Oracle® Solaris 11.1 ネットワークパフォーマンスの管理

Part No: E36607–02
2013 年 9 月

ORACLE®

Copyright © 1999, 2013, Oracle and/or its affiliates. All rights reserved.

このソフトウェアおよび関連ドキュメントの使用と開示は、ライセンス契約の制約条件に従うものとし、知的財産に関する法律により保護されています。ライセンス契約で明示的に許諾されている場合もしくは法律によって認められている場合を除き、形式、手段に関係なく、いかなる部分も使用、複写、複製、翻訳、放送、修正、ライセンス供与、送信、配布、発表、実行、公開または表示することはできません。このソフトウェアのリバース・エンジニアリング、逆アセンブル、逆コンパイルは互換性のために法律によって規定されている場合を除き、禁止されています。

ここに記載された情報は予告なしに変更される場合があります。また、誤りが無いことの保証はいたしかねます。誤りを見つけた場合は、オラクル社までご連絡ください。

このソフトウェアまたは関連ドキュメントを、米国政府機関もしくは米国政府機関に代わってこのソフトウェアまたは関連ドキュメントをライセンスされた者に提供する場合は、次の通知が適用されます。

U.S. GOVERNMENT END USERS:

Oracle programs, including any operating system, integrated software, any programs installed on the hardware, and/or documentation, delivered to U.S. Government end users are "commercial computer software" pursuant to the applicable Federal Acquisition Regulation and agency-specific supplemental regulations. As such, use, duplication, disclosure, modification, and adaptation of the programs, including any operating system, integrated software, any programs installed on the hardware, and/or documentation, shall be subject to license terms and license restrictions applicable to the programs. No other rights are granted to the U.S. Government.

このソフトウェアもしくはハードウェアは様々な情報管理アプリケーションでの一般的な使用のために開発されたものです。このソフトウェアもしくはハードウェアは、危険が伴うアプリケーション（人的傷害を発生させる可能性があるアプリケーションを含む）への用途を目的として開発されていません。このソフトウェアもしくはハードウェアを危険が伴うアプリケーションで使用する際、安全に使用するために、適切な安全装置、バックアップ、冗長性（redundancy）、その他の対策を講じることは使用者の責任となります。このソフトウェアもしくはハードウェアを危険が伴うアプリケーションで使用したことに起因して損害が発生しても、オラクル社およびその関連会社は一切の責任を負いかねます。

OracleおよびJavaはOracle Corporationおよびその関連企業の登録商標です。その他の名称は、それぞれの所有者の商標または登録商標です。

Intel、Intel Xeonは、Intel Corporationの商標または登録商標です。すべてのSPARCの商標はライセンスをもとに使用し、SPARC International, Inc.の商標または登録商標です。AMD、Opteron、AMDロゴ、AMD Opteronロゴは、Advanced Micro Devices, Inc.の商標または登録商標です。UNIXは、The Open Groupの登録商標です。

このソフトウェアまたはハードウェア、そしてドキュメントは、第三者のコンテンツ、製品、サービスへのアクセス、あるいはそれらに関する情報を提供することがあります。オラクル社およびその関連会社は、第三者のコンテンツ、製品、サービスに関して一切の責任を負わず、いかなる保証もいたしません。オラクル社およびその関連会社は、第三者のコンテンツ、製品、サービスへのアクセスまたは使用によって損失、費用、あるいは損害が発生しても一切の責任を負いかねます。

130924@25097

目次

はじめに

『Oracle Solaris 11.1 ネットワークパフォーマンスの管理』へようこそ。このドキュメントは、Oracle Solaris 11.1 ネットワークの確立に関するシリーズの一部で、Oracle Solaris ネットワークを構成するための基本的なトピックおよび手順について説明しています。このドキュメントの記述は、Oracle Solaris がインストール済みであることが前提です。さらに、ネットワークを構成できる状態であり、そのネットワークに必要なネットワークソフトウェアを構成できる状態である必要があります。

対象読者

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

Oracle サポートへのアクセス

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

表記上の規則

次の表では、このドキュメントで使用される表記上の規則について説明します。

表 P–1　表記上の規則

字体	説明	例
AaBbCc123	コマンド名、ファイル名、ディレクトリ名、画面上のコンピュータ出力、コード例を示します。	.login ファイルを編集します。 ls -a を使用してすべてのファイルを表示します。 machine_name% you have mail.

表 P–1 表記上の規則 *(続き)*

字体	説明	例
AaBbCc123	ユーザーが入力する文字を、画面上のコンピュータ出力と区別して示します。	`machine_name%` **`su`** `Password:`
aabbcc123	プレースホルダ: 実際に使用する特定の名前または値で置き換えます。	ファイルを削除するには、 `rm` *filename* と入力します。
AaBbCc123	書名、新しい単語、および強調する単語を示します。	『ユーザーズガイド』の第 6 章を参照してください。 キャッシュは、ローカルに格納されるコピーです。 ファイルを保存しないでください。 注: いくつかの強調された項目は、オンラインでは太字で表示されます。

コマンド例のシェルプロンプト

次の表に、Oracle Solaris OS に含まれるシェルの UNIX システムプロンプトおよびスーパーユーザーのプロンプトを示します。コマンド例のシェルプロンプトは、そのコマンドを標準ユーザーで実行すべきか特権ユーザーで実行すべきかを示します。

表 P–2 シェルプロンプト

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

◆ ◆ ◆ 第 1 章

1 ネットワークパフォーマンス管理の概要

この紹介の章では、ネットワークパフォーマンスの管理の概要を示します。この章では、このドキュメントのほかの部分で説明されている、ネットワークパフォーマンスの向上を可能にする機能についても紹介します。

このドキュメントで説明されている構成を実行する前に、システムをネットワークに接続するための基本的なネットワーク構成を完了しておく必要があります。基本的なネットワーク構成については、『Oracle Solaris 11.1 でのリアクティブネットワーク構成を使用したシステムの接続』および『Oracle Solaris 11.1 での固定ネットワーク構成を使用したシステムの接続』を参照してください。

Oracle Solaris 11 システムのネットワーク全般の概要については、『Oracle Solaris 11 ネットワーキングの紹介』を参照してください

ネットワークパフォーマンス向上のための機能

ネットワークインタフェースを構成することによってシステムをネットワークに接続します。ただし、Oracle Solaris 11 のほかの機能は、ネットワークの全体的なパフォーマンスを向上させます。ネットワークパフォーマンスの管理とは、これらの機能とテクノロジを使用して、システムでネットワークトラフィックを処理する方法を微調整することです。これらのテクノロジを使用して構成されたシステムはネットワークトラフィックをより適切に管理でき、それがネットワークの総合パフォーマンスの向上に寄与します。これらの機能は、ネットワーク操作のさまざまな領域に対応しています。ただし、次のような共通の利点が提供されます。

- ネットワーク接続 – リンクアグリゲーションや IPMP などの機能は、システムがネットワークに継続的にアクセスできるようにします。
- 効率 – 負荷分散などの機能は、システムがネットワーク処理の負荷を使用可能なリソース全体に分散できるようにして、効率を高めます。

- ネットワーク管理 – 仮想 LAN (VLAN) などの機能は管理を容易にします。ネットワークを向上させるその他の機能を使用することによっても、ネットワーク管理が簡単になります。
- コスト – Oracle Solaris 11 のこれらすべてのテクノロジは、高価な追加ハードウェアの購入を必要とせずに、ネットワークパフォーマンスを向上させます。

このドキュメントでは、システムでネットワークトラフィックをホストして処理する方法を向上させる機能について説明します。次のような例を考えます。

- データリンクとインタフェースを単一のユニットに構成できます。プールされたリソースをネットワーク処理専用で使用でき、結果としてネットワークのスループットが向上します。詳細は、第 2 章「リンクアグリゲーションの使用」または第 5 章「IPMP の概要」を参照してください。
- 管理を簡単にするために、ローカルエリアネットワークを仮想サブネットワークに分割できます。詳細は、第 3 章「VLAN の操作」を参照してください。
- システムが最短の接続ルートを使用してパケットを宛先に送信するようにすることができます。詳細は、第 4 章「ブリッジネットワークの管理 (タスク)」を参照してください。
- さまざまなシステムや構成を含むネットワークでは、ネットワーク上のすべてのピアが相互に構成を通知するメカニズムを有効にすることができます。そのため、両方のピアがサポートしているネゴシエート済みの構成セットに基づいて、ネットワークパケットを交換できます。ピアのネゴシエーションは自動です。したがって、手動構成は必要ありません。詳細は、第 7 章「LLDP によるネットワーク接続情報の交換」を参照してください。

このドキュメントで説明されているさまざまなテクノロジは、具体的な状況に応じて使用します。また、ハードウェア構成によっては、特定のタイプの機能を使用することが必要になる場合があります。たとえば、同じサブネットに属するように構成された複数のインタフェースがシステムに存在する場合は、IP マルチパス (IPMP) テクノロジを使用する必要があります。したがって、このドキュメントの構成手順をすべて完了する必要があるわけではありません。代わりに、ネットワークの要件に対応するテクノロジを選択して配備してください。

また、ネットワークの向上に寄与するほかのネットワーク関連のシステム構成について、Oracle Solaris 11.1 のドキュメントライブラリも参照してください。たとえば、仮想ネットワーク、ゾーン、およびリソース管理を使用すると、複数のネットワークを 1 つにまとめた形でパフォーマンスを提供でき、利用可能なネットワークリソースを最大限に使用できます。詳細は、『Oracle Solaris 11.1 での仮想ネットワークの使用』および『Oracle Solaris 11.1 の管理: Oracle Solaris ゾーン、Oracle Solaris 10 ゾーン、およびリソース管理』を参照してください。

◆ ◆ ◆ 第 2 章

2 リンクアグリゲーションの使用

リンクアグリゲーションを使用すると、複数のデータリンクのリソースをプールして、単一のユニットとして管理できます。複数のデータリンクのリソースを組み合わせ、それらをシステムのネットワーク操作に専用で利用することにより、システムのパフォーマンスが大幅に向上します。この章では、リンクアグリゲーションを構成および保守するために使用する手順について説明します。

この章で扱う内容は、次のとおりです。

リンクアグリゲーションの概要

トランキングとも呼ばれるリンクアグリゲーションは、ネットワークトラフィックのスループットを向上させるために、システム上で単一の論理的なユニットとして構成された複数のインタフェースから成ります。次の図は、システム上に構成されたリンクアグリゲーションの例を示しています。

図 2–1 リンクアグリゲーション構成

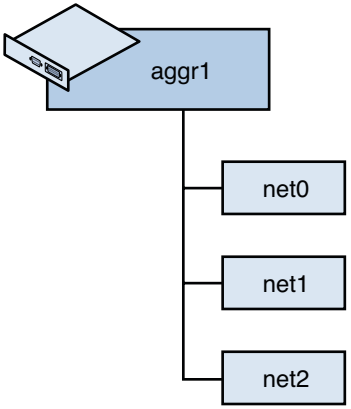


図 2–1 は、ベースとなる 3 つのデータリンク net0 から net2 で構成されたアグリゲーション aggr1 を示しています。これらのデータリンクは、このアグリゲーションを介してシステムを通過するトラフィック専用で利用されます。ベースとなるリンクは外部アプリケーションには隠されています。代わりに、データリンク aggr1 がアクセス可能になります。

リンクアグリゲーションには次の機能があります:

- 帯域幅の増加 – 複数のリンクの伝送容量が 1 つの論理的なリンクに統合されます。
- 自動フェイルオーバーおよびフェイルバック – リンクベースの障害検出をサポートすることにより、障害が発生したリンクのトラフィックがアグリゲーション内のほかの正常なリンクにフェイルオーバーされます。
- 管理の向上 – ベースとなるすべてのリンクが単一のユニットとして管理されます。
- ネットワークアドレスプールのアドレスの節約 – アグリゲーション全体に 1 つの IP アドレスを割り当てることができます。
- リンク保護 – アグリゲーションを通過するパケットのリンク保護を有効にするデータリンクプロパティーを構成できます。
- リソース管理 – ネットワークリソースに関するデータリンクプロパティーおよびフロー定義を使用して、アプリケーションでのネットワークリソースの使用を制御できます。リソース管理の詳細については、『Oracle Solaris 11.1 での仮想ネットワークの使用』の第 3 章「Oracle Solaris でのネットワークリソースの管理」を参照してください。

注 – リンクアグリゲーションは IP マルチパス (IPMP) と同様の機能を実行して、ネットワークのパフォーマンスと可用性を向上させます。これら 2 つのテクノロジの比較については、付録 B 「リンクアグリゲーションと IPMP: 機能比較」を参照してください。

Oracle Solaris は 2 種類のリンクアグリゲーションをサポートしています。

- トランクアグリゲーション
- データリンクマルチパス (DLMP) アグリゲーション

これら 2 種類のリンクアグリゲーションの相違点の概要については、付録 A 「リンクアグリゲーションの種類: 機能比較」を参照してください。

次のセクションでは、各タイプのリンクアグリゲーションについて詳細に説明します。

トランクアグリゲーション

トランクアグリゲーションは、トラフィック負荷の異なるさまざまなネットワークに役立ちます。たとえば、ネットワーク内のあるシステムが、分散された多くのトラフィックを処理するアプリケーションを実行している場合、トランクアグリゲーションをそのアプリケーションのトラフィック専用で使用すると、より大きい帯域幅を利用できます。IP アドレス空間が制限されていながら大容量の帯域幅が必要なサイトの場合、大規模なインタフェースのアグリゲーションでも 1 つの IP アドレスのみで済みます。内部インタフェースの存在を隠す必要があるサイトの場合、アグリゲーションの IP アドレスによって、内部インタフェースを外部アプリケーションから隠します。

Oracle Solaris では、アグリゲーションを作成すると、トランクアグリゲーションがデフォルトで構成されます。通常、リンクアグリゲーションが構成されているシステムは、ほかのシステムへの接続に外部スイッチも使用します。次の図を参照してください。

図2–2 スイッチを使用するリンクアグリゲーション

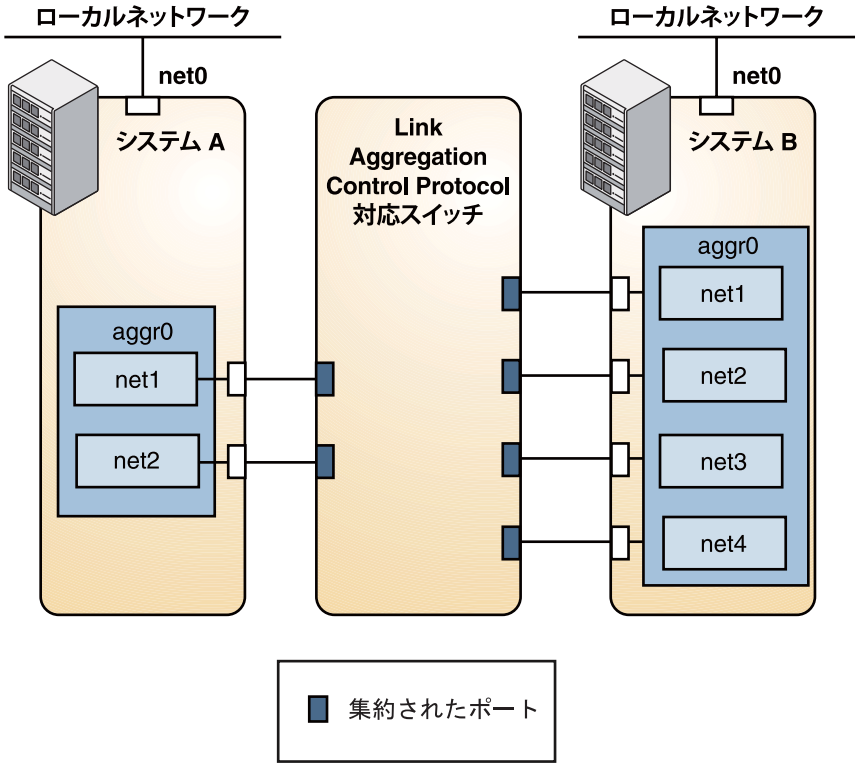


図 2–2 は、それぞれにアグリゲーションが構成された 2 つのシステムを含むローカルネットワークを示しています。2 つのシステムはスイッチによって接続されており、このスイッチには Link Aggregation Control Protocol (LACP) が構成されています。

システム A は、net1 と net2 の 2 つのインタフェースで構成されるアグリゲーションを使用しています。これらのインタフェースは、集約されたポートを介してスイッチに接続されています。システム B は、net1 から net4 の 4 つのインタフェースのアグリゲーションを使用しています。これらのインタフェースもスイッチの集約されたポートに接続されています。

このリンクアグリゲーショントポロジでは、スイッチがアグリゲーションテクノロジをサポートしている必要があります。したがって、そのスイッチポートは、システムからのトラフィックを管理するように構成されている必要があります。

トランクアグリゲーションでは、バックツーバック構成もサポートされます。次の図に示すように、スイッチを使用する代わりに、2 つのシステムが相互に直接接続され、並列アグリゲーションを実行します。

図 2–3 バックツーバックリンクアグリゲーション構成

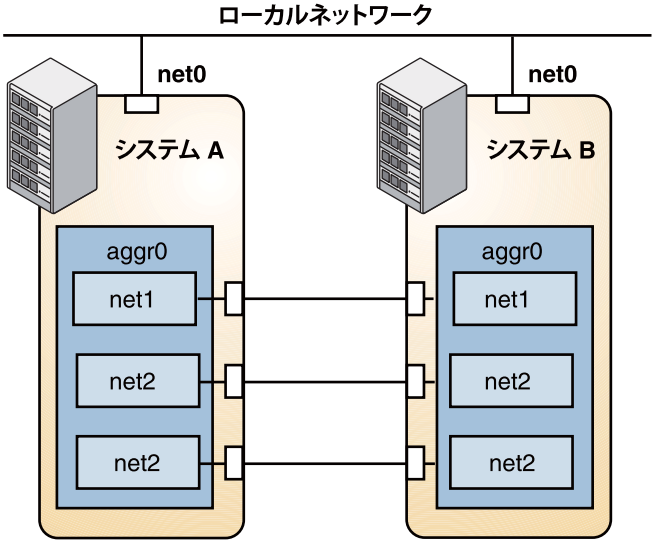


図 2–3 では、システム A のリンクアグリゲーション `aggr0` とシステム B のリンクアグリゲーション `aggr0` が、それぞれのベースとなるデータリンク間の対応するリンクを使用して直接接続されています。この方法では、システム A とシステム B は、冗長性と高可用性を提供し、さらに両方のシステム間での高速通信を提供します。各システムではさらに、ローカルネットワーク内のトラフィックフロー用の `net0` も構成されています。

