipnodes.byaddr です。/etc/inet/ipnodes と同様に、これらのマップには、IPv4 情報と IPv6 情報の両方が保存されます。既存の hosts.byname と hosts.byaddr マップは、IPv4 情報だけを保存しています。既存のアプリケーションの便宜上変更されていません。

IPv6 の NIS+ 拡張機能

NIS+ 用に ipnodes.org_dir という新しいテーブルが追加されました。このテーブルには、ホスト用の IPv4 アドレスと IPv6 アドレスの両方が保存されています。既存の hosts.org_dir テーブルは IPv4 アドレス情報だけを保存しています。このテーブルは、既存のアプリケーションの便宜上変更されていません。

IPv6 の DNS 拡張機能

AAAA レコードとして定義された新しいリソースレコードが、RFC 1886 で定義されています。この AAAA レコードは、ホスト名を 128 ビット IPv6 アドレスにマップします。PTR レコードは IPv6 でも、IP アドレスをホスト名にマップするときに使用されています。128 ビットアドレスの 32 の 4 ビットニブルは、IPv6 アドレス用に反転されています。各ニブルは対応する 16 進 ASCII 値に変換されます。変換後、ip6.int が追加されます。

nsswitch.conf ファイルへの変更

/etc/inet/ipnodes で IPv6 アドレスを調べる機能に加え、IPv6 サポートは、NIS ネームサービス、NIS+ ネームサービス、DNS ネームサービスに追加されています。その結果、nsswitch.conf (4) ファイルは IPv6 検索をサポートするように変更されました。ipnodes 行が /etc/nsswitch.conf ファイルに追加されました。この追加により、Solaris ネームサービス (NIS、NIS+、DNS、ファイル) の新しいデータベースで検索が可能になりました。次の太字で示された行は、ipnodes エントリの例です。

```
hosts: files dns nisplus [NOTFOUND=return]
ipnodes: files dns nisplus [NOTFOUND=return]
```

注 - IPv4 アドレスと IPv6 アドレスでこれらの ipnodes データベースを生成してから、複数のネームサービスで ipnodes を探すように /etc/nsswitch.conf ファイルを変更してください。ホストアドレスの解決時に不要な遅延が発生してしまうからです (起動タイミングの遅れが発生することもあります)。

図 3-1 は、gethostbyname() コマンドと getipnodebyname() コマンドを使用するアプリケーションにおける、nsswitch.conf ファイルと新しいネームサービスデータベースの新しい関係を示します。斜体の項目は新規です。gethostbyname() コマンドは、/etc/inet/hosts に保存されている IPv4 アドレスだけを調べます。getipnodebyname() コマンドは、nsswitch.conf ファイルの ipnodes エントリで指定したデータベースを調べます。検索に失敗すると、nsswitch.conf ファイルの hosts エントリで指定したデータベースを調べます。

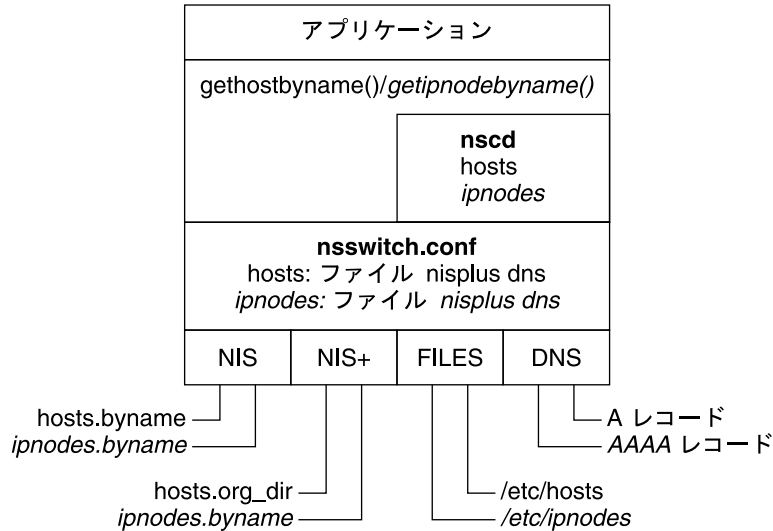


図 3-1 nsswitch.conf とネームサービスの関係

ネームサービスの詳細については、『Solaris のシステム管理 (ネーミングとディレクトリサービス : DNS、NIS、LDAP 編)』を参照してください。

ネームサービスコマンドの変更

IPv6 をサポートできるように、既存のネームサービスコマンドで IPv6 アドレスを調べることができます。たとえば、ypmatch コマンドは、新しい NIS マップに使用できます。nismatch コマンドは、新しい NIS+ テーブルに使用できます。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。ネームサービスの変更については、81 ページの「IPv6 の NIS 拡張機能」、81 ページの「IPv6 の NIS+ 拡張機能」、および 81 ページの「IPv6 の DNS 拡張機能」を参照してください。

これらのコマンドの使用手順については、58 ページの「IPv6 ネームサービス情報の表示」を参照してください。

NFS と RPC による IPv6 のサポート

NFS と RPC ソフトウェアは、シームレスに 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。

ATM 経由の IPV6 サポート

Solaris オペレーティング環境では、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。

第 4 章

IPv4 から IPv6 への移行 (リファレンス)

ホストとルーターを IPv6 にアップグレードする場合は、IPv4 ホストおよび IPv4 ルーターとの間で相互運用が可能ないようにこれらのノードを設定する必要があります。この章では、IPv4 から IPv6 へ移行するための標準的な方法について説明します。RFC 1933 でも、移行問題の詳しい解決法を示しています。

この章では、以下の内容について説明します。

- 85 ページの「移行条件」
- 86 ページの「標準移行ツール」
- 91 ページの「IPv4 と IPv6 の相互運用性」
- 92 ページの「サイト移行のシナリオ」
- 93 ページの「移行機構としての 6to4」
- 102 ページの「その他の移行機構」

移行条件

移行時のグローバルな調整は不要です。サイトとインターネットサービスプロバイダ (ISP) はそれぞれのスケジュールで移行できます。また、移行時の依存条件も最小限に抑えました。たとえば、ホストのアップグレード前にルーターを IPv6 にアップグレードしなくても移行できます。

サイトが異なれば、移行時にはそれぞれの制約が課されます。また、IPv6 の初期アダプタには、IPv6 の製品版ユーザーの場合とは異なる問題があります。RFC 1933 は現在利用できる移行ツールを定義しています。移行の必然性としては、IPv4 アドレス領域の不足または IPv6 の新機能を使用する必要性のどちらか、または両方が考えられます。IPv6 仕様では、移行時には、既存のプロトコルの完全な互換性が求められます。また既存のアプリケーションとの互換性も求められます。

移行方式を理解できるように、次の用語を定義します。

- IPv4 専用ノード – IPv4 だけを実装したホストやルーター。IPv4 専用ノードでは IPv6 は認識できない。移行以前に既存の IPv4 ホストとルーターのインストール可能ベースは IPv4 専用ノード
- IPv6/IPv4 ノード – IPv4 と IPv6 の両方を実装するホストとルーター。デュアルスタックとも呼ぶ
- IPv6 専用ノード – IPv6 を実装するホストまたはルーター。IPv4 を実装しない
- IPv6 ノード – IPv6 を実装するホストまたはルーター。IPv6/IPv4 ノードと IPv6 専用ノードは、どちらも IPv6 ノード
- IPv4 ノード – IPv4 を実装するホストまたはルーター。IPv6/IPv4 ノードと IPv4 専用ノードは、どちらも IPv6 ノード
- 6to4 ルーター – IPv4 ネットワーク接続上で 6to4 擬似インタフェースを使用して設定された任意の境界ルーター。6to4 ルーターは、ルーターがほかの IPv6 サイトへパケットを転送するために使用する 6to4 トンネルのエンドポイントとして機能する
- 6to4 ホスト – 6to4 派生アドレスを使用して設定されたインタフェースを持つ任意の IPv6 ホスト
- サイト – インターネットのプライベートトポロジの 1 つ。すなわちあらゆるユーザーを対象としたトラフィック伝送を行わないトポロジ。サイトが物理的に広範囲に展開されることがある。たとえば、多国籍企業のプライベートネットワークは、1 つのサイト