バックツーバックリンクアグリゲーションのもっとも一般的な用途は、ミラー化されたデータベースサーバーの構成です。両方のサーバーを同時に更新する必要があるため、大きな帯域幅、高速のトラフィックフロー、および信頼性が必要になります。バックツーバックリンクアグリゲーションのもっとも一般的な使用場所としてデータセンターがあります。

注 – バックツーバック構成は DLMP アグリゲーションではサポートされていません。

次のセクションでは、トランクアグリゲーションに固有のほかの機能について説明します。DLMP アグリゲーションを作成する場合は、これらの機能を構成しないでください。

ポリシーと負荷分散

トランクアグリゲーションを使用する予定の場合は、送信トラフィック用のポリシーを定義することを検討してください。このポリシーでは、使用可能なアグリ

ゲーションのリンク全体にパケットを分散する方法を指定し、負荷分散を確立することができます。次に、使用可能な層指定子とアグリゲーションポリシーに対するそれらの意味について説明します。

- *L2* – 各パケットの MAC (L2) ヘッダーをハッシュすることで送信リンクを決定します
- *L3* – 各パケットの IP (L3) ヘッダーをハッシュすることで送信リンクを決定します
- *L4* – 各パケットの TCP、UDP、またはほかの ULP (L4) ヘッダーをハッシュすることで送信リンクを決定します

これらのポリシーを任意に組み合わせて使用することもできます。デフォルトのポリシーは L4 です。

アグリゲーションの **LACP** モードとスイッチ

トランクアグリゲーションの設定にスイッチが含まれている場合は、スイッチが LACP をサポートするかどうかに注意する必要があります。スイッチが LACP をサポートしている場合は、スイッチとアグリゲーションの LACP を構成する必要があります。アグリゲーションの LACP は、3 つの値のいずれかに設定できます。

- `off` – アグリゲーションのデフォルトのモード。「*LACPDU*」と呼ばれる LACP パケットは生成されません。
- `active` – システムは、ユーザーが指定可能な間隔で LACPDU を定期的に生成します。
- `passive` – システムは、スイッチから LACPDU を受け取った場合のみ LACPDU を生成します。アグリゲーションとスイッチの両方が受動モードで構成されている場合、それらの間で LACPDU を交換することはできません。

データリンクマルチパスアグリゲーション

通常、トランクアグリゲーションはネットワーク設定の要件を十分に満たします。ただし、トランクアグリゲーションは 1 つのスイッチでしか機能しません。そのため、スイッチがシステムのアグリゲーションの単一障害点になります。複数のスイッチにまたがるアグリゲーションを可能にする過去のソリューションには、それ独自の欠点があります。

- スイッチに実装されているソリューションはベンダー固有のもので、標準化されていません。ベンダーの異なる複数のスイッチを使用する場合、あるベンダーのソリューションがほかのベンダーの製品に適用されない可能性があります。
- リンクアグリゲーションを IP マルチパス (IPMP) と組み合わせることは、特に大域ゾーンと非大域ゾーンが関与するネットワーク仮想化の状況では、非常に複雑になります。たとえば、多数のシステム、ゾーン、NIC、仮想 NIC (VNIC)、および IPMP グループを含むシナリオなど、構成を拡大するほど複雑さが増しま

す。このソリューションでは、すべてのシステムの大域ゾーンと各非大域ゾーンに対して構成を実行することも必要になります。

- リンクアグリゲーションと IPMP の組み合わせを実装しても、その構成では、リンク層だけで動作することから得られるほかの利点、たとえば、リンク保護、ユーザー定義のフロー、帯域幅などのリンクプロパティーをカスタマイズする機能などを享受できません。

DLMP アグリゲーションはこのような欠点を解消します。次の図は、DLMP アグリゲーションの動作を示しています。

図 2–4 DLMP アグリゲーション

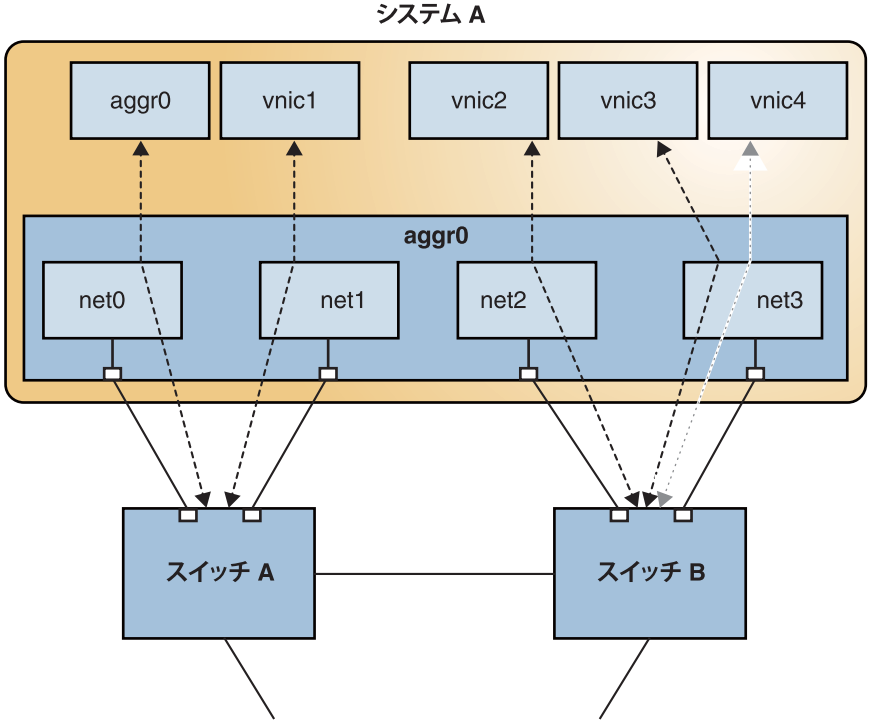


図 2–4 は、リンクアグリゲーション `aggr0` を含むシステム A を示しています。このアグリゲーションは、ベースとなる 4 つのリンク `net0` から `net3` で構成されています。プライマリインタフェースである `aggr0` に加え、アグリゲーション上には VNIC `vnic1` から `vnic4` も構成されています。アグリゲーションはスイッチ A およびスイッチ B に接続されており、これらのスイッチはより広いネットワーク内のほかの宛先システムに接続されています。

トランクアグリゲーションでは、アグリゲーション上に構成されている各データリンクが各ポートに関連付けられます。DLMP アグリゲーションでは、アグリゲーションに構成されている任意のデータリンクと、そのアグリゲーション上のプライマリインタフェースおよび VNIC が、1 つのポートに関連付けられます。

VNIC の数がベースとなるリンクの数を超えている場合、1 つのポートに複数のデータリンクが関連付けられます。たとえば、図 2–4 では、`vnic4` と `vnic3` が 1 つのポートを共有しています。

同様に、アグリゲーションのポートに障害が発生すると、そのポートを使用しているすべてのデータリンクがほかのポートに分散されます。たとえば、`net0` に障害が発生した場合、`aggr0` はポートをほかのデータリンクの 1 つと共有します。アグリゲーションポート間の分散は、ユーザーに対しては透過的に、また、アグリゲーションに接続されている外部スイッチとは無関係に行われます。

スイッチに障害が発生した場合、アグリゲーションはほかのスイッチを使用して、そのデータリンクへの接続を引き続き提供します。したがって、DLMP アグリゲーションでは複数のスイッチを使用できます。

まとめると、DLMP アグリゲーションは次の機能をサポートします。

- アグリゲーションは複数のスイッチにまたがることができます。
- スイッチの構成は不要で、スイッチに対して構成を実行してもいけません。
- そのタイプでサポートされているオプションだけを使用するかぎり、`dladm modify-aggr` コマンドを使用してトランクアグリゲーションと DLMP アグリゲーションを切り替えることができます。

注 – トランクアグリゲーションから DLMP アグリゲーションに切り替える場合は、トランクアグリゲーション用に以前作成したスイッチ構成を削除する必要があります。

リンクアグリゲーションの要件

リンクアグリゲーションの構成には次のような要件があります。

- アグリゲーションに構成するデータリンク上には、IP インタフェースが構成されていてはいけません。
- アグリゲーション内のすべてのデータリンクは、同じ速度かつ全二重モードで実行される必要があります。
- DLMP アグリゲーションの場合、アグリゲーションをほかのシステムのポートに接続するために、少なくとも 1 つのスイッチが必要です。DLMP アグリゲーションを構成する場合、バックツーバック設定は使用できません。

- SPARC ベースのシステムでは、各データリンクに一意の MAC アドレスが必要です。手順については、『Oracle Solaris 11.1 での固定ネットワーク構成を使用したシステムの接続』の「各インタフェースの MAC アドレスが一意であることを確認する方法」を参照してください。

ポートをアグリゲーションに接続したり、アグリゲーションから切り離したりするには、IEEE 802.3ad Link Aggregation Standard で定義されているリンク状態通知がデバイスでサポートされている必要があります。リンク状態通知をサポートしていないデバイスは、`dladm create-aggr` コマンドの `-f` オプションを使用してのみ集約できます。このようなデバイスの場合、リンク状態は常に `UP` として報告されます。`-f` オプションの使用については、23 ページの「リンクアグリゲーションを作成する方法」を参照してください。

リンクアグリゲーションの管理

このセクションでは、リンクアグリゲーションの構成および管理のさまざまな手順について説明します。トランクアグリゲーションの構成と DLMP アグリゲーションの構成で、手順の一部は共通です。一方のタイプに固有の手順には、その旨が明記されています。

▼ リンクアグリゲーションを作成する方法

始める前に

注 – リンクアグリゲーションは、同一の速度で稼働する全二重のポイントツーポイントリンク上でのみ機能します。アグリゲーション内のインタフェースがこの要件を満たしていることを確認してください。

アグリゲーショントポロジ内でスイッチを使用している場合に、トランクアグリゲーションを作成するときは、スイッチ上で次の操作を行なったことを確認してください:

- アグリゲーションとして使用されるようにポートを構成します。
- スイッチが LACP をサポートしている場合は、LACP をアクティブモードまたは受動モードで構成します。

これらの前提条件は DLMP アグリゲーションには適用されません。

1 管理者になります。

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

2 集約するデータリンクを決定するために、データリンク情報を表示します。

```
# dladm show-link
```

3 アグリゲーションに構成しようとしているデータリンクが、どのアプリケーションでも開かれていないことを確認します。

たとえば、データリンク上に IP インタフェースが作成されている場合は、まずその IP インタフェースを削除します。

a. リンクがいずれかのアプリケーションで使用されているかどうかを判定するには、**ipadm show-if** コマンドの出力を調べます。

```
# ipadm show-if
IFNAME       CLASS        STATE     ACTIVE     OVER
lo0          loopback     ok        yes        --
net0         ip           ok        no         --
```

この出力は、データリンク `net0` 上に IP インタフェースが存在することを示しています。

b. IP インタフェースを削除するには、次のコマンドを入力します。

```
# ipadm delete-ip interface
```

ここで、*interface* はリンク上の IP インタフェースを指定します。

4 次のいずれかのコマンドを発行して、リンクアグリゲーションを作成します。

- トランクアグリゲーションを作成するには、次のコマンドを発行します。

```
# dladm create-aggr [-f] [-P policy]  [-L lacpmode] \
 [-T time] [-u address] -l link1 -l link2 [...] aggr
```

`-f`	アグリゲーションを強制的に作成します。このオプションは、リンク状態通知をサポートしていないデバイスを集約しようとする場合に使用します。
`-P` *policy*	アグリゲーションの負荷分散ポリシーを指定します。
`-L` *lacpmode*	LACP を使用する場合は、そのモードを指定します。`off`、`active`、`passive` のいずれかです。20 ページの「アグリゲーションの LACP モードとスイッチ」を参照してください。
`-T` *time*	LACP の時間を指定します。
`-u` *address*	アグリゲーションの固定ユニキャストアドレスを指定します。
`-l` *linkn*	集約するデータリンクを指定します。

aggr アグリゲーションの名前を指定します。カスタマイズした任意の名前を使用できます。名前の割り当ての規則については、『Oracle Solaris 11 ネットワーキングの紹介』の「有効なリンク名のための規則」を参照してください。

- DLMP アグリゲーションを作成するには、次のコマンドを発行します。

  ```
  # dladm create-aggr -m haonly -l link1 -l link2 [...] aggr
  ```

  `-l` *linkn* 集約するデータリンクを指定します。

  aggr アグリゲーションの名前を指定します。

5. (オプション) 作成したアグリゲーションのステータスを確認します。

   ```
   # dladm show-aggr
   ```

   アグリゲーションの状態は UP である必要があります。

6. IP インタフェースや VNIC の作成など、アグリゲーションの追加の構成を実行します。

例 2–1 トランクアグリゲーションの作成

この例は、ベースとなる 2 つのデータリンク net0 および net1 を含むリンクアグリゲーションを作成するコマンドを示しています。また、アグリゲーションは LACP パケットを送信するように構成されます。この例ではまず、ベースとなるデータリンク上の既存の IP インタフェースを削除します。

```
# ipadm show-if
IFNAME       CLASS        STATE     ACTIVE     OVER
lo0          loopback     ok        yes        --
net0         ip           ok        no         --
net1         ip           ok        no         --

# ipadm delete-ip net0
# ipadm delete-ip net1
# dladm create-aggr -L active -l net0 -l net1 aggr0

# dladm show-aggr
LINK    MODE      POLICY   ADDRPOLICY     LACPACTIVITY   LACPTIMER
aggr0   standard  L4       auto           on             short
```

例 2–2 DLMP アグリゲーションの作成

この例は、DLMP アグリゲーションを作成する方法を示しています。アグリゲーションにはベースとなる 3 つのデータリンクがあります。

```
# dladm create-aggr -m haonly -l net0 -l net1 -l net2 aggr0
# dladm show-link
LINK      CLASS     MTU     STATE   BRIDGE     OVER
```

```
net0      phys      1500    up       --         ----
net1      phys      1500    up       --         ----
net2      phys      1500    up       --         ----
aggr0     aggr      1500    up       --         net0, net1, net2

# dladm show-aggr
LINK    MODE       POLICY  ADDRPOLICY      LACPACTIVITY   LACPTIMER
aggr0   haonly     --      ----            ---            ----
```

▼ リンクアグリゲーションのタイプを切り替える方法

トランクアグリゲーションと DLMP アグリゲーションの間でアグリゲーションのタイプを切り替えるには、`dladm modify-aggr` コマンドを使用してアグリゲーションのモードを変更します。アグリゲーションのタイプを切り替えると、構成全体が変更されることに注意してください。したがって、この手順では、ほかのリンクアグリゲーションプロパティーを単に変更する場合よりも総合的にアグリゲーションが影響を受けます。

1. 管理者になります。

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

2. リンクアグリゲーションの現在のタイプを判定します。

   ```
   # dladm show-aggr
   ```

   出力の `MODE` フィールドは、アグリゲーションの現在のタイプを示します。`MODE` の値は、トランクアグリゲーションの場合は `standard`、DLMP アグリゲーションの場合は `haonly` です。

3. アグリゲーションを **DLMP** アグリゲーションに切り替えます。

   ```
   # dladm modify-aggr -m mode aggr
   ```

   ここで、*mode* は、トランクアグリゲーションに切り替える場合は `standard`、DLMP アグリゲーションに切り替える場合は `haonly` です。

4. 新しいタイプのリンクアグリゲーションの要件に従って、スイッチの調整を実行します。

5. (オプション) リンクアグリゲーションの構成を表示します。

   ```
   # dladm show-aggr
   ```

例 2–3 トランクアグリゲーションから DLMP アグリゲーションへの切り替え

この例は、アグリゲーションをトランクアグリゲーションから DLMP アグリゲーションに変更する方法を示しています。

```
# dladm show-aggr
LINK    MODE       POLICY  ADDRPOLICY      LACPACTIVITY   LACPTIMER
aggr0   standard   L2      auto            active         short

# dladm modify-aggr -m haonly aggr0
# dladm show-aggr
LINK    MODE       POLICY  ADDRPOLICY      LACPACTIVITY   LACPTIMER
aggr0   haonly     --      ----            -----          ----
```

次に、スイッチ側で、それまでトランクアグリゲーションに適用されていたスイッチ構成を削除します。

▼ トランクアグリゲーションを変更する方法

この手順では、トランクアグリゲーションのみの、選択した属性を変更する方法を示します。これらの属性は DLMP アグリゲーションではサポートされていません。

1. 管理者になります。

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

2. アグリゲーションのポリシーを変更します。

   # dladm modify-aggr -P *policy aggr*

   policy 19 ページの「ポリシーと負荷分散」で説明されているように 1 つ以上のポリシー L2、L3、および L4 を表します。

   aggr ポリシーを変更するアグリゲーションを指定します。

3. アグリゲーションの **LACP** モードを変更します。

   # dladm modify-aggr -L *lacpmode* **-T** *time aggr*

   -L *lacpmode* アグリゲーションが実行される LACP モードを示します。値は、`active`、`passive`、および `off` です。

   -T *time* LACP タイマー値を示します。値は、`short` または `long` です。

   aggr ポリシーを変更するアグリゲーションを指定します。

例 2–4 トランクアグリゲーションの変更

この例は、リンクアグリゲーション aggr0 のポリシーを L2 に変更したあと、active LACP モードをオンに設定する方法を示しています。

```
# dladm modify-aggr -P L2 aggr0
# dladm modify-aggr -L active -T short aggr0
# dladm show-aggr
LINK    MODE       POLICY  ADDRPOLICY      LACPACTIVITY   LACPTIMER
aggr0   standard   L2      auto            active         short
```