標準移行ツール

RFC 1933 は、次の移行方式を定義しています。

- ホストとルーターを IPv6 にアップグレードするとき、それらの IPv4 の機能を残す。したがって、すべての IPv4 プロトコルおよびアプリケーションとの互換性が確保される。このようなホストおよびルーターをデュアルスタックと呼ぶ
- IPv6 対応ノードに関する情報は、DNS などのネームサービスを利用して伝送する
- IPv6 アドレス形式には、IPv4 アドレスを保存する
- IPv4 パケットで IPv6 パケットをトンネル処理して、IPv6 にアップグレードされていないルーターを通過できる

デュアルスタックの実装

デュアルスタックとは、アプリケーションからネットワーク層に至るプロトコルスタックのすべてのレベルの完全な複製をいいます。デュアルスタックの例として、同じマシンで実行する OSI プロトコルと TCP/IP プロトコルがあります。ただし、IPv6

移行の観点からは、プロトコルスタックに IPv4 と IPv6 の両方を組み込むことを表します。残りスタックは同一となります。この場合、同じ伝送プロトコル (TCP、UDP など) が IPv4 と IPv6 の両方で実行します。また、同じアプリケーションも IPv4 と IPv6 の両方で実行します。

次の図は、OSI 層全体にわたるデュアルスタックプロトコルを表します。

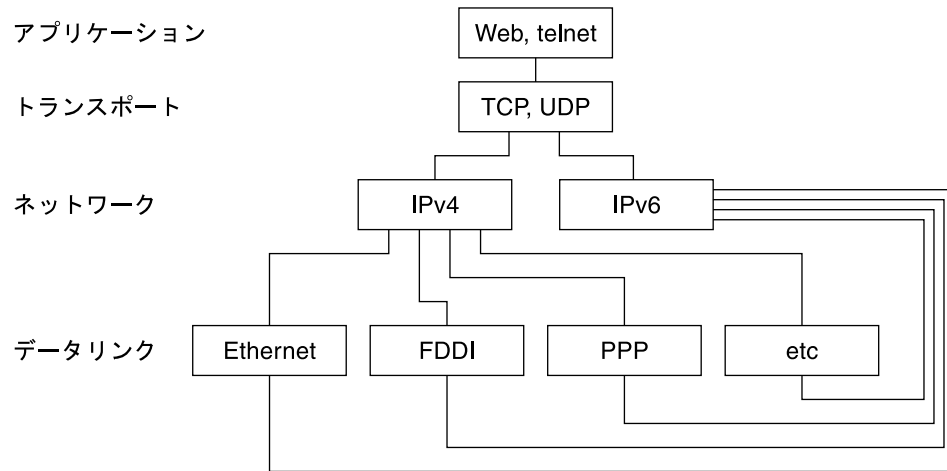


図 4-1 デュアルスタックプロトコル

デュアルスタック方式では、ホストとルーター両方のサブセットをアップグレードして、IPv4 に加えて IPv6 をサポートします。この方法では、アップグレードされた後のノードからも IPv4 で常に IPv4 専用ノードと相互運用できます。

ネームサービスの設定

デュアルノードでは、ピアが IPv6 と IPv4 のどちらをサポートしているか明確でないと、伝送時にどちらの IP バージョンを使用するのかが決まりません。そこで、ネームサービスでどんな情報を伝達するかを制御すると、デュアルノードで使用する IP バージョンを決定できます。さらに、ネームサービスで IPv4 ノードの IP アドレスと IPv6 ノードの IP アドレスを定義します。それによって、デュアルノードでは、両方のアドレスをネームサービスで使用できます。

IPv6 アドレスをネームサービスに指定した場合も、IPv6 でノードにアクセスできます。ただし、ノードにアクセスできるのは、ネームサービスから情報を得たノードだけです。たとえば、NIS に IPv6 アドレスを指定すると、その IPv6 ホストは IPv6 からアクセスできます。ただし、IPv6 ホストにアクセスできるのは、NIS ドメインに所属する IPv6 とデュアルノードだけです。グローバル DNS に IPv6 アドレスを指定するには、そのノードがインターネット IPv6 バックボーンからアクセスできることが条件です。これは、IPv4 の場合も同様です。たとえば、メール配信の操作は、IPv4

でアクセスできるノードの IPv4 アドレスがあるかどうか依存します。これは、HTTP プロキシの操作の場合も同様です。たとえば、ファイアウォールなどの理由で IPv4 でアクセスできない場合、ネームサービスは内部ファイアウォールと外部ファイアウォールのデータベースに分けます。これにより、IPv4 アドレスがアクセスできる範囲だけで認識できるようになります。

ネームサービスのアクセスに使用するプロトコルは、ネームサービスで検索できるアドレスタイプに依存しません。このネームサービスサポートとデュアルスタックでは、デュアルノードが IPv4 専用ノードと通信するときに、IPv4 を使用できます。さらに、IPv6 ノードと通信するときに、このネームサービスでは IPv6 を使用することができます。ただし、宛先までの IPv6 ルートが必要です。

IPv4 互換アドレスフォーマットの使用

通常 32 ビット IPv4 アドレスは、128 ビット IPv6 アドレスで表現できます。移行機能では、次の 2 つの形式を定義しています。

■ IPv4 互換アドレス

000 ... 000	IPv4 アドレス
-------------	-----------

■ IPv4 マップアドレス

000 ... 000	0xffff	IPv4 アドレス
-------------	--------	-----------

IPv6 ノードは互換フォーマットで表現します。このフォーマットでは、実際の IPv6 アドレスがなくても IPv6 ノードを使用できます。また、IPv4 専用ルーターで自動トンネルを使用できるため、このアドレスフォーマットではさまざまな IPv6 設定の試用が可能です。ただし、IPv6 ステータスアドレス自動設定機構では、このアドレスは設定できません。IPv6 ステータスアドレス自動設定機構には、DHCPv4 など既存の IPv4 機構や静的設定ファイルが必要なためです。

マップアドレスフォーマットでは、IPv4 ノードを表現します。現在ソケット API の一部でだけ、このアドレスフォーマットの使用方法が定義されています。アプリケーションでは、IPv6 アドレスと IPv4 アドレスの両方に共通のアドレスフォーマットを使用できます。共通のアドレスフォーマットは、IPv4 アドレスを 128 ビットマップアドレスで表現します。ただし、IPv4 プロトコルトランスレータと IPv6 プロトコルトランスレータがないと、これらのアドレスは使用できません。

トンネル機構

移行時の依存状態を最小限に抑える目的から、2つのIPv6ノード間にあるすべてのルーターでIPv6をサポートする必要がありません。この機構をトンネルといいます。基本的にIPv6パケットはIPv4パケット内部に組み込まれ、IPv4ルーター間を転送されます。以下の図は、IPv4ルーター(R)間のトンネル機構を示します。

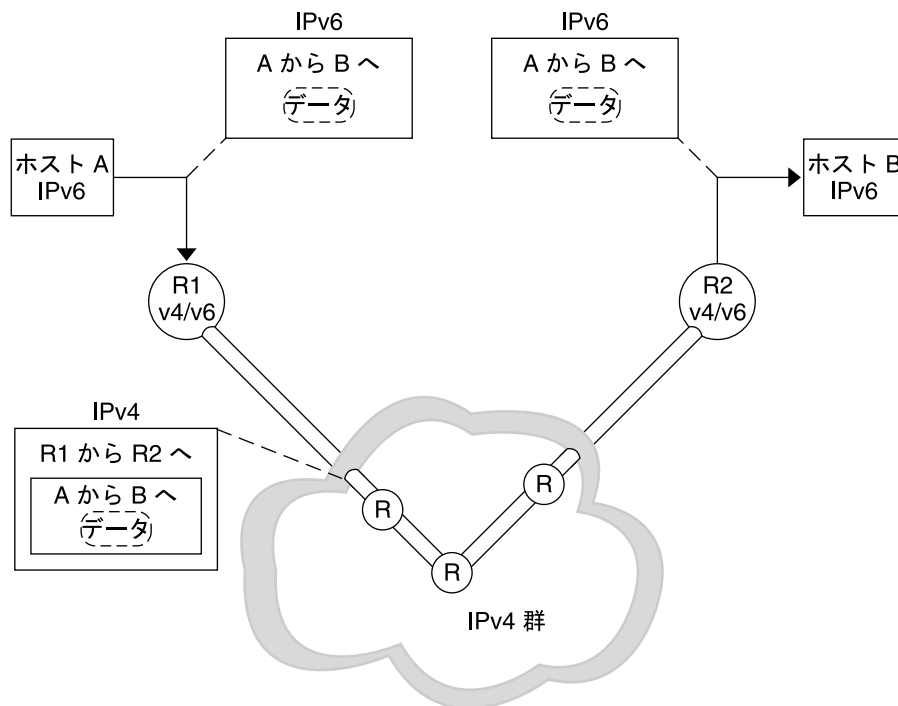


図 4-2 トンネル機構

その他、移行時には次のようなトンネル機構の使用方法があります。

- 2つのルーター間で設定したトンネル(上記の図を参照)
- デュアルホストで終了する自動トンネル

設定トンネルは、MBONE (IPv4 マルチキャストバックボーン) など現在はインターネットで他の目的に使用します。設定トンネルの作成手順からいうと、2つのルーターを設定して、その間に IPv4 ネットワーク経由の仮想ポイントツーポイントリンクを作成します。近い将来インターネットのさまざまな局面にこの種のトンネルが利用されるでしょう。