▼ アグリゲーションにリンクを追加する方法

1. 管理者になります。

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

2. 追加するリンクに、そのリンク上で **plumb** されている **IP** インタフェースが含まれていないことを確認します。

   # ipadm delete-ip *interface*

   ここで、*interface* は、データリンク上に構成されている IP インタフェースです。

3. アグリゲーションにリンクを追加します。

   # dladm add-aggr -l *link* **[-l** *link***] [...]** *aggr*

   ここで、*link* はアグリゲーションに追加するデータリンクを表し、*aggr* はアグリゲーションの名前です。

4. **(**トランクアグリゲーションの場合のみ**)** リンクアグリゲーションに **LACP** が構成されていない場合は、追加のデータリンクを収容できるようにスイッチを再構成します。

   スイッチに対して何らかの再構成タスクを実行するには、スイッチのドキュメントを参照してください。

例 2–5 アグリゲーションへのリンクの追加

この例は、アグリゲーション aggr0 にリンクを追加する方法を示しています。

```
# dladm show-link
LINK     CLASS     MTU     STATE     BRODGE   OVER
net0     phys      1500    up        --       ----
net1     phys      1500    up        --       ----
aggr0    aggr      1500    up        --       net0, net1
net3     phys      1500    up        --       ----
```

```
# ipadm delete-ip net3
# dladm add-aggr -l net3 aggr0
# dladm show-link
LINK      CLASS     MTU     STATE     BRIDGE   OVER
net0      phys      1500    up        --       ----
net1      phys      1500    up        --       ----
aggr0     aggr      1500    up        --       net0, net1, net3
net3      phys      1500    up        --       ----
```

▼ アグリゲーションからリンクを削除する方法

1. 管理者になります。

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

2. アグリゲーションからリンクを削除します。

   ```
   # dladm remove-aggr -l link aggr
   ```

例 2–6 アグリゲーションからのリンクの削除

この例は、アグリゲーション `aggr0` からリンクを削除する方法を示しています。

```
dladm show-link
LINK      CLASS     MTU     STATE     OVER
net0      phys      1500    up        --       ----
net1      phys      1500    up        --       ----
aggr0     aggr      1500    up        --       net0, net1, net3
net3      phys      1500    up        --       ----

# dladm remove-aggr -l net3 aggr0
# dladm show-link
LINK      CLASS     MTU     STATE     BRIDGE   OVER
net0      phys      1500    up        --       ----
net1      phys      1500    up        --       ----
aggr0     aggr      1500    up        --       net0, net1
net3      phys      1500    unknown   --       ----
```

▼ リンクアグリゲーションを削除する方法

1. 管理者になります。

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

2 リンクアグリゲーション上に構成されている IP インタフェースを削除します。

```
# ipadm delete-ip IP-aggr
```

ここで、*IP-aggr* はリンクアグリゲーション上の IP インタフェースです。

3 リンクアグリゲーションを削除します。

```
# dladm delete-aggr aggr
```

例 2–7 リンクアグリゲーションの削除

この例は、アグリゲーション `aggr0` を削除する方法を示しています。この削除は永続します。

```
# ipadm delete-ip aggr0
# dladm delete-aggr aggr0
```

◆ ◆ ◆ 第 3 章

3 VLANの操作

仮想LANによって、物理的なネットワーク環境を追加することなく、ネットワークをサブネットワークに分割できます。したがって、サブネットワークは仮想であり、同じ物理ネットワークリソースを使用します。小さいグループのほうが保守しやすいため、VLANによってネットワーク管理が容易になります。VLANの機能と利点については、次のセクションで詳細に説明します。

この章では、仮想ローカルエリアネットワーク(VLAN)を構成および保守するための手順について説明します。内容は次のとおりです。

VLANの配備: 概要

仮想ローカルエリアネットワーク *(VLAN)* は、ローカルエリアネットワークをプロトコルスタックのデータリンク層で分割したものです。スイッチテクノロジを使用するローカルエリアネットワークのVLANを作成できます。ユーザーのグループをVLANに割り当てることで、ローカルネットワーク全体のネットワーク管理とセキュリティーを改善できます。さらに、同じシステム上のインタフェースを異なるVLANに割り当てることもできます。

次のセクションでは、VLANの簡単な概要を示します。

どのような場合にVLANを使用するか

次のことを行う必要がある場合は、ローカルネットワークをVLANに分割することを検討してください:

- 作業グループの論理的な分割を作成する。

たとえば、ある建物の 1 つの階に置かれたすべてのホストが 1 つのスイッチベースのローカルネットワークに接続されているとします。この階の各作業グループ用に個別の VLAN を作成できます。

- 作業グループに異なるセキュリティーポリシーを適用する。

  たとえば、財務部門と情報技術部門のセキュリティー要件はかなり異なります。両方の部門のシステムが同じローカルネットワークを共有している場合、各部門用の個別の VLAN を作成できます。そのあとで、適切なセキュリティーポリシーを VLAN ごとに適用できます。

- 作業グループを管理可能なブロードキャストドメインに分割する。

  VLAN を使用すると、ブロードキャストドメインのサイズが小さくなり、ネットワークの効率が向上します。

VLAN およびカスタマイズされた名前

VLAN では、汎用またはカスタマイズされた名前を使用することに明らかな利点があります。以前のリリースでは、VLAN は物理接続点 (PPA) で識別されていたため、ハードウェアベースのデータリンク名と VLAN ID を組み合わせる必要がありました。Oracle Solaris 11 では、VLAN の識別のためにより意味のある名前を選択できます。名前は、『Oracle Solaris 11 ネットワーキングの紹介』の「有効なリンク名のための規則」で説明されているデータリンクの命名規則に適合している必要があります。したがって、VLAN には `sales0` や `marketing1` などの名前を割り当てることができます。

VLAN 名は VLAN ID と連携します。ローカルエリアネットワーク内の各 VLAN は、VLAN タグとも呼ばれる VLAN ID によって識別されます。VLAN ID は VLAN の構成時に割り当てられます。VLAN をサポートするようにスイッチを構成する場合は、各ポートに VLAN ID を割り当てる必要があります。ポートの VLAN ID は、そのポートに接続するインタフェースに割り当てられた VLAN ID と同じでなければなりません。

カスタマイズされた名前と VLAN ID の使用については、次のセクションで説明します。

VLAN トポロジ

スイッチ LAN テクノロジを使用すると、ローカルネットワーク上のシステムを VLAN に編成できます。ローカルネットワークを VLAN に分割する前に、VLAN テクノロジをサポートするスイッチを入手する必要があります。VLAN トポロジの設計に応じて、スイッチ上のすべてのポートで単一の VLAN を処理するか、複数の VLAN を処理するように構成できます。スイッチのポートを構成する手順はスイッチの製造元によって異なります。

次の図は、3 つの VLAN に分割されているローカルエリアネットワークを示しています。

図 3–1　3 つの VLAN を含むローカルエリアネットワーク

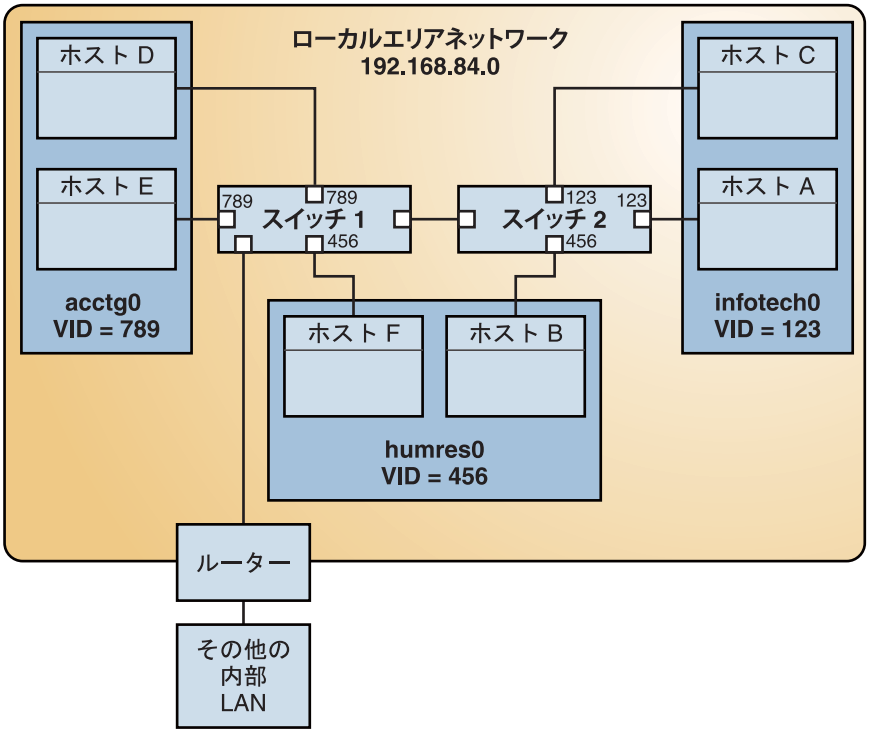


図 3–1 で、LAN のサブネットアドレスは 192.168.84.0 です。この LAN は、3 つの作業グループに対応する 3 つの VLAN に分割されています。

- acctg0 (VLAN ID 789) – 経理グループ。このグループはホスト D とホスト E を所有しています。
- humres0 (VLAN ID 456) – 人事グループ。このグループはホスト B とホスト F を所有しています。
- infotech0 (VLAN ID 123) – 情報技術グループ。このグループはホスト A とホスト C を所有しています。

図 3–1 の変形を図 3–2 に示します。ここではスイッチが 1 つだけ使用され、さまざまな VLAN に属している複数のホストがその単一のスイッチに接続します。

図 3–2 VLAN を使用するネットワークのスイッチの構成

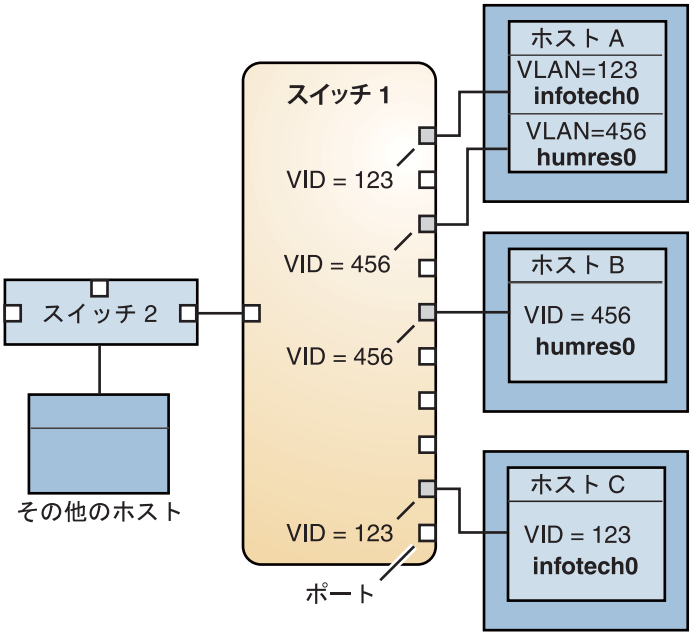


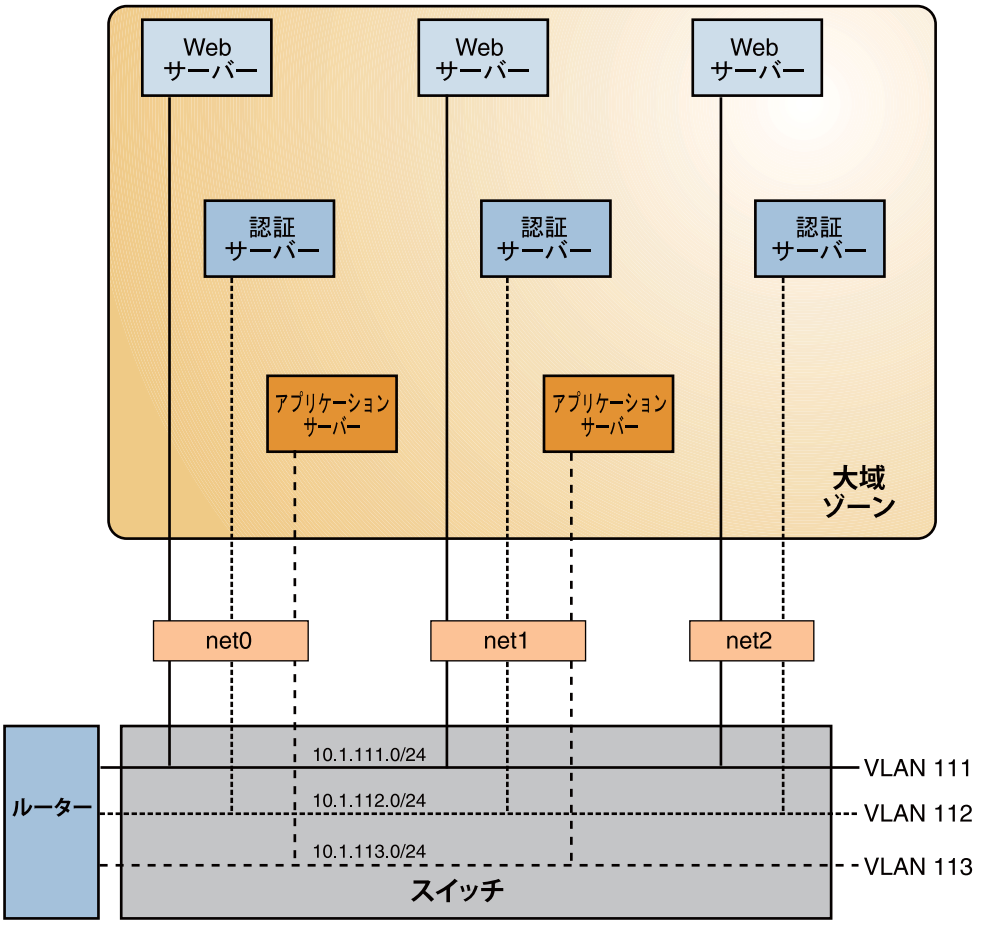
図 3–2 で、ホスト A およびホスト C は、VLAN ID 123 を持つ情報技術 VLAN に属しています。したがって、ホスト A のインタフェースの 1 つに VLAN ID 123 が構成されています。このインタフェースはスイッチ 1 のポート 1 に接続しており、そのポートにも VLAN ID 123 が構成されています。ホスト B は、VLAN ID 456 を持つ人事 VLAN のメンバーです。ホスト B のインタフェースはスイッチ 1 のポート 5 に接続しており、そのポートには VLAN ID 456 が構成されています。最後に、ホスト C のインタフェースには VLAN ID 123 が構成されています。このインタフェースはスイッチ 1 のポート 9 に接続しています。ポート 9 にも VLAN ID 123 が構成されています。

図 3–2 は、1 つのホストが複数の VLAN に所属できることも示しています。たとえば、ホスト A には、そのインタフェース上に 2 つの VLAN が構成されています。2 番目の VLAN には VLAN ID 456 が構成されており、ポート 3 に接続されています。このポートにも VLAN ID 456 が構成されています。したがって、ホスト A は `infotech0` VLAN と `humres0` VLAN の両方のメンバーです。

VLAN とゾーンの使用

VLAN と Oracle Solaris ゾーンを組み合わせることにより、スイッチなどの単一のネットワーク装置内に複数の仮想ネットワークを構成できます。次の図に示す、3 つの物理ネットワークカード `net0`、`net1`、および `net2` を備えたシステムについて考えます。

図 3–3　複数の VLAN を含むシステム



VLAN を使用しない場合は、特定の機能を実行する異なるシステムを構成し、これらのシステムを個別のネットワークに接続することになります。たとえば、Web サーバーをある LAN に、認証サーバーを別の LAN に、さらにアプリケーションサーバーを 3 番目の LAN に接続します。VLAN とゾーンを使用した場合は、8 つのすべてのシステムを解体し、それらを 1 つのシステム内のゾーンとして構成することができます。次に、VLAN ID を使用して、同じ機能を実行する各ゾーンのセットに VLAN を割り当てます。この図で提供される情報は次のように表に表すことができます。

機能	ゾーン名	VLAN 名	VLAN ID	IP アドレス	NIC
Web サーバー	webzone1	web1	111	10.1.111.0	net0
認証サーバー	authzone1	auth1	112	10.1.112.0	net0
アプリケーションサーバー	appzone1	app1	113	10.1.113.0	net0
Web サーバー	webzone2	web2	111	10.1.111.0	net1
認証サーバー	authzone2	auth2	112	10.1.112.0	net1
アプリケーションサーバー	appzone2	app2	113	10.1.113.0	net1
Web サーバー	webzone3	web3	111	10.1.111.0	net2
認証サーバー	authzone3	auth3	112	10.1.112.0	net2

この図に示されている構成を作成するには、例 3–1 を参照してください。

VLAN の管理

このセクションでは、VLAN の構成および管理の手順について説明します。

▼ VLAN 構成を計画する方法

1. **LAN のトポロジを調べて、VLAN への分割が適切かどうかを判断します。**
   このようなトポロジの基本的な例については、図 3–1 を参照してください。

2. **VLAN ID の番号指定スキームを作成し、各 VLAN に VLAN ID を割り当てます。**

注 – VLAN の番号指定スキームは、ネットワーク上にすでに存在している場合があります。その場合は、既存の VLAN 番号指定スキームに従って VLAN ID を作成する必要があります。

3 各システム上で、特定の **VLAN** のコンポーネントにするインタフェースを決定します。

a. システム上で構成するインタフェースを決定します。

```
# dladm show-link
```

b. システム上の各データリンクに関連付ける **VLAN ID** を特定します。

c. **VLAN** を作成します。

4 インタフェースとネットワークのスイッチの接続を確認します。

各インタフェースの VLAN ID と各インタフェースが接続されているスイッチポートを書き留めます。

5 スイッチの各ポートに、そのポートが接続されているインタフェースと同じ **VLAN ID** を構成します。

構成手順については、スイッチの製造元のドキュメントを参照してください。

▼ VLAN を構成する方法

始める前に この手順では、ゾーンがすでにシステム上に作成されていることを前提としています。ゾーンを作成する手順やゾーンにインタフェースを割り当てる手順は、この手順では説明されていません。ゾーンの構成の詳細については、『Oracle Solaris 11.1 の管理: Oracle Solaris ゾーン、Oracle Solaris 10 ゾーン、およびリソース管理』の第 17 章「非大域ゾーンの計画と構成 (タスク)」を参照してください。