自動トンネル

注 – 自動トンネルは、6to4 トンネリングを通して作成することをお勧めします。6to4 ルーティングと 6to4 トンネリングの機構詳細は、93 ページの「移行機構としての 6to4」を参照してください。

自動トンネルは、IPv4 – 互換アドレスを要求します。自動トンネルは、IPv6 ルータが使用できない場合に IPv6 ノードを接続するために使用できます。自動トンネル ネットワークインタフェースを設定すれば、トンネルの発信元はデュアルホストとデュアルルーターのどちらが発信元でも使用できます。終点は必ずデュアルホストになります。トンネルのはたらきにより、宛先 IPv4 アドレス (トンネルの終点) が IPv4 互換宛先アドレスから抽出されて動的に指定されます。

アプリケーションとの対話

IPv6 にアップグレードしたノードでも、IPv6 を使用できるかどうかはアプリケーション次第です。アプリケーションで、IPv6 アドレスのネームサービスを要求する ネットワーキング API を使用しない場合があります。また、アプリケーション側で変更が必要な API (ソケットなど) を使用する場合があります。さらに、API のプロバイダ (java.net クラスなどの実装) が IPv6 アドレスをサポートしていない場合もあります。どの場合も、ノードが送受信するのは IPv4 ノードのように IPv4 パケットだけです。

次の用語は、インターネットの世界では標準用語として使用されています。

- **IPv6-unaware (非認識)** – IPv6 アドレスを処理できないアプリケーション。IPv4 アドレスのないノードとは通信できない
- **IPv6-aware (認識)** – IPv4 アドレスがないノードとも通信できるアプリケーション。長い IPv6 アドレスも処理できる。アプリケーションに透過な場合がある。たとえば実際のアドレスの内容や形式が API によって非表示になる場合など
- **IPv6-enabled (有効化)** – IPv6-aware であるだけでなく、フローラベルなど IPv6 固有の機能が利用できる。有効化アプリケーションは低下モードで IPv4 も処理できる
- **IPv6-required (要求)** — このアプリケーションは、IPv6 – 特定機能をいくつか要求する。このアプリケーションは IPv4 では作動できない

IPv4 と IPv6 の相互運用性

IPv4 から IPv6 に段階的に移行する場合、新しく導入する IPv6 有効化アプリケーションと共に既存の IPv4 アプリケーションも使用しなければなりません。最初の段階で、ベンダーは、デュアルスタックを実行しているホストとルータのプラットフォームを提供します。デュアルスタックとは、IPv4 プロトコルスタックと IPv6 プロトコルスタックのことです。IPv4 アプリケーションは、少なくとも 1 つの IPv6 インタフェースで IPv6 有効化になっているデュアルスタックでも実行できます。アプリケーションの変更や移植は不要です。

デュアルスタックで実行する IPv6 アプリケーションも、IPv4 プロトコルを使用できます。IPv6 アプリケーションは、IPv4 マップ IPv6 アドレスを使用します。IPv6 は設計上、IPv4 と IPv6 で別々のアプリケーションは不要です。たとえば、デュアルホストの IPv4 クライアントがなくても IPv4 専用ホストのサーバーと「通信」できます。また独立した IPv6 クライアントがなくても IPv6 サーバーと通信できます。IPv4 クライアントアプリケーションを新しい IPv6 API に移植するだけです。クライアントは、IPv4 専用サーバーと通信できます。また、デュアルホストまたは IPv6 専用ホストで実行中の IPv6 サーバーとも通信できます。

ネームサーバーからクライアントが取り出すアドレスで、IPv6 や IPv4 を使用するかどうかが決まります。たとえば、ネームサーバーにそのサーバーの IPv6 アドレスが指定されている場合、サーバーは IPv6 を処理できます。

表 4-1 に IPv4 と IPv6 のクライアントとサーバー間の相互運用性をまとめます。表 4-1 では、デュアルスタックホストに、IPv4 と IPv6 両方のアドレスがそれぞれのネームサービスデータベースに存在するものとしてします。

表 4-1 クライアントサーバーアプリケーション: IPv4 と IPv6 の相互運用性

アプリケーションの種類 (ノードの種類)	IPv6-unaware (非認識) サーバー (IPv4 専用ノード)	IPv6-unaware (非認識) サーバー (IPv6 有効化ノード)	IPv6-aware (認識) サーバー (IPv6 専用ノード)	IPv6-aware (認識) サーバー (IPv6 有効化ノード)
IPv6-unaware (非認識) クライアント (IPv4 専用ノード)	IPv4	IPv4	X	IPv4
IPv6-unaware (非認識) クライアント (IPv6 有効化ノード)	IPv4	IPv4	X	IPv4
IPv6-aware (認識) クライアント (IPv6 専用ノード)	X	X	IPv6	IPv6
IPv6-aware (認識) クライアント (IPv6 有効化ノード)	IPv4	(IPv4)	IPv6	IPv6

X は、それぞれのサーバーとクライアント間の通信ができないことを表します。

(IPv4) は、クライアントの選択するアドレスによって相互運用性が決まることを表します。IPv6 アドレスを選択すると、クライアントの処理はエラーになります。ただし、IPv4 アドレスを選択すると、IPv4 マップ IPv6 アドレスとしてクライアントに戻り、IPv4 データグラム が送信されて処理が成功します。

IPv6 配置の初期段階では、IPv6 のほとんどの実装がデュアルスタックノードで処理されます。一般ベンダーではほとんど、初期状態では IPv6 専用実装をリリースしません。

サイト移行のシナリオ

サイトや ISP では、それぞれ事情が異なり、移行段階の手順が異なります。ここでは、サイト移行シナリオの例をいくつか紹介します。

IPv6 へのサイトの移行では、最初に IPv6 アドレスをサポートするためのネームサービスをアップグレードします。DNS の場合、BIND 4.9.4 以降などの新しい AAAA (クアドA) レコードをサポートする DNS サーバーにアップグレードします。2つの新しい NIS マップと NIS+ テーブルが IPv6 アドレスを保存するために導入されました。これらの新しい NIS マップと NIS+ テーブルは、任意の Solaris システムで作成、管理できます。新しいデータベースの詳細については、79 ページの「Solaris ネームサービスに対する IPv6 拡張機能」を参照してください。

ネームサービスで IPv6 アドレスを処理できるようになったら、ホストの移行を開始します。ホストは、次の手順で移行します。

- ホストを1つずつアップグレードします。IPv4 互換アドレスと自動トンネルを使用します。ルーターのアップグレードは不要です。この方法は最初の試験的な移行に適しています。IPv6 の機能のすべてが利用できるわけではありません。したがって、ステートレスアドレス自動設定や IP マルチキャストは利用できません。このシナリオはアプリケーションが IPv6 で実行できるかどうかを確認するときに使用します。また、アプリケーションが IPv6 IP 層セキュリティを利用できるかどうかを確認するときも使用します。
- サブネットを1つずつアップグレードします。ルーター間に設定したトンネルを使用します。このシナリオでは、サブネットごとに少なくとも1つのルーターをデュアルにアップグレードします。サイト内のデュアルルーターは設定したトンネルで結合します。これで、サブネット上のホストでは、IPv6 の全機能を利用できます。このように段階的にアップグレードしていく中で徐々にアップグレードされるルーターが増加するとともに、設定済みのトンネルは削除できます。
- ホストをアップグレードする前にすべてのルーターをデュアルにアップグレードします。この方法は逐次行われるように思えますが、すべてのルーターがアップグレードされるまでは IPv6 の機能を利用できません。このシナリオでは、段階的な配置方式は制約されます。

移行機構としての 6to4

Solaris オペレーティングシステムには、IPv4 アドレス指定から IPv6 アドレス指定へ移行するための暫定的な推奨方法として 6to4 が含まれています。独立した IPv6 サイトで 6to4 を利用すると、IPv6 をサポートしない IPv4 ネットワーク上で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、IPv6 ネットワーク上の境界ルーターを 6to4 自動トンネルの一方のエンドポイントとして設定する必要があります。そのあと、この 6to4 ルーターをほかの 6to4 サイトとの間のトンネルの構成要素として使用することも、あるいは必要に応じて 6to4 以外のネイティブ IPv6 サイトとの間のトンネルで使用することもできます。

この節では、6to4 に関連した以下の参考情報を示します。

- 6to4 トンネルのトポロジ
- 6to4 アドレス指定 (通知の書式など)
- 6to4 トンネルを介したパケットフローの説明
- 6to4 ルーターと 6to4 リレールーター間のトンネルのトポロジ
- 6to4 リレールーターサポートを設定する前の考慮事項

6to4 ルーティングについての詳細は、以下のページ / 文書を参照してください。

作業または技術情報	参照先
6to4 ルーティングの概要	33 ページの「IPv4 ネットワーク経由の 6to4 トンネル」
6to4 サイトの設定作業	38 ページの「6to4 ルーターの設定方法」
6to4 関連の RFC、『Connection of IPv6 Domains via IPv4 Clouds』	RFC 3056、 『Connection of IPv6 Domains via IPv4 Clouds』
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) のマニュアルページ
6to4 のセキュリティ問題についてのインターネット文書、『Security Considerations for 6to4』	『Security Considerations for 6to4』

6to4 トンネルの構成

次の図は、2つの 6to4 サイト間の 6to4 トンネルを示しています。

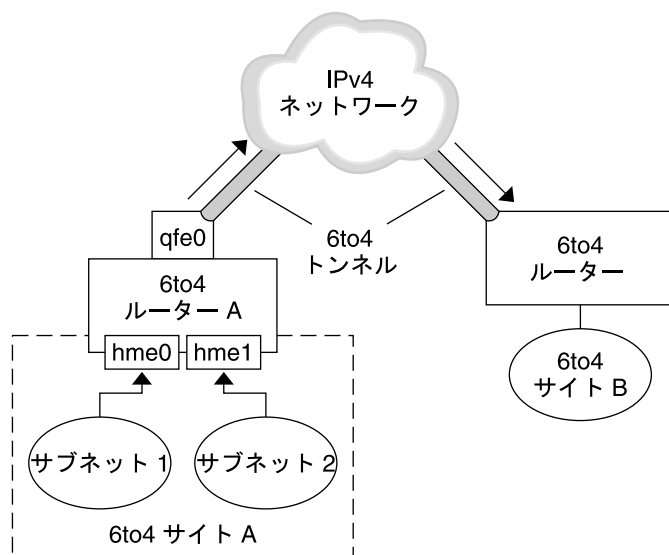


図 4-3 2つの 6to4 サイト間のトンネル

この図は、独立した2つの 6to4 ネットワーク、サイト A とサイト B を示しています。各サイトは、IPv4 ネットワークに外部接続するようにルーターを設定してあります。この図では、IPv4 ネットワークを介した 6to4 トンネルによって 6to4 サイトが接続されています。

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ーフェースを設定する必要があります。このインターフェースは、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 で設定するアドレスは、一意(世界で唯一)のものでなければなりません。上記の図では、境界ルーター A のインターフェース qfe0 によってサイト A が IPv4 ネットワークに接続されています。qfe0 を 6to4 擬似インターフェースとして設定するには、IPv4 アドレスを使用してあらかじめインターフェース qfe0 を設定しておく必要があります。

上記の図では、6to4 サイト A はルーター A 上のインターフェース hme0 と hme1 に接続した2つのサブネットから構成されています。サイト A の両サブネットのすべての IPv6 ホストは、ルーター A からの通知を受け取ると 6to4 派生アドレスを使用して自動的に再設定を行います。

サイト B は、サイト A からのトンネルの反対側のエンドポイントにあたります。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設定する必要があります。この設定を行わないと、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 派生のアドレス指定

ネイティブ IPv6 ルーターの場合と同様に、`/etc/inet/ndpd.conf` 中の 6to4 プレフィックスから派生したサブネットプレフィックスを通知する必要があります。次の図は、6to4 サイトのプレフィックスの構成を示しています。各部の説明は、96 ページの「6to4 プレフィックスの書式」と96 ページの「6to4 通知の例」を参照してください。

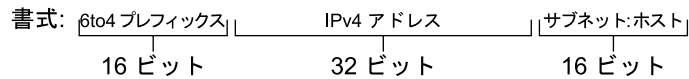
書式: $\underbrace{\text{6to4 プレフィックス}}_{16 \text{ ビット}} \mid \underbrace{\text{IPv4 アドレス}}_{32 \text{ ビット}}$

6to4 アドレスの例: `2002:8192:5666::/48`

書式例の説明: $\underbrace{\text{2002}}_{\text{プレフィックス}} \mid \underbrace{\text{8192.5666}}_{\text{IPv4 アドレス}} \mid \underbrace{\text{/48}}_{\text{プレフィックスの長さ (48 ビット)}}$

図 4-4 サイトプレフィックスの構成

次の図は、`ndpd.conf` ファイルに指定する 6to4 サイトのサブネットプレフィックスの構成を示しています。



6to4 アドレスの例: 2002:8192.5666:1: :/64

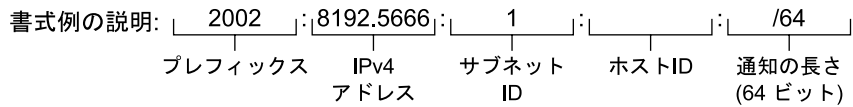


図 4-5 サブネットプレフィックスの構成

6to4 プレフィックスの書式

上記の図に示した書式には以下の要素が含まれます。

構成要素	長さ	定義
プレフィックス	16 ビット	6to4 プレフィックス 2002 (0x2002)
IPv4 アドレス	32 ビット	6to4 インタフェースですでに設定されている一意の IPv4 アドレス。通知のために、ドット付きの 10 進表記ではなく 16 進表記の IPv4 アドレスを指定する
Subnet ID	16 ビット	サブネット ID。これは、6to4 サイトにおけるリンクで一意となる値でなければならない

6to4 通知の例

上記図の例では以下の値を使用しています。

通知の要素	対応する値
6to4 プレフィックス	2002
IPv4 アドレス	8192:56bb。これは IPv4 アドレス 129.146.87.188 に対応するもの
Subnet ID	1

通知の要素	対応する値
/64	プレフィックスの長さ

ホストにおける 6to4 派生のアドレス指定

ルーター通知によって 6to4 派生のプレフィックスを受信する際、IPv6 ホストはインタフェース上の 6to4 派生アドレスを自動的に設定し直します。このアドレスの書式は以下のとおりです。

prefix:IPv4 address:subnet ID:host ID/64

6to4 インタフェースを持つホストで `ifconfig -a` を実行すると、以下のような結果となります。

```
qfe1:3: flags=2180841<UP,RUNNING,MULTICAST,ADDRCONF,ROUTER,IPv6>
mtu 1500 index 7
    inet6 2002:8192:56bb:9258:a00:20ff:fea9:4521/64
```

6to4 派生アドレスは、`ifconfig` の出力結果の中で `inet6` のあとに続きます。

アドレスの構成要素	対応する値
<i>Prefix</i>	2002。これは 6to4 プレフィックス
<i>IPv4 value</i>	8192:56bb。これは、6to4 ルーターで設定されている 6to4 擬似インタフェースに対する 16 進表記による IPv4 アドレス
<i>subnet ID</i>	9258。これは、このホストの所属先であるサブネットのアドレス
<i>MAC address</i>	a00:20ff:fea9:4521。これは、現在 6to4 に設定されているホストインタフェースのリンク層アドレス

6to4 トンネルを介したパケットフロー

この節では、1つの 6to4 サイトにあるホストから遠隔の 6to4 サイト内のホストへのパケットの経路について説明します。次のシナリオは、例として図 4-3 で示しているトポロジを使用しています。このシナリオは、6to4 ルーターと 6to4 ホストがすでに設定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。フロー内の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. 6to4 ルーター A は、出力パケットを受け取り、IPv4 ネットワーク上で 6to4 サイト B との間のトンネルを構築します。

3. サイト A のルーターは、IPv4 ヘッダーを付けて各 6to4 パケットをカプセル化します。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。
4. パケットが遭遇する IPv4 ルーターが、パケットの宛先 IPv4 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。
5. 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。
6. 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ため、Solaris オペレーティングシステム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以下に示すようなトンネリングを有効にすることができます。