1 管理者になります。

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

2 システムで使用されているリンクのタイプを判定します。

```
# dladm show-link
```

3 データリンク上に **VLAN** リンクを作成します。

```
# dladm create-vlan -l link -v vid vlan-link
```

link VLAN インタフェースの作成に使用するリンクを指定します。

vid VLAN ID 番号を示します。

vlan-link VLAN の名前を指定します。管理用に選択された名前を指定することもできます。

4 **VLAN 構成を確認します。**

```
# dladm show-vlan
```

5 **VLAN 上に IP インタフェースを作成します。**

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

ここで、*interface* は VLAN 名を使用します。

6 **IP インタフェースに IP アドレスを構成します。**

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

例 3–1 VLAN の構成

この例では、図 3–3 に示されている VLAN 構成を作成する方法を示します。この例では、システム内に異なるゾーンがすでに構成されていることを前提にしています。ゾーンの構成の詳細については、『Oracle Solaris 11.1 の管理: Oracle Solaris ゾーン、Oracle Solaris 10 ゾーン、およびリソース管理』のパート II「Oracle Solaris ゾーン」を参照してください。

管理者は、まず VLAN の構成に使用できるリンクを確認してから、特定のリンク上に VLAN を作成します。

```
global# dladm show-link
LINK     CLASS     MTU     STATE     BRIDGE     OVER
net0     phys      1500    up        --         --
net1     phys      1500    up        --         --
net2     phys      1500    up        --         --

global# dladm create-vlan -l net0 -v 111 web1
global# dladm create-vlan -l net0 -v 112 auth1
global# dladm create-vlan -l net0 -v 113 app1
global# dladm create-vlan -l net1 -v 111 web2
global# dladm create-vlan -l net1 -v 112 auth2
global# dladm create-vlan -l net1 -v 113 app2
global# dladm create-vlan -l net2 -v 111 web3
global# dladm create-vlan -l net2 -v 112 auth3

global# dladm show-vlan
LINK      VID     OVER        FLAGS
web1      111     net0        ----
auth1     112     net0        ----
app1      113     net0        ----
web2      111     net1        ----
auth2     112     net1        ----
app2      113     net1        ----
```

```
web3      111     net2        ----
auth3     113     net2        ----
```

リンク情報を表示すると、これらのVLANがリストに含まれています。

```
global# dladm show-link
LINK      CLASS    MTU      STATE     BRIDGE     OVER
net0      phys     1500     up        --         --
net1      phys     1500     up        --         --
net2      phys     1500     up        --         --
web1      vlan     1500     up        --         net0
auth1     vlan     1500     up        --         net0
app1      vlan     1500     up        --         net0
web2      vlan     1500     up        --         net1
auth2     vlan     1500     up        --         net1
app2      vlan     1500     up        --         net1
web3      vlan     1500     up        --         net2
auth3     vlan     1500     up        --         net2
```

次に、管理者はVLANをそれぞれのゾーンに割り当てます。VLANが割り当てられたら、各ゾーンについて次のような情報が表示されます。

```
global# zonecfg -z webzone1 info net
net:
     address not specified
     physical: web1

global# zonecfg -z authzone1 info net
net:
     address not specified
     physical: auth1

global# zonecfg -z appzone2 info net
net:
     address not specified
     physical: app2
```

プロパティー `physical` の値は、特定のゾーンのために設定されているVLANを示します。

次に、管理者は各非大域ゾーンにログインして、VLANにIPアドレスを構成します。

`webzone1` では:

```
webzone1# ipadm create-ip web1
webzone1# ipadm create-addr -a 10.1.111.0/24 web1
ipadm: web1/v4
```

`webzone2` では:

```
webzone2# ipadm create-ip web2
webzone2# ipadm create-addr -a 10.1.111.0/24 web2
ipadm: web2/v4
```

webzone3 では:

```
webzone3# ipadm create-ip web3
webzone3# ipadm create-addr -a 10.1.111.0/24 web3
ipadm: web3/v4
```

authzone1 では:

```
authzone1# ipadm create-ip auth1
authzone1# ipadm create-addr -a 10.1.112.0/24 auth1
ipadm: auth1/v4
```

authzone2 では:

```
authzone2# ipadm create-ip auth2
autzone2# ipadm create-addr -a 10.1.112.0/24 auth2
ipadm: auth2/v4
```

authzone3 では:

```
authzone3# ipadm create-ip auth3
authzone3# ipadm create-addr -a 10.1.112.0/24 auth3
ipadm: auth3/v4
```

appzone1 では:

```
appzone1# ipadm create-ip app1
appzone1# ipadm create-addr -a 10.1.113.0/24 app1
ipadm: app1/v4
```

appzone2 では:

```
appzone2# ipadm create-ip app2
appzone2# ipadm create-addr -a 10.1.113.0/24 app2
ipadm: app2/v4
```

すべての VLAN に IP アドレスが構成されたら、構成は完了です。3 つの VLAN は動作しており、それぞれのゾーンのトラフィックをホストできます。

▼ リンクアグリゲーション上に VLAN を構成する方法

インタフェース上に VLAN を構成する場合と同じ方法で、リンクアグリゲーション上に VLAN を作成することもできます。リンクアグリゲーションについては、第 2 章「リンクアグリゲーションの使用」で説明されています。このセクションでは、VLAN とリンクアグリゲーションの構成について説明します。

1. システムで構成されているリンクアグリゲーションを一覧表示します。

   ```
   # dladm show-link
   ```

2. 選択したリンクアグリゲーション上に作成する**VLAN**ごとに、次のコマンドを発行します。

   ```
   # dladm create-vlan -l link -v vid vlan-link
   ```

   link VLANインタフェースの作成に使用するリンクを指定します。この手順では、リンクはリンクアグリゲーションを指します。

   vid VLAN ID番号を示します。

   vlan-link VLANの名前を指定します。管理用に選択された名前を指定することもできます。

3. 前の手順で作成した**VLAN**ごとに、**VLAN**上に**IP**インタフェースを作成します。

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

   ここで、*interface*はVLAN名を使用します。

4. **VLAN**上の各**IP**インタフェースに、有効な**IP**アドレスを構成します。

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

例3–2 リンクアグリゲーション上に複数のVLANを構成する

この例では、リンクアグリゲーション上に2つのVLANを構成します。VLANにはVLAN ID 193と194がそれぞれ割り当てられます。

```
# dladm show-link
LINK          CLASS     MTU     STATE     BRIDGE     OVER
net0     phys      1500    up        --         ----
net1     phys      1500    up        --         ----
aggr0    aggr      1500    up        --         net0, net1

# dladm create-vlan -l aggr0 -v 193 acctg0
# dladm create-vlan -l aggr0 -v 194 humres0

# ipadm create-ip acctg0
# ipadm create-ip humres0

# ipadm create-addr -a 192.168.10.0/24 acctg0
ipadm: acctg0/v4
# ipadm create-addr -a 192.168.20.0/24 humres0
ipadm: humres0/v4
```

▼ レガシーデバイス上で VLAN を構成する方法

特定のレガシーデバイスは、最大転送単位 (MTU) サイズ (フレームサイズとも呼ばれる) が 1514 バイトのパケットのみを処理します。フレームサイズがこの上限を超えるパケットは破棄されます。このような場合は、37 ページの「VLAN を構成する方法」に示されているのと同じ手順に従います。ただし、VLAN を作成するときは、VLAN を強制的に作成するために `-f` オプションを使用します。

1 **`-f` オプションを使用して VLAN を作成します。**

```
# dladm create-vlan -f -l link -v vid vlan-link
```

link VLAN インタフェースの作成に使用するリンクを指定します。この手順では、リンクはレガシーデバイスを指します。

vid VLAN ID 番号を示します。

vlan-link VLAN の名前を指定します。管理用に選択された名前を指定することもできます。

2 **最大転送単位 (`MTU`) に小さめのサイズ (1496 バイトなど) を設定します。**

```
# dladm set-linkprop -p default_mtu=1496 vlan-link
```

MTU 値を小さくすることによって、リンク層で転送前に VLAN ヘッダーを挿入するための領域が確保されます。

3 **VLAN 内の各ノードの MTU のサイズに同じ小さめの値を設定するために、手順 2 と同じ手順を実行します。**

リンクプロパティー値の変更の詳細については、『Oracle Solaris 11.1 での固定ネットワーク構成を使用したシステムの接続』の「基本的な dladm コマンド」を参照してください。

VLAN 情報の表示

VLAN はデータリンクなので、`dladm show-link` コマンドを使用して VLAN に関する情報を表示できます。ただし、VLAN に固有の情報については `dladm show-vlan` コマンドを使用します。

次の例では、各コマンドで取得される情報の種類を比較します。`dladm show-link` コマンドを使用する最初の出力では、システム上のすべてのデータリンクが、VLAN でないものも含めて表示されます。`dladm show-vlan` コマンドを使用する 2 番目の出力では、データリンク情報のうち、VLAN に関連するものだけが表示されます。

```
# dladm show-link
LINK     CLASS    MTU      STATE     BRIDGE     OVER
net0     phys     1500     up        --         --
```

```
net1      phys      1500      up        --          --
net2      phys      1500      up        --          --
web1      vlan      1500      up        --          net0
auth1     vlan      1500      up        --          net0
app1      vlan      1500      up        --          net0
web2      vlan      1500      up        --          net1
auth2     vlan      1500      up        --          net1
app2      vlan      1500      up        --          net1
web3      vlan      1500      up        --          net2
auth3     vlan      1500      up        --          net2

# dladm show-vlan
LINK       VID     OVER     FLAGS
web1       111     net0     ----
auth1      112     net0     ----
app1       113     net0     ----
web2       111     net1     ----
auth2      112     net1     ----
app2       113     net1     ----
web3       111     net2     ----
auth3      113     net2     ----
```

VLANの変更

`dladm modify-vlan` コマンドを使用して、VLAN を次の方法で変更できます。

- VLAN の VLAN ID を変更します
- 別のベースとなるリンクに VLAN を移行します

VLAN の VLAN ID を変更するには、次のいずれかのコマンドを使用します。

- `dladm modify-vlan -v` *vid* `- L` *datalink*

  このコマンドで、*vid* は、VLAN に割り当てる新しい VLAN ID を指定します。*Datalink* は、VLAN が構成されているベースとなるリンクを表します。このコマンド構文は、データリンク上に VLAN が 1 つだけ存在する場合に使用できます。複数の VLAN が構成されているデータリンクにこのコマンドを使用すると、データリンク上の VLAN には一意の VLAN ID が必要なので、コマンドは失敗します。

- `dladm modify-vlan -v` *vid vlan*

  このコマンドは、単一のデータリンク上にある複数の VLAN の一意の VLAN ID を変更する場合に使用します。データリンク上の各 VLAN には一意の VLAN ID があります。したがって、一度に 1 つずつ VLAN ID を変更する必要があります。図 3–3 の、`net0` 上に構成されている `web1`、`auth1`、および `app1` の VLAN ID を変更するとします。これらの VLAN ID を変更するには、次のように進めます。

  ```
  # dladm modify-vlan -v 123 web1
  # dladm modify-vlan -v 456 app1
  # dladm modify-vlan -v 789 auth1
  ```

VLAN の削除や再構成を行わずに、VLAN をベースとなるデータリンクから別のベースとなるデータリンクに移行します。ベースとなるリンクは、物理リンク、リンクアグリゲーション、または etherstub です。etherstub の詳細については、『Oracle Solaris 11.1 での仮想ネットワークの使用』の「ネットワーク仮想化のコンポーネント」を参照してください。

VLAN を正常に移行するには、VLAN の移動先であるベースとなるデータリンクが、VLAN のデータリンクプロパティーに対応できる必要があります。これらのプロパティーがサポートされていない場合、移行は失敗し、ユーザーに通知されます。移行が正常に行われたあとは、VLAN がネットワークに接続されたままであれば、その VLAN を使用しているすべてのアプリケーションが通常の動作を継続します。

ハードウェアに依存する特定のプロパティーは、VLAN の移行後に変更される場合があります。たとえば、VLAN は常に、そのベースとなるデータリンクと同じ MAC アドレスを共有します。したがって、VLAN を移行すると、その VLAN の MAC アドレスは、ターゲットデータリンクのプライマリ MAC アドレスに変更されます。ほかにも、データリンク状態、リンク速度、MTU サイズなどのプロパティーが影響を受ける可能性があります。ただし、アプリケーションは中断されることなく動作を継続します。

注 – 移行された VLAN では、元のデータリンクのハードウェアレーン統計は一切保持されません。ターゲットデータリンク上で VLAN に使用できるハードウェアレーンが、統計情報の新しい取得元になります。ただし、`dlstat` コマンドによってデフォルトで表示されるソフトウェア統計は保持されます。

VLAN の移行は、グローバルに実行することも選択的に実行することもできます。グローバルな移行とは、あるデータリンク上のすべての VLAN を別のデータリンクに移行することです。グローバルな移行を実行する場合は、移行元のデータリンクとターゲットデータリンクを指定するだけで済みます。次の例では、`ether0` 上のすべての VLAN を `net1` に移動します。

```
# dladm modify-vlan -l net1 -L ether0
```

ここで

- `-L` は、VLAN が構成されている元のデータリンクを表します。
- `-l` は、VLAN の移行先であるターゲットデータリンクを表します。

注 – 移行元データリンクの前にターゲットデータリンクを指定する必要があります。

VLAN の選択的な移行を実行する場合は、移動する VLAN を指定します。図 3–3 に基づく次の例では、net0 から net3 に VLAN が移行されます。

```
# dladm modify-vlan -l net3 web1,auth1,app1
```

注 – VLAN を選択的に移行する場合は、-L オプションを省略します。これはグローバルな移行だけに適用されるオプションです。

移行を実行する間に、VLAN の VLAN ID を変更できます。図 3–3 を基にして、次の例では、複数の VLAN を移行すると同時にそれらの VLAN ID を変更する方法を示します。

```
# dladm show-vlan
LINK    VID      OVER     FLAGS
web1    111      net0     -----
auth1   112      net0     -----
app1    113      net0     -----

# dladm modify vlan -l net3 -v 123 web1
# dladm modify vlan -l net3 -v 456 auth1
# dladm modify vlan -l net3 -v 789 app1
# dladm show-vlan
LINK    VID      OVER     FLAGS
web1    123      net3     -----
auth1   456      net3     -----
app1    789      net3     -----
```

注 – 同等のサブコマンド dladm modify-vnic は、VLAN として構成されている VNIC を移行します。VLAN を移行するのか、VLAN として構成されている VNIC を移行するのかによって、正しいサブコマンドを使用する必要があります。modify-vlan サブコマンドは、dladm show-vlan サブコマンドで表示される VLAN に対して使用します。modify-vnic サブコマンドは、dladm show-vnic サブコマンドで表示される VNIC (VLAN ID を持っているものも含む) に対して使用します。VNIC を変更するには、『Oracle Solaris 11.1 での仮想ネットワークの使用』の「ネットワーク仮想化のコンポーネント」を参照してください。

VLAN の削除

システム上の VLAN 構成を削除するには、dladm delete-vlan コマンドを使用します。

注 – VLANを削除する前に、削除しようとしているVLAN上の既存のIP構成をすべて削除する必要があります。VLAN上にIPインタフェースが存在している場合、VLANの削除は失敗します。

例 3–3 VLAN構成の削除

VLAN構成を削除するには、次の例のような手順を実行します。

```
# dladm show-vlan
LINK      VID     OVER        FLAGS
web1      111     net0     ----
auth1     112     net0     ----
app1      113     net0     ----
web2      111     net1     ----
auth2     112     net1     ----
app2      113     net1     ----
web3      111     net2     ----
auth3     113     net2     ----

# ipadm delete-ip web1
# dladm delete-vlan web1
```

使用事例: リンクアグリゲーションとVLAN構成を組み合わせる

このセクションでは、リンクアグリゲーションとVLANを使用してその上にIPインタフェースを作成するという、ネットワーク構成の組み合わせを作成する方法を例で示します。http://www.oracle.com/us/sun/index.htm には、ほかのネットワークシナリオを紹介する記事があります。

次の例では、4枚のNICを使用するシステムを、8つの個別のサブネットのルーターとして構成する必要があります。この目的を達成するために、サブネットごとに1つずつ、8つのリンクが構成されます。最初に、4枚のすべてのNIC上でリンクアグリゲーションが作成されます。このタグなしリンクが、デフォルトルートが指すネットワークのデフォルトのタグなしサブネットになります。

次に、ほかのサブネットのために、リンクアグリゲーション上でVLANインタフェースが構成されます。これらのサブネットは、色分けされたスキームに基づいて名前が付けられます。それに応じて、VLAN名も同様に、それぞれ対応するサブネットに従って名前が付けられます。最終的な構成は、8つのサブネットに対する8つのリンク (1つのタグなしリンクと7つのタグ付きVLANリンク) で構成されています。この例は、データリンク上にIPインタフェースがすでに存在しているかどうかの確認から始まります。データリンクをアグリゲーションに結合する前に、これらのインタフェースを削除する必要があります。

管理者はまず、データリンク上に構成されている IP インタフェースをすべて削除します。

```
# ipadm show-if
IFNAME    CLASS      STATE   ACTIVE   OVER
lo0       loopback   ok      yes      --
net0      ip         ok      yes      --
net1      ip         ok      yes      --
net2      ip         ok      yes      --
net3      ip         ok      yes      --

# ipadm delete-ip net0
# ipadm delete-ip net1
# ipadm delete-ip net2
# ipadm delete-ip net3
```

その後、管理者はリンクアグリゲーション `default0` を作成します。

```
# dladm create-aggr -P L2,L3 -l net0 -l net1 -l net2 -l net3 default0

# dladm show-link
LINK        CLASS      MTU  STATE    BRIDGE     OVER
net0        phys      1500  up       --         --
net1        phys      1500  up       --         --
net2        phys      1500  up       --         --
net3        phys      1500  up       --         --
default0    aggr      1500  up       --         net0 net1 net2 net3
```