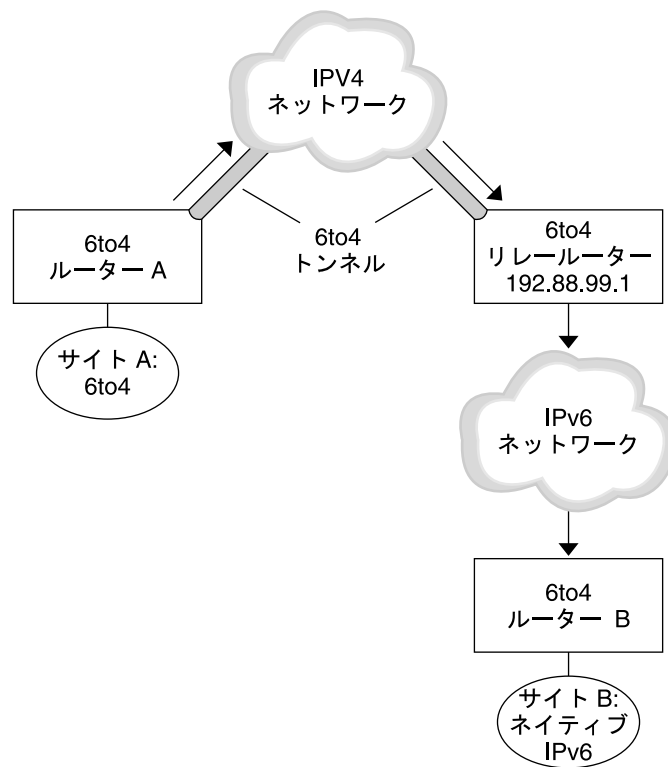


図 4-6 6to4 サイトと 6to4 リレールーター間のトンネル

図 4-6 では、6to4 サイト A はネイティブ IPv6 サイト B に存在するノードと通信を行う必要があります。この図は、サイト A から IPv4 ネットワーク経由の 6to4 トンネルへと続くトラフィックの経路を示しています。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

この節では、6to4 サイトからネイティブ IPv6 サイトに対するパケットフローについて説明します。ここでは、図 4-6 で示されているシナリオを例として使用しています。

1. 6to4 サイト A 上のホストから、ネイティブ IPv6 サイト B 上のホストに向けて伝送を行います。フロー内の各パケットヘッダーには、その発信元アドレスとして 6to4 派生アドレスが指定されています。宛先アドレスは標準の IPv6 アドレスです。

2. 6to4 ルーター A は、出力パケットを受け取り、IPv4 ネットワーク上で 6to4 リレールーターとの間のトンネルを構築します。

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、デフォルトを上書きしてそのルーターの IPv4 アドレスを指定できます。
3. 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 ヘッダーとともにカプセル化します。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。
4. 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。
5. 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。
6. 続いてこのリレールーターは、IPv6 のみとなったパケットを IPv6 ネットワークに送信します。IPv6 ネットワークにおいて、パケットは最終的に Site B のルーターによって検出されます。続いてこのルーターが、宛先である IPv6 ノードにパケットを転送します。

6to4 リレールーターサポートのセキュリティ問題

本来、6to4 ルーターと 6to4 リレールーター間のトンネルは安全ではありません。これらのルーター間のトンネルには、以下のようなセキュリティ問題が内在しています。

- 6to4 リレールーターはパケットのカプセル化とカプセル化の解除を行います。パケット内に含まれるデータのチェックは行いません。
- アドレスのスプーフィングは、6to4 リレールーターとの間で構築されるトンネルにおける際立った問題です。着信トラックについては、6to4 ルーターはリレールーターの IPv4 アドレスを送信元の IPv6 アドレスと対応させることができないという問題があります。このため、IPv6 ホストのアドレスは簡単にスプーフィングされかねません。6to4 リレールーターのアドレスもスプーフィングの可能性があります。
- デフォルトの設定では、6to4 ルーターと 6to4 リレールーター間に信頼できるメカニズムは存在しません。したがって、6to4 ルーターは 6to4 リレールーターが信頼できるものであるかどうかを識別できず、正規の 6to4 リレールーターであるかすら確認できません。このようなことから、6to4 サイトと宛先の IPv6 サイト間に信頼関係が存在していることか、あるいは攻撃を受けるという可能性を両サイトとも受け入れることが求められます。

これらの問題を始めとする 6to4 リレールーターのセキュリティ問題については、インターネット文書『*Security Considerations for 6to4*』で説明されています。一般には、6to4 リレールーターのサポートは以下のような場合だけ検討してください。

- 信頼できるプライベートな IPv6 ネットワークとの間で 6to4 サイトが通信を行う場合。たとえば、独立した 6to4 サイトとネイティブ IPv6 サイトから構成されるキャンパスネットワーク上などでこのサポートを有効にすると便利かもしれません。
- ビジネス上の理由で、6to4 サイトと特定のネイティブ IPv6 ホストとの通信を避けることができない場合。
- インターネット文書『*Security Considerations for 6to4*』で提唱されている検査と信頼できるモデルを導入した場合。

6to4 ルーターの既知の問題

6to4 構成は、以下に示す既知のバグの影響を受けます。

- 4709338 – 静的な経路を認識する RIPng 実装が必要である
- 4152864 – 同じ tsrc と tdst のペアによる 2 つのトンネルの設定

6to4 サイトにおける静的な経路の実装 (バグ ID 4709338)

6to4 境界ルーターが設置された 6to4 サイトでは、以下の問題が発生します。6to4 擬似インタフェースを設定する場合に 6to4 ルーターのルーティングテーブルに静的経路 2002::/16 が自動的に追加されます。バグ 4709338 は、この静的経路が 6to4 サイトに通知されることを防ぐ Solaris RIPng ルーティングプロトコルにおける制限について説明しています。

バグ 4709338 に対しては、以下に示す回避策のどちらかを適用できます。

- 6to4 サイト内のイントラサイトルーターすべてのルーティングテーブルに静的経路 2002::/16 を追加する
- 6to4 サイトの内部ルーターに RIPng 以外のルーティングプロトコルを使用する

同じ発信元アドレスによるトンネルの設定 (バグ ID 4152864)

バグ ID 4152864 は、同じトンネル発信元アドレスを使用して 2 つのトンネルが設定される場合に発生する問題について説明しています。これは、6to4 トンネルの重大な問題です。



注意 – 同じトンネル発信元アドレスを使用して 6to4 トンネルと自動トンネルを設定することは避けてください。

その他の移行機構

先に説明した方法では、デュアルノードと IPv4 ノード間で相互運用をします。その場合、デュアルノードには IPv4 アドレスがあります。ただし、その方法では、IPv6 専用ノードと IPv4 専用ノードの間で相互運用しません。また、IPv4 アドレスのないデュアルノードと IPv4 専用ノード間でも相互運用しません。ほとんどの実装ではデュアルにできます。ただし、デュアル実装には、IPv4 専用ノードとの相互運用が必要なすべてのノードごとに1つのアドレスを割り当てるのに十分な IPv4 アドレス領域が必要です。

次に、新しい移行機構がなくても相互運用を実現できる方法を示します。

- IPv6 専用ノードとインターネットの他の要素との間にアプリケーション層ゲートウェイ (ALG) を配置する。現在使用されている ALG としては、HTTP プロキシとメールリレーがある
- IPv4 用の、NAT ボックス (ネットワークアドレストランスレータ) をすでに売り出している会社もある。これは、内部のプライベート IP アドレス (ネットワーク 10 など。RFC 1918 参照) と、外部の IP アドレスの間の変換を行う。このような会社では、IPv6 から IPv4 アドレスへの変換もサポートするように、NAT ボックスをアップグレードする可能性が高い

残念ながら、ALG と NAT のどちらの方法も、弱点があります。これらの方法を使用すると、インターネットの基盤がかなり弱まります。IETF では、IPv6 専用ノードと IPv4 専用ノードとのより良い相互運用性のために努力しています。1つの提案としては、必要に応じて IPv4 互換アドレスを割り当てる方法でヘッダトランスレータを使用する方法があります。別の方法としては、必要に応じて IPv4 互換アドレスを割り当て、IPv6 トンネルで IPv4 を利用して IPv6 専用ルーターをブリッジできます。