次に、管理者は `default0` 上に VLAN を作成します。

```
# dladm create-vlan -v 2 -l default0 orange0
# dladm create-vlan -v 3 -l default0 green0
# dladm create-vlan -v 4 -l default0 blue0
# dladm create-vlan -v 5 -l default0 white0
# dladm create-vlan -v 6 -l default0 yellow0
# dladm create-vlan -v 7 -l default0 red0
# dladm create-vlan -v 8 -l default0 cyan0

# dladm show-link
LINK        CLASS      MTU  STATE    BRIDGE     OVER
net0        phys      1500  up       --         --
net1        phys      1500  up       --         --
net2        phys      1500  up       --         --
net3        phys      1500  up       --         --
default0    aggr      1500  up       --         net0 net1 net2 net3
orange0     vlan      1500  up       --         default0
green0      vlan      1500  up       --         default0
blue0       vlan      1500  up       --         default0
white0      vlan      1500  up       --         default0
yellow0     vlan      1500  up       --         default0
red0        vlan      1500  up       --         default0
cyan0       vlan      1500  up       --         default0

# dladm show-vlan
LINK          VID   OVER        FLAGS
orange0         2   default0    -----
```

```
green0          3   default0    -----
blue0           4   default0    -----
white0          5   default0    -----
yellow0         6   default0    -----
red0            7   default0    -----
cyan0           8   default0    -----
```

最後に、管理者はVLANリンク上にIPインタフェースを作成し、それらのインタフェースにIPアドレスを割り当てます。

```
# ipadm create-ip orange0
# ipadm create-ip green0
# ipadm create-ip blue0
# ipadm create-ip white0
# ipadm create-ip yellow0
# ipadm create-ip red0
# ipadm create-ip cyan0

# ipadm create-addr -a address orange0
# ipadm create-addr -a address green0
# ipadm create-addr -a address blue0
# ipadm create-addr -a address white0
# ipadm create-addr -a address yellow0
# ipadm create-addr -a address red0
# ipadm create-addr -a address cyan0
```

◆ ◆ ◆ 第 4 章

ブリッジネットワークの管理 (タスク)

この章では、ブリッジネットワークおよびその管理方法について説明します。次の内容について説明します。

ブリッジングの概要

ブリッジは、別々のネットワークセグメントを接続する際に使用され、2 つのノード間のパスとなります。ブリッジで接続すると、接続されたネットワークセグメントは単一のネットワークセグメントであるかように通信を行います。ブリッジングは、ネットワークスタックのデータリンク層 (L2) で実装されます。ブリッジはパケット転送メカニズムを使用して、複数のサブネットワークを相互に接続します。

ブリッジングとルーティングはどちらも、ネットワーク上のリソースの場所に関する情報を配布するために使用できますが、いくつかの点で異なります。ルーティングは IP 層 (L3) で実装され、ルーティングプロトコルを使用します。データリンク層では、ルーティングプロトコルは使用されません。代わりに、パケットの転送先は、ブリッジに接続されているリンクで受信されたネットワークトラフィックを検査することで特定されます。

パケットを受信すると、その発信元アドレスが検査されます。パケットの発信元アドレスには、受信されたリンクにパケットを送信したノードが関連付けられます。その後、受信されたパケットが宛先アドレスと同じアドレスを使用すると、ブリッジはリンク上のパケットをそのアドレスに転送します。

発信元アドレスに関連付けられたリンクは、ブリッジネットワーク内の別のブリッジに接続された中間リンクである場合があります。時間が経過すると、ブリッジネットワーク内のすべてのブリッジが、どのリンクが特定のノードにパ

ケットを送信するのかを「学習」します。したがって、ホップバイホップのブリッジング方式で最終宛先にパケットを送信する際には、パケットの宛先アドレスが使用されます。

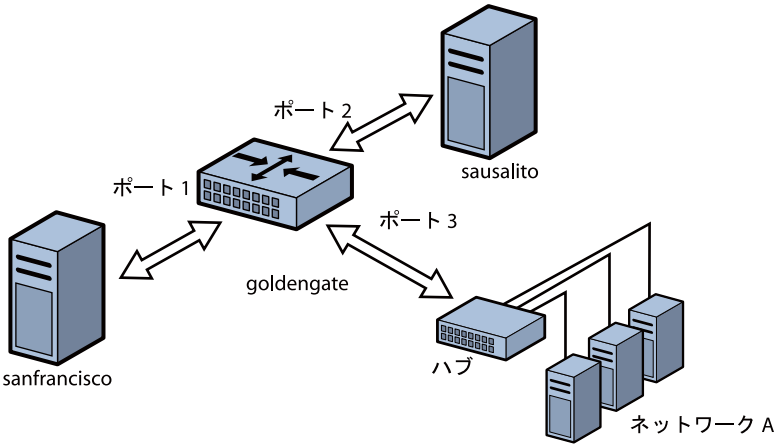
ローカルの「リンク停止」通知は、特定のリンク上のすべてのノードが到達不能になったことを示します。この状況では、リンクへのパケット転送が停止し、リンク上のすべての転送エントリがフラッシュされます。古い転送エントリも時間が経過するとフラッシュされます。リンクが復元されると、リンク上で受信されたパケットは新規として処理されます。パケットの発信元アドレスに基づいた「学習」プロセスが再開します。このプロセスによって、アドレスが宛先アドレスとして使用されたときに、ブリッジが正常にパケットをそのリンク上に転送できるようになります。

パケットを宛先に転送するには、ブリッジに接続されたすべてのリンク上でブリッジがプロミスキャスモードで待機する必要があります。プロミスキャスモードで待機すると、最大回線速度でパケットが永久に循環してしまう転送ループの発生に対して、ブリッジが脆弱になります。したがって、ブリッジングではスパニングツリープロトコル (STP) メカニズムを使用して、サブネットワークが使用不可になるネットワークループを回避します。

ブリッジに STP および RSTP (Rapid Spanning Tree Protocol) を使用することに加えて、Oracle Solaris では TRILL (Transparent Interconnection of Lots of Links) 保護拡張機能もサポートされています。デフォルトでは STP が使用されますが、ブリッジングコマンドで `-P trill` オプションを指定すれば、TRILL も使用できます。

ブリッジ構成を使用して、単一のネットワークに接続すると、ネットワーク内のさまざまなノードの管理が簡略化されます。これらのセグメントをブリッジ経由で接続すると、すべてのノードが単一のブロードキャストネットワークを共有します。したがって、ルーターを使用してトラフィックをネットワークセグメント間で転送するのではなく、IP などのネットワークプロトコルを使用することによって、各ノードはほかのノードに到達できます。ブリッジを使用しない場合は、ノード間の IP トラフィックの転送を許可するように IP ルーティングを構成する必要があります。

図 4–1 単純なブリッジネットワーク

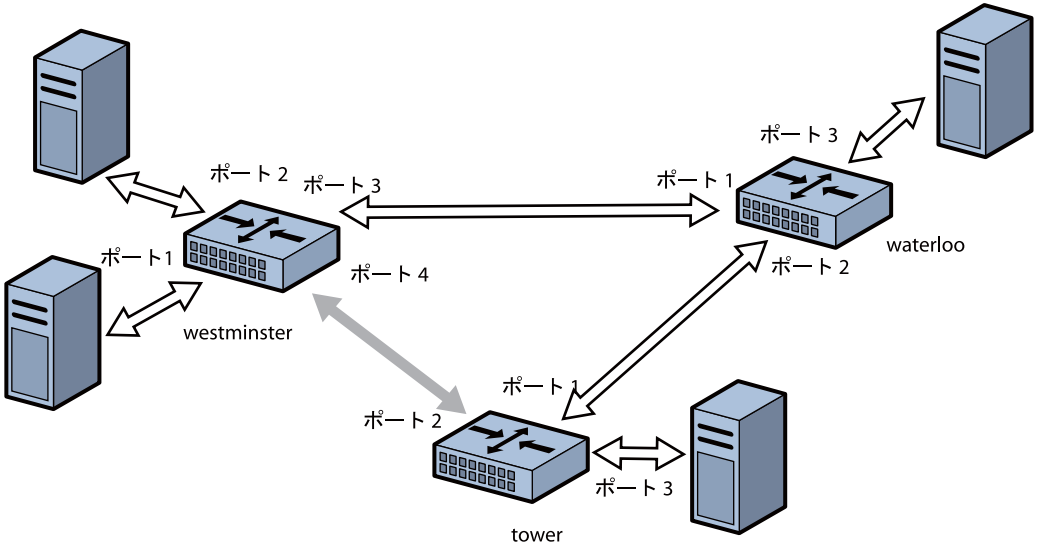


前の図は、単純なブリッジネットワーク構成を示しています。ブリッジ goldengate は、ブリッジングが構成された Oracle Solaris システムです。sanfrancisco および sausalito は、物理的にブリッジに接続されたシステムです。ネットワーク A ではハブが使用され、片側では物理的にブリッジに接続され、反対側ではコンピュータシステムに接続されています。ブリッジポートは、bge0、bge1、bge2 などのリンクです。

ブリッジネットワークはリング状に形成して、物理的に複数のブリッジを相互に接続できます。このような構成は、ネットワークでよく使用されます。このタイプの構成では、古いパケットがリングの周りをエンドレスにループすることによってネットワークリンクが飽和状態になる問題が発生する可能性があります。このようなループ状況から保護するために、Oracle Solaris のブリッジには STP と TRILL の両方のプロトコルが実装されています。大部分のハードウェアブリッジでは、STP ループ保護も実装されています。

次の図は、リング状に構成されたブリッジネットワークを示しています。この構成では、3 台のブリッジがあります。2 台のシステムが物理的に westminster ブリッジに接続されています。1 台のシステムが物理的に waterloo ブリッジに接続されています。さらに、1 台のシステムが物理的に tower ブリッジに接続されています。ブリッジは、物理的にブリッジポートを介して相互に接続されています。

図4–2 ブリッジネットワークリング



ループ保護のために STP または RSTP を使用すると、ループ内の接続の 1 つによるパケット転送を回避することによって、物理的なループが軽減されます。図 4–2 は、`westminster` と `tower` ブリッジ間の物理リンクがパケット転送に使用されないことを示しています。

ループ保護を実行するために使用可能な物理リンクをシャットダウンすると、STP および RSTP によって帯域幅が消費されることに注意してください。

STP および RSTP とは異なり、TRILL ではループを回避するために物理リンクがシャットダウンされません。その代わりに、TRILL はネットワーク内の TRILL ノードごとに最短パス情報を計算し、その情報を使用して個々の宛先にパケットを転送します。

その結果、TRILL を使用すると、すべてのリンクが常に使用中のまま保持できます。IP によるループの処理方法と同様にループが処理されるため、ループが問題になることはありません。つまり、TRILL は必要に応じてルートを作成し、転送のホップ制限を使用することで、一時的なループ状態によって発生する問題を回避します。

注意 – SPARC プラットフォームでは、`local-mac-address?=false` を設定しないでください。これを設定すると、複数のポートや同じネットワーク上で同一の MAC アドレスがシステムによって誤って使用されます。

注意 – 可能な限り高いレベルのパフォーマンスが必要な場合は、ブリッジにリンクを構成しないでください。ブリッジングでは、ベースとなるインタフェースをプロミスキャスモードにして、システムのハードウェア層、ドライバ層、およびその他の層で実装されている数多くの重要な最適化を無効にする必要があります。このようなパフォーマンス拡張機能が無効になることは、ブリッジングメカニズムでは避けられない結果です。

システムのリンクの一部がブリッジされていないために、これらの制約の対象ではないシステムでは、ブリッジを使用できます。このようなパフォーマンスの問題は、ブリッジの一部になるように構成されたリンクにのみ影響を与えます。

STP の詳細は、IEEE 802.1D-1998 を参照してください。RSTP の詳細は、IEEE 820.1Q-2004 を参照してください。TRILL の詳細は、Internet Engineering Task Force (IETF) TRILL draft documents (`http://tools.ietf.org/wg/trill`) を参照してください。

リンクプロパティー

次のリンクプロパティーは、`dladm show-linkprop` コマンドで表示でき、`dladm set-linkprop` および `reset-linkprop` コマンドで変更できます。

`default_tag` リンク間で送受信されるタグなしパケットに、デフォルトの仮想ローカルエリアネットワーク (VLAN) ID を定義します。有効な値は `0` - `4094` です。デフォルト値は `1` です。このプロパティーは、VLAN 以外および仮想ネットワークインタフェースカード (VNIC) 以外のタイプのリンクにのみ存在します。この値を `0` に設定すると、ポートとの間で送受信されるタグなしパケットの転送が無効になります (これは MAC プロパティーです)。

注 – このプロパティーをブリッジングの範囲外で使用して、リンクに IEEE Port VLAN Identifier (PVID) を指定することもできます。ベースリンク自体が自動的に PVID を表すため、`default_tag` をゼロ以外にすると、リンク上に同じ ID を持つ VLAN を作成できません。

たとえば、`net0` で PVID を `5` に設定する場合は、`net0` で ID `5` の VLAN を作成できません。この状況で VLAN `5` を指定するには、`net1` を使用します。

`default_tag` は、そのリンクで作成された既存の VLAN の ID と等しくなるように設定できません。たとえば、次のコマンドは `net0` に VLAN `22` を作成します。

```
# dladm create-vlan -l net0 -v 22 myvlan0
```

この状況では、`net0` と `myvlan0` の両方で同じ VLAN を表すため、`default_tag` を `22` に設定できません。

`default_tag` を `0` に設定すると、`net0` 上のタグなしパケットと VLAN との関連付けを解除できます。この状況では、このようなパケットが構成されたブリッジから転送されなくなります。

`forward` ブリッジ経由のトラフィック転送を有効および無効にします。このプロパティーは、VNIC リンク以外のすべてのリンクに存在します。有効な値は `1` (真) と `0` (偽) です。デフォルト値は `1` です。トラフィック転送を無効にすると、リンクインスタンスに関連付けられた VLAN では、トラフィックがブリッジから転送されなくなります。転送を無効にすることは、従来のブリッジで「許可された設定」から VLAN を削除することに相当します。つまり、ローカルクライアントからベースとなるリンクへの VLAN ベースの入出力は続行されますが、ブリッジベースの転送は実行されません。

`stp` STP と RSTP を有効および無効にします。有効な値は `1` (真) と `0` (偽) です。デフォルト値は `1` であり、STP と RSTP が有効になります。このプロパティーを `0` に設定すると、リンクでは STP も RSTP も使用されず、常に転送モードに移行されます。転送モードでは、ブリッジプロトコルデータユニット (BPDU) ガードが使用されます。エンドノードに接続されるポイントツーポイントリンクを構成する場合は、STP と RSTP を無効にします。このプロパティーは、VLAN および VNIC 以外のタイプのリンクにのみ存在します。

`stp_cost` リンクを使用する際の STP と RSTP のコスト値を表します。有効な値は `1` - `65535` です。デフォルト値は `0` であり、自動的にコストがリンクタイプ別に計算されるように指定されます。次の値は、いくつかのリンクタイプのコストを表します: `100` は 10M ビット/秒、`19` は 100M ビット/秒、`4` は 1G ビット/秒、`2` は 10G ビット/秒。

`stp_edge` ポートがほかのブリッジに接続されているかどうかを指定します。有効な値は `1` (真) と `0` (偽) です。デフォルト値は `1` です。このプロパティーが `0` に設定されていると、どのタイプの BPDU も検出されない場合でも、デーモンはポートがほかのブリッジに接続されているとみなします。

`stp_p2p` 接続モードタイプを指定します。有効な値は、`true`、`false`、および `auto` です。デフォルト値は `auto` であり、ポイントツーポイント接続が自動的に検出されます。`true` を指定すると、ポイントツーポイントモードが強制されます。`false` を指定すると、標準のマルチポイントモードが強制されます。

`stp_priority` STP および RSTP ポートの優先順位値を設定します。有効な値は `0` - `255` です。デフォルト値は `128` です。STP および RSTP ポートの優先順位値を使用してその値を PVID の先頭に付加することで、ブリッジの優先ルートポートが決定されます。数値を小さくすると、優先度が高くなります。

STP デーモン

`dladm create-bridge` コマンドを使用して作成された各ブリッジは、`svc:/network/bridge` の同じ名前の付いたサービス管理機能 (SMF) インスタンスとして表されます。各インスタンスは、STP を実装する `/usr/lib/bridged` デーモンのコピーを実行します。

たとえば、次のコマンドは `pontevecchio` というブリッジを作成します。

```
# dladm create-bridge pontevecchio
```

`svc:/network/bridge:pontevecchio` という SMF サービスインスタンス、および `/dev/net/pontevecchio0` という可観測性ノードが作成されます。

安全のため、デフォルトで標準 STP がすべてのポートで実行されます。STP などの一部のブリッジプロトコル形式が実行されないブリッジでは、ネットワークで長期にわたる転送ループが発生する可能性があります。Ethernet のパケットにはホップ数または TTL (Transistor-to-Transistor Logic) が存在しないため、このようなループはネットワークにとって致命的です。

特定のポートが別のブリッジに接続されていない (たとえば、ポートがホストシステムに直接ポイントツーポイント接続されているため) ときは、管理上そのポートの STP を無効にすることができます。ブリッジ上のすべてのポートで STP が無効になっている場合でも、引き続き STP デーモンは実行されます。デーモンが引き続き実行される理由は、次のとおりです。

- 新たに追加されるポートを処理するため
- BPDU ガードを実装するため
- 必要に応じて、ポート上の転送を有効または無効にするため

ポートで STP が無効になっている場合、`bridged` デーモンは BPDU (BPDU ガード) の待機を続行します。デーモンは `syslog` を使用してエラーにフラグを付け、ポート上の転送を無効にして、重大なネットワークの構成ミスを示します。リンクが停止してから再起動すると、またはリンクを手動で削除してから再度追加すると、ふたたびリンクが有効になります。

ブリッジの SMF サービスインスタンスを無効にすると、STP デーモンが停止したときに、そのポート上のブリッジ転送も停止します。インスタンスが再起動すると、STP は初期状態から起動します。

TRILL デーモン

`dladm create-bridge -P trill` コマンドを使用して作成された各ブリッジは、`svc:/network/bridge` および `svc:/network/routing/trill` の同じ名前の付いた SMF インスタンスとして表されます。`svc:/network/routing/trill` の各インスタンスでは、TRILL プロトコルを実装する `/usr/lib trilld` デーモンのコピーが実行されます。

たとえば、次のコマンドは `bridgeofsighs` というブリッジを作成します。

```
# dladm create-bridge -P trill bridgeofsighs
```

`svc:/network/bridge:bridgeofsighs` と `svc:/network/routing/trill:bridgeofsighs` という 2 つの SMF サービスが作成されます。さらに、`/dev/net/bridgeofsighs0` という可観測性ノードも作成されます。