ステートレスヘッダトランスレータでは、使用中の IPv6 アドレスを IPv4 アドレスとして表現できれば、IPv4 ヘッダーフォーマットと IPv6 ヘッダーフォーマットの間の変換が可能です。つまり、アドレスは、IPv4 互換アドレスである必要があります。もしくは、IPv4 マップアドレスである必要があります。これらのトランスレータのサポートは、IPv6 プロトコルに組み込まれています。暗号化されているパケットを除いて、変換時に情報は失われません。ソースルーティングなどの使用頻度の低い機能は、情報が失われてしまうことがあります。

用語集

この用語集には、このマニュアルで新たに使用した、『Sun Global Glossary』にはない用語の説明だけが記載されています。その他の用語の説明については、『Sun Global Glossary』を参照してください。

DES	Data Encryption Standard。1975年に開発され、1981年にANSI X.3.92としてANSIで標準化された対称鍵の暗号化方式。DESでは56ビットの鍵を使用する。
IPsec	IPデータグラムを保護するためのセキュリティアーキテクチャ。
IPv4	インターネットプロトコルバージョン4。IPv4は、しばしばIPとして参照される。このバージョンは32ビットのアドレス領域を提供する。
IPv6	インターネットプロトコルバージョン6。このバージョンは128ビットのアドレス領域を提供する。
IP内IPカプセル化	IPパケット内のIPパケットをトンネリングするための機構
IPリンク	リンク層でノード間の通信に使用される通信設備や通信媒体。リンク層は、IPv4/IPv6のすぐ下にある。例としては、Ethernet(単一、またはブリッジ)またはATMネットワークがある。1つまたは複数のIPv4サブネット番号またはネットワーク接頭辞がIPリンクに割り当てられる。同じサブネット番号またはネットワーク接頭辞を複数のIPリンクに割り当てることはできない。ATM LANEでは、IPリンクは1つのエミュレートされたLANである。ARPを使用する場合、ARPプロトコルの有効範囲は単一のIPリンクである。
MD5	デジタル署名などのメッセージ認証に使用する繰り返し暗号化のハッシュ関数。1991年にRivest氏によって開発された。
MTU	最大転送単位。リンクに転送できるサイズ(オクテット単位)。たとえば、EthernetのMTUは1500オクテット。
PKI	Public Key Infrastructure。インターネットトランザクションに関係する各関係者の有効性を確認および承認する、デジタル署名、認証局、他の登録機関のシステム。

RSA	デジタル署名と公開鍵暗号化システムを取得するための方法。1978年に最初に公開され、Rivest氏、Shamir氏、Adleman氏によって開発された。
SADB	セキュリティアソシエーションデータベース。暗号化鍵とアルゴリズムを指定するテーブル。鍵とアルゴリズムは、安全なデータ転送で使用される。
SHA-1 アルゴリズム	セキュリティ保護されたハッシュアルゴリズム。メッセージ要約を作成するために2 ⁶⁴ 文字以下の長さを入力するときに操作する。これはDSAへの入力となる。
SPI	セキュリティパラメータインデックス。受信したパケットの暗号解除に受信側が使用するSADBの行を指定する整数
Triple-DES	Triple-Data Encryption Standard。168ビットの鍵を提供する対称鍵暗号化方法。
エニーキャストアドレス	(一般的に別のノードに属す)複数のインタフェースに割り当てられるIPアドレス。エニーキャストアドレスに送られたパケットは、そのアドレスを持つ、プロトコルに基づき「最も近い」インタフェースに配送される。パケットの経路指定は、経路指定プロトコルの距離測定に応じて決定される。
回復検出	障害の発生後、NICやNICから第3層の装置への経路が、正しく動作し始めたことを検出する処理
鍵管理	セキュリティアソシエーションを管理するための手法。
カプセル化	ヘッダーとペイロードを1番目のパケット内に配置し、そのパケットを2番目のパケットのペイロード内に配置すること。
カプセル化セキュリティヘッダー	IPv6データグラムに対して認証と完全性を提供する拡張ヘッダー。
逆方向トンネル	モバイルノードの気付アドレスで始まり、ホームエージェントで終わるトンネル
近傍検索	接続されているリンク上にある他のホストをホストが特定できるようにするためのIPメカニズム。
近傍通知	近傍要請メッセージに対する応答、またはデータリンク層アドレスの変更を通知するために、ノードが自発的に近傍通知メッセージを送ること。
近傍要請	近傍のリンク層アドレスを決定するために、ノードによって送信される要請。また、キャッシュされたリンク層アドレスによって近傍が到達可能であるかを確認する。
公開鍵暗号化	次の2つの鍵を使用する暗号化システム。公開鍵は相互に認識している。非公開鍵は、メッセージの受信者だけが認識している。IKEにより、IPsecの公開鍵が提供される。

最小カプセル化	ホームエージェント、外来エージェント、およびモバイルノードによってサポートされる任意の形態の IPv4 内 IPv4 トンネリング。最小カプセル化は、IP 内 IP カプセル化よりも 8 ないし 12 バイト少ないオーバーヘッドしか持たない。
サイトローカルアドレス	単一サイト上でアドレスを指定するために使用する。
自動設定	IPv6 において、ホストが自身のインタフェースを自動的に設定すること。
順方向トンネル	ホームエージェントから始まり、モバイルノードの気付アドレスで終わるトンネル
ステートフル自動設定	ホストが、インタフェースアドレスや設定情報、およびパラメータをサーバーから取得すること。
ステートレス自動設定	ホストが、ローカルに入手可能な情報と、ルーターが通知した情報を組み合わせて自身のアドレスを生成すること。
セキュリティアソシエーション (SA)	1 つのホストから 2 番目のホストに、セキュリティ属性を指定するアソシエーション。
セキュリティパラメータインデックス (SPI)	受信者が受信したパケットを復号化するために使用する必要がある SADB (セキュリティアソシエーションデータベース) 内の行を特定する整数値
専用アドレス	インターネット経由で経路指定ができない IP アドレス
双方向トンネル	双方向にデータグラムを送信するトンネル
待機	グループ内の他の物理インタフェースに障害が発生するまでデータの伝送には使用されない物理インタフェース
対称鍵暗号化	送信側と受信側が単一の共通鍵を使用する暗号化システム。この共通鍵は、メッセージの暗号解除に使用する。対称鍵は、IPsec での大量データ転送の暗号化に使用する。対称鍵システムの一例として DES がある。
デュアルスタック	IPv6 への移行に使用する、IPv4 と IPv6 の両機能を併せ持つプロトコルスタックで、スタックの残り部分は同じ。
登録	モバイルノードが、ホームにないときに自分の気付アドレスを自分のホームエージェントおよび外来エージェントに登録すること。
トンネル	カプセル化される間データグラムが通過するパス
認証局 (CA)	デジタル署名および公開鍵と非公開鍵のペアの作成に使用するデジタル証明書を発行する、公証された第三者機関または企業。CA は、一意の証明書を付与された個人の ID を保証する。
認証ヘッダー	IP データグラムに対し認証と完全性を提供する拡張ヘッダー。機密性は提供されない。
ネットワークアクセス識別子 (NAI)	user@domain 形式でモバイルノードを一意に特定するために使用する。

ネットワークインタフェースカード (NIC)	リンクとのインタフェースになる、内部ネットワークアダプタおよび独立したネットワークアダプタカード
ノード	ホストまたはルーター
パケット	通信回線上で、1 単位として送られる情報の集合。ヘッダーとペイロードで構成される。
汎用経路指定カプセル化 (GRE)	ホームエージェント、外来エージェント、およびモバイルノードによってサポートされる任意の形態のトンネリング。他の任意の (または同じ) ネットワーク層プロトコルの配信パケット内で任意のネットワーク層プロトコルのパケットをカプセル化できるようにする。
ファイアウォール	組織内の私的ネットワークまたはイントラネットを、インターネットなどの外部ネットワークからの侵入に対して保護する装置またはソフトウェア。
物理インタフェース	リンクに対するノードの接続。この接続は通常、デバイスドライバとネットワークアダプタとして実装される。ネットワークアダプタによっては、qfe のように複数の接続点を持つ場合もある。このマニュアルでは、「ネットワークアダプタ」は「単一接続点」を示す。
物理インタフェースグループ	同じリンクに接続されている、システムの物理インタフェース群。グループ内のすべての物理インタフェースには、識別のための空文字列でない同じ名前が割り当てられる。
物理インタフェースグループ名	グループを識別する、物理インタフェースに割り当てられる名前。この名前の有効範囲は 1 つのシステム。同じグループ名を共有する複数の物理インタフェースは、物理インタフェースグループを構成する
ホップ	2 つのホストを分離するルーターの数を判別するための手段。たとえば、始点ホストと終点ホストが 3 つのルーターで分離されている場合、ホストは互いに 4 ホップ離れている、という。
マルチキャストアドレス	特定の方法でインタフェースのグループを特定する IP アドレス。マルチキャストアドレスに送信されるパケットは、グループにあるすべてのインタフェースに配信される。
ユニキャストアドレス	単一のインタフェースを指示する IP アドレス。
リダイレクト	特定の終点到達するために、ホストに対して最適な次のホップノードを、ルーターが通知すること。
リンクローカル使用アドレス	自動アドレス設定などのために、単一リンク上でアドレスを指定するために使用する。
ルーター通知	ルーターが、各種のリンクパラメータおよびインターネットパラメータと共に、その存在を定期的にあるいはルーター要請メッセージに応じて通知すること。
ルーター発見	ホストが、接続されているリンク上にあるルーターを特定すること。
ルーター要請	ホストがルーターに対し、次に予定されている時刻ではなく、ただちにルーター通知メッセージを送信するように要求すること。

ローカル使用アドレス ローカルの経路指定可能な範囲だけを対象とするユニキャストアドレス (サブネット内またはネットワーク内)。また、ローカルまたはグローバルな一意の範囲を対象とすることもできます。

索引

数字・記号

- 6to4relay コマンド
 - 構文, 68
 - トンネル設定作業, 57
 - 例, 69
- 6to4 アドレス
 - 書式, 95
 - ホストアドレス, 97
- 6to4 擬似インタフェースの設定, 38
- 6to4 通知, 39
- 6to4 トンネル
 - 6to4 リレールーターとの間の, 57
 - 概要, 33
 - 既知の問題, 101
 - トポロジ例, 93
 - パケットフロー, 97
 - パケットフロー、リレールーターを使用した, 99
- 6to4 の問題, 101
- 6to4 プレフィックス
 - /etc/inet/ndpd.conf 通知における, 40
 - 各部の説明, 96
- 6to4 リレールーター
 - 6to4 トンネルにおける, 68
 - セキュリティ問題, 100
 - トンネル設定作業, 57, 58
 - トンネルトポロジ, 99
- 6to4 ルーター
 - 構成例, 41
 - 設定作業, 38, 39, 40

A

- AAAA レコード, 42, 59, 81, 92
- ATM サポート, IPv6 over, 83

D

- DEFAULT_IP 変数, 49
- DNS
 - AAAA レコード, 42, 81, 92
 - IPv6 アドレスを追加, 42
 - IPv6 拡張機能, 81
 - PTR レコード, 60
 - 逆ゾーンファイル, 42
 - ゾーンファイル, 42

E

- /etc/default/inet_type ファイル, 49
 - DEFAULT_IP 値, 75, 76
- /etc/hostname6.interface ファイル, 36, 53
 - IPv6 トンネリング, 77
 - 複数のネットワークインタフェース, 64, 65
- /etc/hostname6.ip.6to4tun0 ファイル, 38
- /etc/hosts ファイル, 80
- /etc/inet/inetd.conf ファイル, 73
- /etc/inet/ipnodes ファイル, 41, 80, 81
- /etc/inet/ndpd.conf ファイル, 37, 56, 65, 71
 - 6to4 通知, 95

/etc/inet/ndpd.conf ファイル (続き)
6to4 ルーター通知, 39
キーワード, 71
/etc/nsswitch.conf ファイル, 81

G

getent コマンド, ipnodes オプション, 61
gethostbyname コマンド, 82
getipnodebyname コマンド, 82

H

hostname.interface ファイル
複数のネットワークインタフェース, 64, 65
hosts.byaddr マップ, 42, 81
hosts.byname マップ, 42, 81
hosts.org_dir テーブル, 41, 81

I

ifconfig コマンド, 44, 63, 76, 77
6to4 拡張機能, 39
-a オプション, 27
IPv6 拡張機能, 66
IPv6 の例, 67
アドレスの追加, 65
in.ndpd デーモン, 64, 65
オプション, 70
in.ripngd デーモン, IPv6 オプション, 73
inet6 オプション, route コマンド, 76
inetd デーモン, 73
ipnodes.byaddr マップ, 42
ipnodes.byname マップ, 42
ipnodes.org_dir テーブル, 41, 81
ipnodes オプション, getent コマンド, 61
IPsec, 33
IPv6 カプセル化セキュリティヘッダー, 33
IPv6 認証ヘッダー, 33
IPv4, IPv6 との相互運用性, 91
IPv4 互換 IPv6 アドレス, 18
IPv4 対応ホストアドレス, 16
IPv4 マップ IPv6 アドレス, 19
IPv6
ATM サポート, 83

IPv6 (続き)

DNS AAAA レコード, 59, 92
DNS 拡張機能, 81
DNS にアドレスを追加, 42
/etc/hostname6.interface ファイル, 53
/etc/inet/inetd.conf ファイル, 73
/etc/inet/ipnodes ファイル, 80, 81
/etc/inet/ndpd.conf ファイル, 56
getent コマンド, 61
ifconfig コマンド, 44
ifconfig コマンドの拡張機能, 66
in.ndpd デーモン, 70
in.ripngd デーモン, 73
IPv4 対応ホストアドレス, 16
IPv4 との相互運用性, 91
IPv4 との比較, 24
netstat コマンド, 45, 75
NFS と RPC のサポート, 83
NIS+ 拡張機能, 81
NIS+ テーブル, 92
NIS+ にアドレスを追加, 41
NIS 拡張機能, 81
NIS にアドレスを追加, 41
NIS マップ, 92
nslookup コマンド, 59, 60
ping コマンド, 50, 76
route コマンド, 76
snoop コマンド, 50, 75
traceroute コマンド, 51, 76
アドレス, 27
アドレス解決, 21
アドレス指定, 14
プレフィックスフォーマット割り当て, 15
アドレス自動設定, 21, 27, 70
アドレス領域, 15
アドレス割り当てを表示, 44
アプリケーションとの相互作用, 90
移行, 85
IPv4 互換アドレス, 88
移行シナリオ, 92
移行ツール, 85, 86
移行要求, 85
エニーキャストアドレス, 14, 19, 24
拡張ヘッダー, 14
拡張ヘッダーフィールド, 14
宛先オプション, 14
カプセル化, 14, 33

拡張ヘッダーフィールド (続き)
断片化, 14
認証, 14
ホップバイホップオプション, 14, 30, 31
ルーティング, 14
監視, 43
機能, 11
近傍検索, 21, 24, 25, 65
近傍不到達検出, 22, 25
近傍要請, 22
近傍要請と不到達, 23
サービス品質機能, 30
フローラベル, 30
サイトローカルアドレス, 26
サイトローカル使用アドレス, 16, 17
自動トンネル, 90
重複アドレス検出, 22
情報を NIS+ で表示, 61
情報を NIS で表示, 60
ステートフルアドレス自動設定, 26, 28
ステートレスアドレス自動設定, 26, 28, 29,
92
セキュリティの改善, 32
次のホップの決定, 21
デュアルスタック, 86, 91
動作, 70
トンネリング, 77, 86
トンネリング機構, 89
トンネルの設定, 52
認証ヘッダー, 21, 32
ネームサービス情報の表示, 58, 59
ネームサービスの設定, 87
ネットワークステータスを表示, 45
ネットワークトラフィックの監視, 50
ノード使用可能, 35
パケットのカプセル化, 63
パラメータ探索, 21
表示出力を制御, 49
プレフィックス探索, 21
プロトコル概要, 27
ヘッダー
トラフィッククラスフィールド, 13, 32
ヘッダーオプション, 14
ヘッダーと拡張機能, 12
ヘッダーフィールド
宛先アドレス, 13
ソースアドレス, 13
次のヘッダー, 13

ヘッダーフィールド (続き)
トラフィッククラス, 30, 31
フローラベル, 13
ペイロードの長さ, 13
ホップ制限, 13
ヘッダーフォーマット, 12
マルチキャストアドレス, 14, 16, 19, 25
マルチホームホストの探査, 50
モビリティサポート
ホームアドレス, 29
ユーティリティの拡張機能, 75
ユニキャストアドレス, 14, 16
リダイレクト, 22, 25
リンクローカルアドレス, 25, 26, 27, 29
リンクローカル使用アドレス, 16, 17
ルーター通知, 22, 23, 24, 25, 28
ルーターの設定, 37, 56
ルーター要請, 22, 28
ルータ発見, 70
ルーティング, 20
ルートのトレース, 51
ローカル使用アドレス, 16, 17
IPv6 アドレス
一意性, 29
組み込み IPv4 アドレス, 18
IPv6 内のサブネット, 6to4 トポロジにおけ
る, 94
IPv6 の既知の問題, 101
IPv6 の既知の問題点, 39
IPv6 パケットのカプセル化
6to4 シナリオ, 33
概要, 63
IPv6 への移行, 6to4 機構, 93
IPX アドレス, 16
IP アドレス, IPv6, 14