ブリッジのデバッグ

各ブリッジインスタンスには、可観測性ノードが割り当てられます。これは、`/dev/net/` ディレクトリに表示され、ブリッジ名の末尾に `0` を付加した名前が付けられます。

可観測性ノードは、snoop および wiresha k ユーティリティーで使用することを目的としています。このノードは、パケット転送が暗黙的に破棄される点を除いて、標準の Ethernet インタフェースと同様に動作します。可観測性ノード上の IP は plumb することができず、パッシブオプションを使用しない限り、バインド要求 (DL_BIND_REQ) も実行できません。

可観測性ノードが使用されると、ブリッジで処理されるすべてのパケットの未変更コピーが 1 つ作成され、ユーザーはそれをモニターとデバッグに使用できます。この動作は、従来のブリッジのモニタリングポートと同様であり、通常のデータリンクプロバイダインタフェース (DLPI) のプロミスキャスモード規則の対象になります。pfmod コマンド、または snoop と wireshark ユーティリティーの機能を使用すると、VLAN ID に基づいてパケットをフィルタリングできます。

配信されたパケットは、ブリッジで受信されたデータを表します。

注 – ブリッジングプロセスで VLAN タグが追加、削除、または変更される場合、dlstat コマンドで表示されるデータには、このプロセスが発生する前の状態が記述されます。さまざまなリンクで別々の default_tag 値が使用されると、このようなまれな状況で混乱する可能性があります。

特定のリンクで転送および受信されるパケットを (ブリッジングプロセスの完了後に) 確認するには、可観測性ノード上ではなく、個々のリンクで snoop コマンドを実行します。

また、dlstat コマンドを使用して、ネットワークパケットがリンク上のネットワークリソースをどのように使用しているかについて統計を取得することもできます。詳細は、『Oracle Solaris 11.1 での仮想ネットワークの使用』の第 4 章「Oracle Solaris でのネットワークトラフィックとリソース使用状況の監視」を参照してください。

ブリッジが使用された場合にリンク動作がどのように変化するか

次のセクションでは、ネットワーク構成でブリッジが使用された場合にリンク動作がどのように変化するかついて説明します。

標準のリンク動作の詳細は、31 ページの「VLAN の配備: 概要」を参照してください。

DLPI の動作

次に、ブリッジが有効になっているときのリンク動作の相違点について説明します。

- リンク起動 (DL_NOTE_LINK_UP) およびリンク停止 (DL_NOTE_LINK_DOWN) の通知は集約して配信されます。つまり、すべての外部リンクがリンク停止ステータスを示しているときは、MAC 層を使用している上位のクライアントにもリンク停止イベントが表示されます。ブリッジ上の任意の外部リンクがリンク起動ステータスを示しているときは、すべての上位クライアントにリンク起動イベントが表示されます。

  この集約されたリンク起動およびリンク停止の報告が実行される理由は、次のとおりです。

  - リンク停止ステータスが表示されると、リンク上のノードは到達不能になります。ブリッジングコードにより別のリンク経由で引き続きパケットを送受信できる場合は、これに該当しません。実際のリンクステータスを必要とする管理アプリケーションは、既存の MAC 層のカーネル統計を使用してステータスを表示できます。このようなアプリケーションは、ハードウェアのステータス情報を報告し、パケットの転送には関与しないという点で、IP などの通常のクライアントとは異なります。
  - すべての外部リンクが停止すると、ブリッジ自体がシャットダウンしたかのようにステータスが表示されます。このような特殊なケースでは、どのノードにも到達できない可能性があるとシステムで認識されます。すべてのインタフェースが「実体」であり (仮想ではなく)、すべてのリンクが接続解除されている場合は、ローカルのみの通信を使用可能にする際にブリッジを使用できないというトレードオフがあります。
- リンク固有の機能はすべて汎用になっています。実際の出力リンクは完全にクライアントによって決定されるわけではないため、特殊なハードウェアの高速化機能がサポートされているリンクでは、これらの機能を使用できません。ブリッジ転送機能は、宛先 MAC アドレスに基づいて出力リンクを選択する必要があり、この出力リンクはブリッジ上の任意のリンクにすることができます。

ブリッジネットワーク上の VLAN の管理

デフォルトでは、システムに構成された VLAN は、ブリッジインスタンス上のすべてのポート間でパケットを転送します。ベースとなるリンクがブリッジの一部である場合に、`dladm create-vlan` または `dladm create-vnic -v` コマンドを呼び出すと、該当ブリッジリンク上の指定された VLAN のパケット転送も有効になります。

リンク上に VLAN を構成して、ブリッジ上のその他のリンク間のパケット転送を無効にするには、`dladm set-linkprop` コマンドで VLAN の `forward` プロパティーを設定して、転送を無効にする必要があります。

ベースとなるリンクがブリッジの一部として構成されているときに、ブリッジングで VLAN を自動的に有効にするには、`dladm create-vlan` コマンドを使用します。

標準準拠の STP では、VLAN は無視されます。ブリッジプロトコルは、タグなし BPDU メッセージを使用して 1 つのループなしのトポロジのみを計算し、このツリートポロジを使用してリンクを有効および無効にします。ネットワークでプロビジョニングされている複製リンクは、STP によって自動的に無効になっても、構成された VLAN が接続解除されないように構成する必要があります。つまり、ブリッジされたバックボーンのあらゆるところですべての VLAN を実行するか、すべての冗長リンクを慎重に検査する必要があります。

TRILL プロトコルは、複雑な STP の規則には従いません。その代わりに、TRILL は自動的に、VLAN タグが変更されていないパケットをカプセル化して、ネットワーク経由で渡します。つまり、TRILL は、単一のブリッジネットワーク内で同じ VLAN ID が再利用されている場合、分離された VLAN を結び付けます。

この STP との重要な相違点は、ネットワークの分離されたセクションで VLAN タグを再利用して 4094 の制限よりも多くの VLAN セットを管理できることを意味します。このようにネットワークを管理する際には TRILL を使用できませんが、プロバイダベースの VLAN などのその他のソリューションは実装できる場合があります。

VLAN を使用した STP ネットワークでは、STP によって「不正な」リンクが無効になったときに、VLAN パーティション分割を回避するようにフェイルオーバーの特性を構成することが困難になる可能性があります。分離された VLAN では比較的小さい機能喪失がありますが、TRILL モデルの堅牢性に比べれば些細なことです。

VLAN の動作

ブリッジは許可された VLAN セットおよび `default_tag` プロパティーをリンクごとに検査することで、パケット転送を実行します。一般的なプロセスは次のとおりです。

1. 入力 **VLAN** の決定。このプロセスは、システムがリンクで着信パケットを受信すると開始されます。パケットが受信されると、VLAN タグがチェックされます。タグが存在しない場合や、タグが優先順位のみ (ゼロに設定) の場合は、該当するリンクに構成された `default_tag` プロパティー値 (ゼロに設定されていない場合) が内部 VLAN タグとみなされます。タグが存在しない、またはゼロで、`default_tag` がゼロの場合は、パケットが無視されます。タグなしの転送は実行されません。タグが存在し、`default_tag` 値に等しい場合も、パケットが無視されます。それ以外の場合は、タグが着信 VLAN パケットとみなされます。
2. リンクメンバーシップのチェック。入力 VLAN がこのリンクで許可された VLAN として構成されていない場合は、パケットが無視されます。その後、転送が計算され、出力リンクにも同じチェックが行われます。

3. タグの更新。出力リンクの VLAN タグ (この時点でゼロ以外) が default_tag 値に等しい場合は、優先順位に関係なく、パケット上のタグ (存在する場合) が削除されます。出力リンクの VLAN タグが default_tag 値に等しくない場合は、タグが現在存在しなければ追加され、そのタグが発信パケットに設定され、現在の優先順位がパケットにコピーされます。

注 – 転送で複数のインタフェースにパケットが送信される場合 (ブロードキャスト、マルチキャスト、および不明な宛先の場合) は、出力リンクごとに個別に出力リンクのチェックおよびタグの更新を実行する必要があります。転送はタグ付きの場合も、タグなしの場合もあります。

ブリッジ構成の表示

次の例は、ブリッジ構成およびブリッジサービスに関する情報を表示する方法を示しています。

- 次のコマンドを実行すると、ブリッジに関する情報を表示できます。

```
# dladm show-bridge
BRIDGE      PROTECT ADDRESS            PRIORITY DESROOT
tonowhere   trill   32768/66:ca:b0:39:31:5d 32768 32768/66:ca:b0:39:31:5d
sanluisrey  stp     32768/ee:2:63:ed:41:94 32768 32768/ee:2:63:ed:41:94
pontoon     trill   32768/56:db:46:be:b9:62 32768 32768/56:db:46:be:b9:62
```

- 次のコマンドを実行すると、ブリッジに関する TRILL ニックネーム情報を表示できます。

```
# dladm show-bridge -t tonowhere
NICK FLAGS LINK        NEXTHOP
38628 --   simblue2    56:db:46:be:b9:62
58753 L    --          --
```

ブリッジの管理

Oracle Solaris では、dladm コマンドおよび SMF 機能を使用してブリッジを管理します。インスタンスの障害管理リソース識別子 (FMRI) svc:/network/bridge を使用してブリッジインスタンスを有効化、無効化、およびモニターするには、SMF コマンドを使用します。ブリッジを作成または削除することに加えて、ブリッジにリンクを割り当てたり、ブリッジからリンクを削除したりするには、dladm を使用します。

ブリッジの管理 (タスクマップ)

次の表は、ブリッジを管理するために使用可能なタスクを示しています。

タスク	説明	参照先
構成されているブリッジに関する情報を表示します。	システムに構成されているブリッジに関する情報を表示するには、`dladm show-bridge` コマンドを使用します。構成されているブリッジ、リンク、統計情報、およびカーネル転送エントリに関する情報を表示します。	62 ページの「構成されているブリッジに関する情報を表示する方法」
ブリッジに接続されているリンクに関する構成情報を表示します。	システムに構成されているリンクに関する情報を表示するには、`dladm show-link` コマンドを使用します。リンクがブリッジに関連付けられている場合は、`BRIDGE` フィールドの出力を参照してください。	63 ページの「ブリッジリンクに関する構成情報を表示する方法」
ブリッジを作成します。	ブリッジを作成して、オプションのリンクを追加するには、`dladm create-bridge` コマンドを使用します。 デフォルトでは、ブリッジは STP を使用して作成されます。代わりに TRILL を使用してブリッジを作成するには、`dladm create-bridge` コマンド行に `-P trill` を追加します。または、`dladm modify-bridge` コマンドを使用して TRILL を有効にします。	64 ページの「ブリッジを作成する方法」
ブリッジの保護タイプを変更します。	ブリッジの保護タイプを変更するには、`dladm modify-bridge` コマンドを使用します。	65 ページの「ブリッジの保護タイプを変更する方法」
ブリッジにリンクを追加します。	既存のブリッジに 1 つ以上のリンクを追加するには、`dladm add-bridge` コマンドを使用します。	65 ページの「既存のブリッジに 1 つ以上のリンクを追加する方法」
ブリッジからリンクを削除します。	ブリッジからリンクを削除するには、`dladm remove-bridge` コマンドを使用します。すべてのリンクが削除されるまで、ブリッジを削除できません。	66 ページの「ブリッジからリンクを削除する方法」
システムからブリッジを削除します。	システムからブリッジを削除するには、`dladm delete-bridge` コマンドを使用します。	67 ページの「システムからブリッジを削除する方法」

▼ 構成されているブリッジに関する情報を表示する方法

この手順では、さまざまなオプションとともに dladm show-bridge コマンドを使用して、構成されているブリッジに関する各種情報を表示する方法を説明します。

dladm show-bridge コマンドオプションの詳細は、dladm(1M) のマニュアルページを参照してください。

1 管理者になります。

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

2 1 台のブリッジまたは構成されているすべてのブリッジに関する情報を表示します。

- ブリッジのリストを表示します。

  ```
  # dladm show-bridge
  ```

- ブリッジのリンク関連のステータスを表示します。

  ```
  # dladm show-bridge -l bridge-name
  ```

- ブリッジの統計情報を表示します。

  ```
  # dladm show-bridge -s bridge-name
  ```

  注 – 報告されるブリッジ統計情報の名前および定義は、変更されることがあります。

- ブリッジのリンク関連の統計情報を表示します。

  ```
  # dladm show-bridge -ls bridge-name
  ```

- ブリッジのカーネル転送エントリを表示します。

  ```
  # dladm show-bridge -f bridge-name
  ```

- ブリッジに関する TRILL 情報を表示します。

  ```
  # dladm show-bridge -t bridge-name
  ```

例 4–1 ブリッジの情報の表示

次に、さまざまなオプションを付けた dladm show-bridge コマンドの使用例を示します。

- 次のコマンドは、システムに構成されているすべてのブリッジに関する情報を表示します。

```
# dladm show-bridge
BRIDGE       PROTECT ADDRESS           PRIORITY DESROOT
goldengate   stp     32768/8:0:20:bf:f 32768    8192/0:d0:0:76:14:38
baybridge    stp     32768/8:0:20:e5:8 32768    8192/0:d0:0:76:14:38
```

- 次のコマンドは、単一のブリッジインスタンス tower に関するリンク関連のステータス情報を表示します。構成されているプロパティーを表示するには、dladm show-linkprop コマンドを使用します。

```
# dladm show-bridge -l tower
LINK         STATE        UPTIME   DESROOT
net0         forwarding   117      8192/0:d0:0:76:14:38
net1         forwarding   117      8192/0:d0:0:76:14:38
```

- 次のコマンドは、指定されたブリッジ terabithia に関する統計情報を表示します。

```
# dladm show-bridge -s terabithia
BRIDGE       DROPS        FORWARDS
terabithia   0            302
```

- 次のコマンドは、指定されたブリッジ london 上のすべてのリンクに関する統計情報を表示します。

```
# dladm show-bridge -ls london
LINK         DROPS     RECV      XMIT
net0         0         360832    31797
net1         0         322311    356852
```

- 次のコマンドは、指定されたブリッジ avignon のカーネル転送エントリを表示します。

```
# dladm show-bridge -f avignon
DEST              AGE     FLAGS  OUTPUT
8:0:20:bc:a7:dc   10.860  --     net0
8:0:20:bf:f9:69   --      L      net0
8:0:20:c0:20:26   17.420  --     net0
8:0:20:e5:86:11   --      L      net1
```

- 次のコマンドは、指定されたブリッジ key に関する TRILL 情報を表示します。

```
# dladm show-bridge -t key
NICK FLAGS LINK        NEXTHOP
38628 --   london      56:db:46:be:b9:62
58753 L    --          --
```

▼ ブリッジリンクに関する構成情報を表示する方法

dladm show-link の出力には、BRIDGE フィールドが含まれています。リンクがブリッジのメンバーである場合、このフィールドによってメンバーであるブリッジの名前が識別されます。このフィールドはデフォルトで表示されます。リンクがブリッジの一部でない場合、-p オプションが使用されていれば、このフィールドは空白になります。それ以外の場合は、フィールドには -- と表示されます。

dladm show-link の出力には、別々のリンクとしてブリッジの可観測性ノードも表示されます。このノードの場合、既存の OVER フィールドにブリッジのメンバーであるリンクが一覧表示されます。

1 管理者になります。

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

2 ブリッジのメンバーであるリンクに関する構成情報を表示します。

```
# dladm show-link [-p]
```

-p オプションを付けると、解析可能な形式で出力されます。

▼ ブリッジを作成する方法

この手順では、デフォルトのプロトコルである STP を使用してブリッジを作成する方法を説明します。ブリッジ作成のオプションの詳細は、dladm(1M) のマニュアルページで dladm create-bridge コマンドの説明を参照してください。

注 – TRILL を使用してブリッジを作成するには、dladm create-bridge コマンドに -P trill を追加します。または、dladm modify-bridge コマンドを使用して TRILL を有効にします。

---

dladm create-bridge コマンドは、ブリッジインスタンスを作成し、任意で新しいブリッジに 1 つ以上のネットワークリンクを割り当てます。デフォルトではブリッジインスタンスがシステムに存在しないため、Oracle Solaris はデフォルトでネットワークリンク間にブリッジを作成しません。

リンク間にブリッジを作成するには、少なくとも 1 つのブリッジインスタンスを作成する必要があります。各ブリッジインスタンスは個別のものです。ブリッジにはブリッジ間の転送接続は含まれず、リンクは最大で 1 つのブリッジのメンバーです。

bridge-name は任意の文字列であり、正当な SMF サービスインスタンス名である必要があります。この名前は、エスケープシーケンスが含まれない FMRI コンポーネントです。つまり、空白、ASCII 制御文字、および次の文字が存在してはいけません:

```
; / ? : @ & = + $ , % < > # "
```

default という名前は、SUNW 文字列で始まるすべての名前と同様、予約されています。末尾に数字が付く名前は、デバッグに使用する可観測性デバイスを作成するために予約されています。可観測性デバイスが使用されるため、正当なブリッジインスタンス名は、正当な dlpi 名となるようにさらに制限されます。この名前は、英字

または下線文字で開始および終了する必要があります。名前の残りの部分には、英数字と下線文字を含めることができます。

1 管理者になります。

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

2 ブリッジを作成します。

```
# dladm create-bridge [-l link]... bridge-name
```

`-l` *link* オプションは、ブリッジにリンクを追加します。指定されたリンクのいずれかを追加できない場合、コマンドは失敗し、ブリッジは作成されません。

次の例は、`net0` と `net1` リンクを接続して、`brooklyn` ブリッジを作成する方法を示しています。

```
# dladm create-bridge -l net0 -l net1 brooklyn
```

▼ ブリッジの保護タイプを変更する方法

この手順では、`dladm modify-bridge` コマンドを使用して、保護タイプを STP から TRILL または TRILL から STP に変更する方法を説明します。

- ブリッジの保護タイプを変更します。

```
# dladm modify-bridge -P protection-type bridge-name
```

`-P` *protection-type* オプションは、使用する保護タイプを指定します。デフォルトの保護タイプは STP (`-P stp`) です。TRILL 保護タイプを使用するには、`-P trill` オプションを使用します。

次の例は、`brooklyn` ブリッジの保護タイプをデフォルトの STP から TRILL に変更する方法を示しています。

```
# dladm modify-bridge -P trill brooklyn
```

▼ 既存のブリッジに 1 つ以上のリンクを追加する方法

この手順では、ブリッジインスタンスに 1 つ以上のリンクを追加する方法を説明します。

1 つのリンクは、最大 1 つのブリッジのメンバーになることができます。したがって、別のブリッジインスタンスにリンクを移動する場合は、まず現在のブリッジからリンクを削除してから、別のブリッジに追加する必要があります。

ブリッジに割り当てるリンクを VLAN、VNIC、またはトンネルにすることはできません。ブリッジに割り当てることができるのは、アグリゲーションの一部として受け入れ可能なリンク、またはアグリゲーション自体であるリンクのみです。

同じブリッジに割り当てるリンクはすべて、MTU 値が同じである必要があります。Oracle Solaris では、既存のリンクの MTU 値を変更できることに注意してください。ただし、ブリッジを再起動する前に、割り当てられたリンクを削除または変更して MTU 値が一致するまで、ブリッジインスタンスは保守状態になります。

ブリッジに割り当てられたリンクは、Ethernet タイプ (802.3 および 802.11 メディアを含む) である必要があります。

1 管理者になります。

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

2 既存のブリッジに新しいリンクを追加します。

```
# dladm add-bridge -l new-link  bridge-name
```

次の例は、既存のブリッジ `rialto` に `net2` リンクを追加する方法を示しています。

```
# dladm add-bridge -l net2 rialto
```

▼ ブリッジからリンクを削除する方法

この手順では、ブリッジインスタンスから 1 つ以上のリンクを削除する方法を説明します。ブリッジを削除する場合は、この手順を使用します。ブリッジを削除する前に、まずブリッジのリンクをすべて削除する必要があります。

1 管理者になります。

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

2 ブリッジからリンクを削除します。

```
# dladm remove-bridge [-l link]... bridge-name
```

次の例は、ブリッジ `charles` から `net0`、`net1`、および `net2` リンクを削除する方法を示しています。

```
# dladm remove-bridge -l net0 -l net1 -l net2 charles
```

▼ システムからブリッジを削除する方法

この手順では、ブリッジインスタンスを削除する方法を説明します。ブリッジを削除する前に、まず `dladm remove-bridge` コマンドを実行して、接続されているリンクを停止する必要があります。66 ページの「ブリッジからリンクを削除する方法」を参照してください。

1. 管理者になります。

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

2. システムからブリッジを削除します。

   ```
   # dladm delete-bridge bridge-name
   ```

   次の例は、まず `coronado` ブリッジから `net0`、`net1`、および `net2` リンクを削除してから、システムからブリッジ自体を削除する方法を示しています。

   ```
   # dladm remove-bridge -l net0 -l net1 -l net2 coronado
   # dladm delete-bridge coronado
   ```

◆ ◆ ◆ 第 5 章

IPMP の概要

IP ネットワークマルチパス (IPMP) は、複数の IP インタフェースを単一の論理インタフェースにグループ化できるレイヤー 3 テクノロジです。障害検出、透過的なアクセスフェイルオーバー、パケット負荷分散などの機能を提供する IPMP は、システムに対してネットワークを常に使用可能に保つことで、ネットワークパフォーマンスを向上させます。

この章で扱う内容は、次のとおりです。

注 – この章と第 6 章「IPMP の管理 (タスク)」に含まれる IPMP の説明で使われているインタフェースという用語はすべて、特に *IP* インタフェースを意味しています。ネットワークインタフェースカード (NIC) のように、修飾語がこの用語の異なる使用法を明示的に示す場合を除き、この用語は常に、IP 層で構成されたインタフェースを指します。

Oracle Solaris での IPMP

Oracle Solaris では、IPMP には次の機能があります。

- IPMP により、複数の IP インタフェースを、IPMP グループと呼ばれる単一のグループに構成できます。IPMP グループは、そのベースとなる複数の IP インタフェースを含め、全体で単一の *IPMP* インタフェースとして表されます。このインタフェースは、ネットワークスタックの IP 層のほかのインタフェースとまったく同様に扱われます。IP 管理タスク、ルーティングテーブル、アドレス解決プロトコル (ARP) テーブル、ファイアウォール規則、およびその他の IP 関連手順はすべて、IPMP インタフェースを参照することで IPMP グループを操作します。

- システムは、ベースとなるアクティブインタフェース間でデータアドレスの分配を処理します。IPMP グループの作成時に、データアドレスはアドレスプールとして IPMP インタフェースに属します。続いてカーネルが自動的かつランダムに、グループのデータアドレスとベースとなるアクティブインタフェースをバインドします。
- `ipmpstat` ツールは、IPMP グループに関する情報を取得するための主要ツールです。このコマンドは、グループのベースとなる IP インタフェース、検査用アドレスとデータアドレス、使用される障害検出のタイプ、故障したインタフェースなど、IPMP 構成のあらゆる側面についての情報を提供します。`ipmpstat` の機能、使用できるオプション、および各オプションが生成する出力についてはすべて、110 ページの「IPMP 情報のモニタリング」で説明しています。
- IPMP インタフェースにカスタマイズされた名前を割り当てて、IPMP グループをより簡単に識別できます。91 ページの「IPMP グループの構成」を参照してください。

IPMP を使用する利点

インタフェースの故障や、インタフェースが保守のためにオフラインになっている場合など、さまざまな要因によりインタフェースが使用不可能になる可能性があります。IPMP を使用しないと、その使用不可能になったインタフェースに関連付けられたどの IP アドレスを使用しても、システムと通信できなくなります。さらに、それらの IP アドレスを使用する既存の接続が切断されます。

IPMP を使用すると、複数の IP インタフェースを 1 つの *IPMP* グループに構成できます。このグループは、ネットワークトラフィックを送受信するデータアドレス付きの IP インタフェースのように機能します。グループ内のベースとなるインタフェースの 1 つが故障すると、グループ内の残りのアクティブなベースとなるインタフェースの間でデータアドレスが再分配されます。したがって、インタフェースの 1 つが故障しても、グループはネットワークの接続性を維持します。IPMP では、グループで最低 1 つのインタフェースが使用可能であれば、ネットワーク接続を常に使用できます。

IPMP は、IPMP グループ内のインタフェースセット全体にアウトバウンドネットワークトラフィックを自動的に分散させることにより、全体的なネットワークパフォーマンスを向上させます。このプロセスは、アウトバウンド「負荷分散」と呼ばれます。システムはさらに、アプリケーションによって発信元 IP アドレスが指定されなかったパケットに対して発信元アドレス選択を実行することにより、インバウンド負荷分散も間接的に制御します。ただし、アプリケーションが発信元 IP アドレスを明示的に選択した場合は、システムはその発信元アドレスを変更しません。

注 – リンクアグリゲーションは IPMP と同様の機能を実行して、ネットワークのパフォーマンスと可用性を向上させます。これら 2 つのテクノロジの比較については、付録 B 「リンクアグリゲーションと IPMP: 機能比較」を参照してください。

IPMP を使用するための規則

IPMP グループの構成はシステム構成によって決まります。

IPMP を使用するときは、次の規則に従ってください。

- 同じ LAN 上の複数の IP インタフェースは、1 つの IPMP グループに構成される必要があります。LAN は、同じリンク層ブロードキャストドメインに属するノードを含む VLAN や有線と無線の両方のローカルネットワークなど、さまざまなローカルネットワーク構成を広く指します。

  注 – 同じリンク層 (L2) ブロードキャストドメイン上の複数の IPMP グループはサポートされていません。通常、L2 ブロードキャストドメインは特定のサブネットに対応します。したがって、サブネットごとに IPMP グループを 1 つだけ構成する必要があります。

- IPMP グループのベースとなる IP インタフェースが異なる LAN にまたがってはいけません。

たとえば、3 つのインタフェースを備えたシステムが 2 つの別個の LAN に接続されているとします。一方の LAN に 2 つの IP インタフェースが接続し、他方の LAN に単一の IP インタフェースが接続します。この場合、最初の規則により、1 つ目の LAN に接続する 2 つの IP インタフェースは 1 つの IPMP グループとして構成される必要があります。2 番目の規則のため、2 つ目の LAN に接続する単一の IP インタフェースがその IPMP グループのメンバーになることはできません。単一の IP インタフェースでは、IPMP 構成は必須ではありません。ただし、その単一のインタフェースの可用性をモニターするために、そのインタフェースを 1 つの IPMP グループに構成してもかまいません。単一インタフェースの IPMP 構成の詳細については、73 ページの「IPMP インタフェース構成のタイプ」を参照してください。

別のケースとして、1 つ目の LAN へのリンクが 3 つの IP インタフェースから構成され、もう 1 つのリンクが 2 つのインタフェースから構成される場合を考えます。この設定は 2 つの IPMP グループの構成を必要とします (1 つ目の LAN に接続する 3 つのインタフェースから成るグループと、2 つ目の LAN に接続する 2 つのインタフェースから成るグループ)。

IPMP のコンポーネント

IPMP のソフトウェアコンポーネントは次のとおりです。

- マルチパスデーモン `in.mpathd` - インタフェースの故障や修復を検出します。ベースとなるインタフェースで検査用アドレスが構成されている場合、このデーモンは、リンクベースの障害検出とプローブベースの障害検出の両方を実行します。このデーモンは、使用される障害検出手法のタイプに応じて、インタフェース上で適切なフラグを設定または解除し、そのインタフェースが故障しているかどうかや修復されたかどうかを示します。オプションとして、IPMP グループに属するように構成されていないインタフェースを含むすべてのインタフェースの可用性をモニターするようにこのデーモンを構成することもできます。障害検出については、82 ページの「IPMP での障害検出」を参照してください。

  `in.mpathd` デーモンは、IPMP グループ内のアクティブインタフェースの指定も制御します。このデーモンは、IPMP グループの作成時に最初に構成されたのと同じ数のアクティブインタフェースを維持しようとします。したがって、`in.mpathd` は、ベースとなるインタフェースを必要に応じて起動したり停止したりして、管理者が構成したポリシーとの一貫性を保ちます。`in.mpathd` デーモンがベースとなるインタフェースの起動を管理する方法の詳細については、74 ページの「IPMP の動作方法」を参照してください。このデーモンの詳細は、`in.mpathd(1M)` のマニュアルページを参照してください。

- **IP** カーネルモジュール - IPMP グループ内で使用可能な一連の IP データアドレスをグループ内で使用可能な一連のベースとなる IP インタフェースに分配することで、アウトバウンド負荷分散を管理します。さらにこのモジュールは、発信元アドレス選択を実行してインバウンド負荷分散を管理します。このモジュールのどちらの役割も、ネットワークトラフィックのパフォーマンスを改善します。

- **IPMP** 構成ファイル (`/etc/default/mpathd`) - デーモンの動作を定義します。

  次のパラメータを設定するには、このファイルをカスタマイズします。

  - プローブベースの障害検出を実行している場合に、プローブするターゲットインタフェース
  - 障害を検出するためにターゲットをプローブする時間
  - 故障したインタフェースが修復されたあとにフラグ付けするステータス
  - モニターする IP インタフェースの範囲 (IPMP グループに属するように構成されていない、システム内の IP インタフェースも含めるかどうか)

  この構成ファイルを変更する手順については、108 ページの「IPMP デーモンの動作を構成する方法」を参照してください。

- `ipmpstat` コマンド - IPMP の全体としてのステータスに関するさまざまなタイプの情報を提供します。さらにこのツールは、各 IPMP グループのベースとなる IP インタフェースに関するその他の情報や、グループで構成されているデータアドレ

スや検査用アドレスも表示します。このコマンドの詳細は、110 ページの「IPMP 情報のモニタリング」および `ipmpstat(1M)` のマニュアルページを参照してください。

IPMP インタフェース構成のタイプ

IPMP 構成は、通常同じ LAN に接続された同じシステムの複数の物理インタフェースで構成されます。これらのインタフェースは、次のいずれかの構成の IPMP グループに属することができます。

- アクティブ - アクティブ構成 – ベースとなるインタフェースのすべてがアクティブである IPMP グループ。「アクティブインタフェース」とは、IPMP グループによって現時点で使用可能な IP インタフェースのことです。

  注 – デフォルトでは、あるベースとなるインタフェースを IPMP グループの一部になるように構成すると、そのインタフェースはアクティブになります。

- アクティブ - スタンバイ構成 – 少なくとも 1 つのインタフェースがスタンバイインタフェースとして管理上構成されている IPMP グループ。スタンバイインタフェースはアイドル状態になっていますが、その構成方法に応じて、可用性を追跡するためにマルチパスデーモンによってモニターされます。インタフェースによってリンク障害通知がサポートされている場合は、リンクベースの障害検出が使用されます。インタフェースで検査用アドレスが構成されている場合は、プローブベースの障害検出も使用されます。いずれかのアクティブインタフェースが故障すると、スタンバイインタフェースが必要に応じて自動的に配備されます。1 つの IPMP グループには、スタンバイインタフェースを必要な数だけ構成できます。

単独インタフェースをそれ自体の IPMP グループ内で構成することもできます。単独インタフェース IPMP グループは、複数のインタフェースを持つ IPMP グループと同じように動作します。ただし、この IPMP 構成は、ネットワークトラフィックの高可用性を提供しません。ベースとなるインタフェースが故障すると、システムはトラフィックを送受信する機能をすべて失います。単一インタフェースの IPMP グループを構成する目的は、障害検出を使用してインタフェースの可用性をモニターすることです。インタフェースで検査用アドレスを構成することにより、プローブベースの障害検出を使用してマルチパスデーモンでそのインタフェースを追跡できます。

単一インタフェースの IPMP グループ構成は通常、Oracle Solaris Cluster ソフトウェアなど、より幅広いフェイルオーバー機能を備えたほかのテクノロジとともに使用されます。システムは引き続き、ベースとなるインタフェースのステータスをモニターできますが、Oracle Solaris Cluster ソフトウェアは、障害発生時にネットワークの可用性を保証するための機能を提供します。Oracle Solaris Cluster ソフトウェアの詳細は、『Oracle Solaris Cluster Concepts Guide』を参照してください。

ベースとなるインタフェースが削除されたグループなど、ベースとなるインタフェースを持たない IPMP グループも存在できます。この IPMP グループは破棄はされませんが、このグループを使用してトラフィックを送受信することはできません。このグループでベースとなるインタフェースがオンラインになると、それらのインタフェースに IPMP インタフェースのデータアドレスが割り当てられ、システムがネットワークトラフィックのホスティングを再開します。

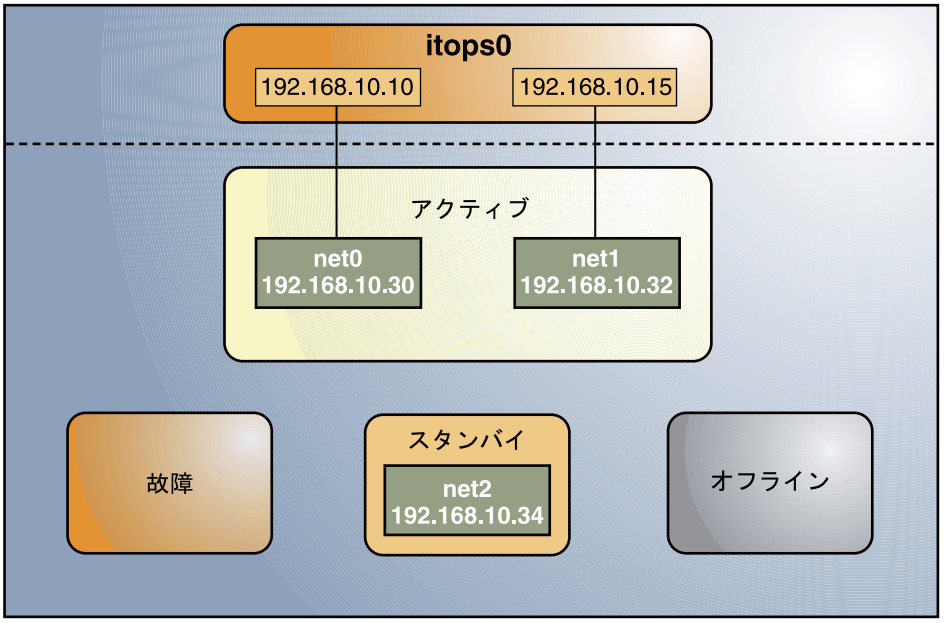
IPMP の動作方法

IPMP は、IPMP グループの作成時に構成されたアクティブインタフェースとスタンバイインタフェースの元の数を保持しようとすることによって、ネットワークの可用性を維持します。

グループ内の特定のベースとなる IP インタフェースの可用性を判定するための IPMP 障害検出は、リンクベースまたはプローブベース、あるいはその両方にすることができます。あるベースとなるインタフェースが故障したと IPMP が判定した場合、そのインタフェースは故障としてフラグが付けられ、使用できなくなります。次に、故障したインタフェースに関連付けられていたデータ IP アドレスが、グループ内で機能している別のインタフェースに再分配されます。さらに、使用可能な場合は、スタンバイインタフェースも配備され、アクティブインタフェースの元の数を維持します。

次の図に示すような、3 つのインタフェースを含むアクティブ - スタンバイ構成の IPMP グループ `itops0` を考えます。

図 5–1 IPMP アクティブ-スタンバイ構成



IPMP グループ `itops0` は次のように構成されています。

- このグループには、2 つのデータアドレス `192.168.10.10` と `192.168.10.15` が割り当てられています。
- ベースとなる 2 つのインタフェースがアクティブインタフェースとして構成され、柔軟なリンク名 `net0` と `net1` が割り当てられています。
- このグループにはスタンバイインタフェースが 1 つ含まれており、これにも柔軟なリンク名 `net2` が割り当てられています。
- プローブベースの障害検出が使用されるため、アクティブインタフェースとスタンバイインタフェースは次のような検査用アドレスで構成されています。
  - `net0: 192.168.10.30`
  - `net1: 192.168.10.32`
  - `net2: 192.168.10.34`

注 – 図 5–1、図 5–2、図 5–3、および図 5–4 のアクティブ、オフライン、スタンバイ、および故障の各領域は、ベースとなるインタフェースのステータスを示しているだけであり、物理的な場所を示しているわけではありません。この IPMP 実装内では、インタフェースまたはアドレスの物理的な移動や IP インタフェースの転送は一切発生しません。これらの領域の役割は、ベースとなるインタフェースのステータスが故障、修復のいずれかの結果としてどのように変化するかを示すことだけです。