M
MTU, 24

N
netstat コマンド, 45, 76
-a オプション, 45
-f オプション, 45, 75
inet6 オプション, 45

netstat コマンド (続き)
 inet オプション, 45
 IPv6, 75
 -p オプション, 75
NFS のサポート, IPv6, 83
NIS
 IPv6 アドレスの追加, 41
 IPv6 拡張機能, 81
NIS+
 IPv6 アドレスの追加, 41
 IPv6 拡張機能, 81
NIS+ テーブル, IPv6, 92
nisaddent コマンド, 41
nisserver コマンド, 41
nissetup コマンド, 41
nistbladm コマンド, 41
NIS マップ, IPv6, 92
NSAP アドレス, 16
nslookup コマンド, 82
nslookup コマンド
 IPv6, 59, 60

P

ping コマンド
 -A オプション, 76
 -a オプション, 50
 -a コマンド, 76
 IPv6, 50, 76
PTR レコード, DNS, 60

R

route コマンド
 inet6 オプション, 76
 IPv6, 76
RPC のサポート, IPv6, 83

S

snoop コマンド
 ip6 オプション, 50
 ip6 プロトコルキーワード, 75
 IPv6, 75

T

TCP/IP 構成ファイル
 /etc/hostname6.interface, 64, 65
TCP/IP ネットワーク
 構成ファイル
 /etc/hostname6.interface, 64, 65
traceroute コマンド, 76
 -a オプション, 51, 76
 IPv6, 76
tun モジュール, 63, 77

V

/var/inet/ndpd_state.interface ファイル, 71

あ

宛先アドレスフィールド, IPv6 ヘッダー, 13
宛先オプションフィールド, IPv6 拡張ヘッダー, 14

アドレス

 リンクローカル使用, 17
 ローカル使用, 16
アドレス, IPv6
 6to4 の書式, 95
 IPv4 対応ホスト, 16
 IPv6, 27
 IPX, 16
 NSAP, 16
 アドレスの種類, 14
 エニーキャスト, 14
 サイトローカル使用, 16, 17
 集約グローバルユニキャスト, 16
 ニュートラル相互接続, 16
 マルチキャスト, 14, 16
 ユニキャスト, 14, 16
 集約グローバル, 16
 リンクローカル使用, 16
 ローカル使用, 17
アドレス解決, IPv6, 21
アドレス自動設定
 IPv6, 21, 27, 70
アドレス領域, IPv6, 15
アプリケーション層, ゲートウェイ, 102

い

- 移行シナリオ, IPv6, 92
- 移行シナリオ、IPv6、6to4 機構, 33
- インターネットプロトコルセキュリティ, 33
- インタフェース ID
 - IPv6 サイトローカルアドレス, 18
 - IPv6 リンクローカル使用アドレス, 18
- インタフェースアドレス
 - 6to4, 38
 - IPv6, 14

え

- エニーキャストアドレス, 57
 - IPv6, 19, 24
- エニーキャストグループ、6to4 リレールーター, 57

か

- 拡張ヘッダー, IPv6, 14
- カプセル化フィールド
 - IPv6 拡張ヘッダー, 14, 33
- 管理対象アドレス設定フラグ, ルーター通知, 28

き

- 逆ゾーンファイル, 42
- 境界ルーター、6to4 サイトにおける, 94
- 近傍検索
 - IPv6, 21, 24, 25
- 近傍検索デーモン, 65
- 近傍不到達検出
 - IPv6, 22, 25
- 近傍要請, IPv6, 22
- 近傍要請と不到達, 23

く

- グループ ID, マルチキャストアドレス, 20

さ

- サービス品質
 - IPv6, 30
 - IPv6 フローラベルフィールド, 30
- サイトローカルアドレス
 - IPv6, 26
 - インタフェース ID, 18
 - サブネット ID, 18
- サイトローカル使用アドレス, 16, 17
- サブネット ID, IPv6 サイトローカルアドレス, 18

し

- 自動アドレス設定フラグ, ルーター通知プレフィックスフィールド, 28
- 自動トンネル, IPv6, 90
- 次ホップ, 25
- 重複アドレス検出, IPv6, 22
- 重複アドレスの検出, アルゴリズム, 27
- 集約グローバルユニキャストアドレス, 16

す

- スコープの値, マルチキャストアドレス, 20
- ステートフルアドレス自動設定, 26, 27, 28
- ステートレスアドレス自動設定, 26, 27, 28
 - IPv6, 92

せ

- セキュリティ、IPv6
 - 6to4 リレールーターの問題, 100
 - IPv6 における改善, 32

そ

- ソースアドレスフィールド, IPv6 ヘッダー, 13
- ゾーンファイル, 42
- その他のステートフル設定フラグ, ルーター通知, 28

た

断片化フィールド, IPv6 拡張ヘッダー, 14

つ

通知, 6to4 ルーター, 96

次のヘッダーフィールド, IPv6 ヘッダー, 13

次のホップの決定, IPv6, 21

て

デーモン

in.ndpd, 70

in.ripngd, 73

inetd インターネットサービス, 73

IPv6, 70

デュアルスタック

IPv6, 86, 87, 91

と

トラフィッククラスフィールド

IPv6 ヘッダー, 13, 31, 32

トラフィックフィールド, IPv6 ヘッダー, 30

トンネリング, 86

IPv6, 77, 89

ルーターの設定, 56

トンネル

6to4 トンネルトポロジ, 93

6to4 トンネルのパケットフロー, 97

IPv6 設定, 52

トポロジ, 6to4 リレールーターに対する, 99

に

ニュートラル相互接続アドレス, 16

入力負荷均衡, 24

認証フィールド, IPv6 拡張ヘッダー, 14

認証ヘッダー

IPv6, 21, 32

ね

ネームサービス

IPv6 拡張機能, 79

IPv6 情報の表示, 58, 59

ネットワークインタフェース

複数のネットワークインタフェース

/etc/hostname6.interface ファイル, 64, 65

ネットワークマスク, 25

は

パケットフロー, IPv6

6to4 とネイティブ IPv6, 99

6to4 トンネルを介した, 97

同じフローに属する, 31

定義, 30

パラメータ探索, IPv6, 21

ふ

フォーマットプレフィックス, IPv6, 15

負荷均衡, 入力, 24

複数のネットワークインタフェース

/etc/hostname6.interface ファイル, 64, 65

プレフィックス

ルーター通知, 23, 24, 28

プレフィックス探索, IPv6, 21

プレフィックスフォーマット割り当て, IPv6 アドレス, 15

フロー, パケット, 30

フローラベルフィールド

IPv6 サービス品質, 30

IPv6 ヘッダー, 13

プロキシ通知, 24

へ

ペイロードの長さフィールド, IPv6 ヘッダー, 13

ヘッダーフィールド, IPv6, 13

ほ

ホストの設定, 6to4 ホストアドレス, 97
ホップ制限フィールド, IPv6 ヘッダー, 13
ホップバイホップオプションフィールド
IPv6 拡張ヘッダー, 14, 30, 31

ま

マルチキャストアドレス, 16
IPv6, 19, 25
グループ ID, 20
スコープの値, 20

め

メッセージ, ルーター通知, 23

も

モビリティサポート
IPv6, 29
ホームアドレス, 29

ゆ

ユニキャストアドレス, 16
集約グローバル, 16
フォーマットプレフィックス, 17

り

リダイレクト
IPv6, 22, 25
リレールーター、6to4 トンネル設定, 57, 58
リンク層アドレス, 24
リンクローカルアドレス
IPv6, 25, 26, 27, 29, 78
リンクローカル使用アドレス, 16, 17
インタフェース ID, 18

る

ルーター
6to4 トポロジにおける役割, 93
設定, 6to4, 38
パケットのフロー, 31
ルーター設定, IPv6, 37
ルーター通知
IPv6, 22, 23, 24, 25, 28
プレフィックス, 28
ルーター要請
IPv6, 22, 28
ルータ発見, IPv6, 70
ルーティング, IPv6, 20
ルーティングフィールド, IPv6 拡張ヘッダー
, 14
ルートのトレース, IPv6, 51

ろ

ローカル使用, 16
ローカル使用アドレス, 17