さまざまなオプションとともに `ipmpstat` コマンドを使用して、既存の IPMP グループに関する特定の種類の情報を表示できます。その他の例については、110 ページの「IPMP 情報のモニタリング」を参照してください。

次の `ipmpstat` コマンドは、図 5–1 の IPMP 構成に関する情報を表示します。

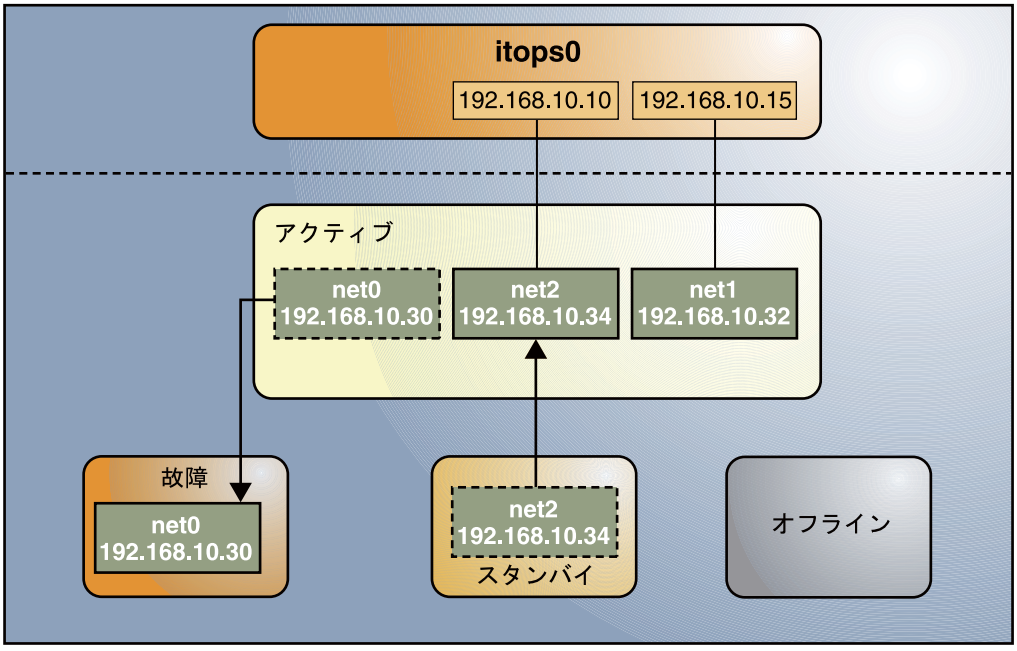
```
# ipmpstat -g
GROUP     GROUPNAME     STATE     FDT       INTERFACES
itops0    itops0        ok        10.00s    net1 net0 (net2)
```

グループのベースとなるインタフェースに関する情報を表示するには、次のように入力します。

```
# ipmpstat -i
INTERFACE   ACTIVE     GROUP     FLAGS      LINK       PROBE     STATE
net0        yes        itops0    -------    up         ok        ok
net1        yes        itops0    --mb---    up         ok        ok
net2        no         itops0    is-----    up         ok        ok
```

IPMP は、アクティブインタフェースの元の数を維持できるようにベースとなるインタフェースを管理することで、ネットワークの可用性を維持します。したがって、`net0` が故障すると、IPMP グループが引き続き 2 つのアクティブインタフェースを持てるように、`net2` が配備されます。`net2` のアクティブ化を次の図に示します。

図 5–2　IPMP でのインタフェースの故障



注 – 図 5–2 のデータアドレスとアクティブインタフェースとの 1 対 1 のマッピングは、図を単純化するためのものにすぎません。IP カーネルモジュールは、データアドレスとインタフェースとの間の 1 対 1 の関係に必ずしも縛られることなく、データアドレスをランダムに割り当てることができます。

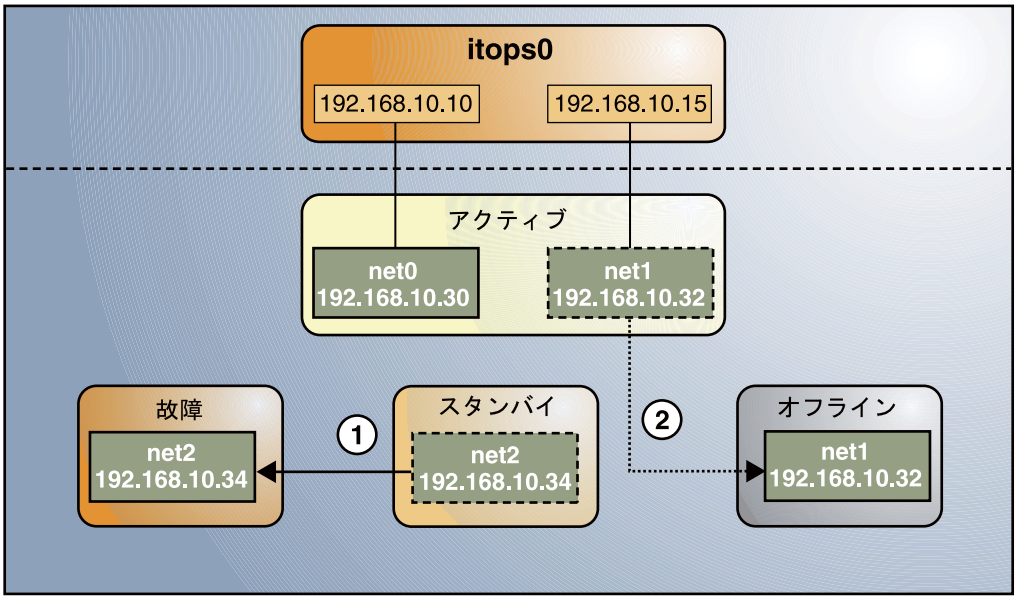
ipmpstat コマンドは、図 5–2 の情報を次のように表示します。

```
# ipmpstat -i
INTERFACE   ACTIVE     GROUP      FLAGS      LINK       PROBE      STATE
net0        no         itops0     -------    up         failed     failed
net1        yes        itops0     --mb---    up         ok         ok
net2        yes        itops0     -s-----    up         ok         ok
```

net0 が修復されると、アクティブインタフェースとしてのステータスに戻ります。一方、net2 は元のスタンバイステータスに戻されます。

別の故障シナリオを図 5–3 に示します。このシナリオでは、スタンバイインタフェース net2 が故障します (1)。そのあと、アクティブインタフェースの 1 つである net1 が管理者によってオフラインに切り替えられます (2)。その結果、この IPMP グループには、機能しているインタフェース net0 が 1 つ残されます。

図 5–3 IPMP でのスタンバイインタフェースの故障



ipmpstat コマンドは、図 5–3 の情報を次のように表示します。

```
# ipmpstat -i
INTERFACE   ACTIVE     GROUP     FLAGS       LINK        PROBE     STATE
net0        yes        itops0    -------     up          ok        ok
net1        no         itops0    --mb-d-     up          ok        offline
net2        no         itops0    is-----     up          failed    failed
```

この故障では、インタフェースが修復されたあとの回復の動作は異なります。この回復プロセスは、修復後の構成と比較した IPMP グループのアクティブインタフェースの元の数に依存します。この回復プロセスを次の図に視覚的に示します。

図 5–4　IPMP の回復プロセス

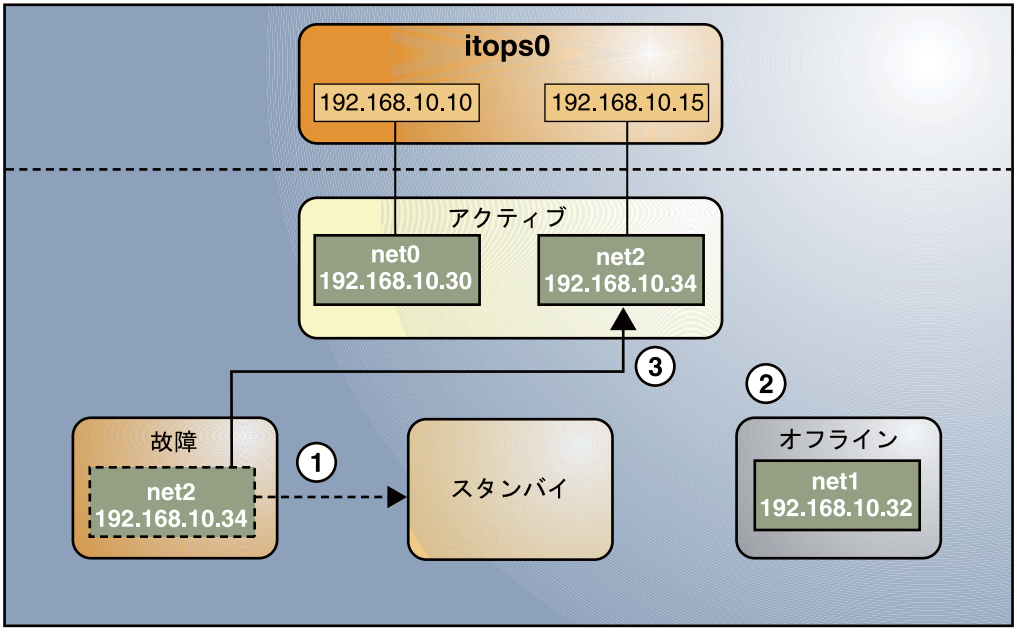


図 5–4 で、net2 が修復されると、通常、スタンバイインタフェースとしての元のステータスに戻ります (1)。ところが、この IPMP グループは依然として、アクティブインタフェースの元の数である 2 個を反映していません。これは、net1 が引き続きオフラインのままになっているからです (2)。したがって、IPMP は代わりに net2 をアクティブインタフェースとして配備します (3)。

ipmpstat コマンドは、修復後の IPMP シナリオを次のように表示します。

```
# ipmpstat -i
INTERFACE   ACTIVE     GROUP      FLAGS       LINK        PROBE      STATE
net0        yes        itops0     -------     up          ok         ok
net1        no         itops0     --mb-d-     up          ok         offline
net2        yes        itops0     -s-----     up          ok         ok
```

FAILBACK=no モードも構成されたアクティブインタフェースが故障に関与している場合にも、同様の回復プロセスが発生します。その場合、故障したアクティブインタフェースが修復されても、自動的にはアクティブステータスに戻りません。図 5–2 の net0 が FAILBACK=no モードに構成されているとします。そのモードでは、修復された net0 は、最初はアクティブインタフェースであったとしても、スタンバイインタフェースになります。この IPMP グループのアクティブインタフェースの元の数である 2 個を維持するように、インタフェース net2 はアクティブのままになります。

ipmpstat コマンドは、回復情報を次のように表示します。

```
# ipmpstat -i
INTERFACE   ACTIVE     GROUP      FLAGS      LINK         PROBE      STATE
net0        no         itops0     i------    up           ok         ok
net1        yes        itops0     --mb---    up           ok         ok
net2        yes        itops0     -s-----    up           ok         ok
```

このタイプの構成の詳細については、86 ページの「FAILBACK=no モード」を参照してください。

IPMP のアドレス指定

IPMP 障害検出は、IPv4 ネットワークと、IPv4 および IPv6 のデュアルスタックネットワークの両方で構成できます。IPMP で構成されたインタフェースは、次のセクションで説明する 2 種類のアドレスをサポートします。Oracle Solaris 11 から、IP アドレスは IPMP インタフェース上にのみあり、データアドレスとして指定されるのに対して、テストアドレスはベースとなるインタフェース上にあります。

データアドレス

データアドレスとは従来の IPv4 および IPv6 アドレスのことであり、ブート時に DHCP サーバーによって IP インタフェースに動的に割り当てられるか、あるいは `ipadm` コマンドを使用して手動で割り当てられます。データアドレスは IPMP インタフェースに割り当てられます。標準の IPv4 パケットトラフィック (および該当する場合は IPv6 パケットトラフィック) が「データトラフィック」とみなされます。データトラフィックは、IPMP インタフェースでホストされているデータアドレスを使用し、その IPMP インタフェースまたはグループのアクティブインタフェースを通過します。

検査用アドレス

検査用アドレスとは、プローブベースの障害および修復検出を実行するために `in.mpathd` デーモンによって使用される、IPMP 固有のアドレスのことです。検査用アドレスもやはり、DHCP サーバーによって動的に割り当てることも、`ipadm` コマンドを使用して手動で割り当てることもできます。IPMP グループのベースとなるインタフェースには、検査用アドレスのみ割り当てられます。あるベースとなるインタフェースが故障した場合、そのインタフェースの検査用アドレスが引き続き `in.mpathd` デーモンによってプローブベースの障害検出のために使用され、そのインタフェースがその後修復されたかどうかがチェックされます。

注 – プローブベースの障害検出を使用する場合のみ、検査用アドレスを構成する必要があります。それ以外の場合は、推移的プローブを有効にすることで、検査用アドレスを使用しなくても障害を検出できます。検査用アドレスを使用する場合としない場合のプローブベースの障害検出の詳細については、82 ページの「プローブベースの障害検出」を参照してください。

以前の IPMP 実装では、特にインタフェースの故障中にアプリケーションによって使用されないように、検査用アドレスは DEPRECATED としてマークされる必要がありました。現在の実装では、検査用アドレスはベースとなるインタフェース内に存在しています。したがって、IPMP を認識しないアプリケーションによってこれらのアドレスが間違って使用されることはなくなりました。ただし、これらのアドレスがデータパケットの発信元の候補として考慮されないように、システムは自動的に、NOFAILOVER フラグの付いたすべてのアドレスを DEPRECATED としてマークします。

サブネット上の任意の IPv4 アドレスを検査用アドレスとして使用できます。IPv4 アドレスは、多くのサイトでは限定リソースなので、ルート指定できない RFC 1918 プライベートアドレスを検査用 IP アドレスとして指定したい場合もあります。in.mpathd デーモンは、検査用アドレスと同じサブネット上にあるほかのホストと ICMP プローブのみを交換します。RFC 1918 形式の検査用アドレスを使用していない場合は、ネットワーク上のほかのシステム (ルーターが望ましい) を適切な RFC 1918 サブネットのアドレスで必ず構成してください。この構成により、in.mpathd デーモンは、ターゲットシステムと正常に検査信号を交換できます。RFC 1918 プライベートアドレスの詳細については、RFC 1918, Address Allocation for Private Internets (http://www.ietf.org/rfc/rfc1918.txt?number=1918) を参照してください。

有効な IPv6 検査用 IP アドレスは、物理インタフェースのリンクローカルアドレスだけです。IPMP 検査用 IP アドレスとして機能する別の IPv6 アドレスは必要ありません。IPv6 リンクローカルアドレスは、インタフェースのメディアアクセスコントロール (MAC) アドレスに基づいています。リンクローカルアドレスは、インタフェースがブート時に IPv6 を使用できるようになったり、インタフェースが ipadm コマンドによって手動で構成されたりした場合に、自動的に構成されます。

IPMP グループですべてのグループのインタフェースに IPv4 と IPv6 の両方が使用される場合には、別個の IPv4 検査用アドレスを構成する必要はない場合があります。in.mpathd デーモンは、IPv6 リンクローカルアドレスを検査用 IP アドレスとして使用します。

IPMP での障害検出

IPMP は、トラフィックを送受信するネットワークの継続的な可用性を保証するために、IPMP グループのベースとなる IP インタフェースに対して障害検出を実行します。故障したインタフェースは、修復されるまで使用不可能なままになります。残りのアクティブインタフェースが機能し続ける一方で、既存のスタンバイインタフェースが必要に応じて配備されます。

`in.mpathd` デーモンは、次の種類の障害検出を処理します。

- プローブベースの障害検出 (2 種類)
  - 検査用アドレスが構成されない (推移的プローブ)。
  - 検査用アドレスが構成される。
- リンクベースの障害検出 (NIC ドライバがサポートしている場合)

プローブベースの障害検出

プローブベースの障害検出では、ICMP プローブを使用してインタフェースが故障しているかどうかをチェックします。この障害検出手法の実装は、検査用アドレスが使用されるかどうかによって決まります。

検査用アドレスを使用するプローブベースの障害検出

この障害検出手法では、検査用アドレスを使用する ICMP 検査信号メッセージを送受信します。プローブトラフィックまたはテストトラフィックとも呼ばれるこれらのメッセージは、インタフェース経由で同じローカルネットワーク上の 1 つ以上のターゲットシステムに送信されます。`in.mpathd` デーモンは、プローブベースの障害検出用に構成されたすべてのインタフェースを経由してすべてのターゲットを個別にプローブします。ある特定のインタフェースで、連続する 5 つのプローブに対して応答がない場合、`in.mpathd` はそのインタフェースに障害があるものとみなします。プローブを発信する頻度は、障害検出時間 (*FDT*) に依存します。障害検出時間のデフォルト値は 10 秒です。ただし、FDT は IPMP 構成ファイルで調整できます。手順については、108 ページの「IPMP デーモンの動作を構成する方法」を参照してください。

プローブベースの障害検出を最適化するには、`in.mpathd` デーモンからのプローブを受信する複数のターゲットシステムを設定する必要があります。複数のターゲットシステムを設定することで、報告された障害の性質をより正確に判定できます。たとえば、唯一定義されたターゲットシステムから応答がない場合、そのターゲットシステムの障害を示している可能性もあれば、IPMP グループのインタフェースの 1 つの障害を示している可能性もあります。これに対し、いくつかのターゲットシステムのうちの 1 つのシステムだけがプローブに応答しない場合は、IPMP グループ自体ではなく、ターゲットシステムで障害が発生している可能性があります。

in.mpathd デーモンは、プローブするターゲットシステムを動的に決定します。まず、デーモンはルーティングテーブル内で、IPMP グループのインタフェースに関連付けられた検査用アドレスと同じサブネット上にあるターゲットシステムを検索します。そのようなターゲットが見つかった場合、デーモンはそれらをプローブのターゲットとして使用します。同じサブネット上でターゲットシステムが見つからない場合、デーモンは、リンク上の近くのホストをプローブするマルチキャストパケットを送信します。ターゲットシステムとして使用するホストを決定するために、すべてのホストのマルチキャストアドレス (IPv4 では 224.0.0.1、IPv6 では ff02::1) にマルチキャストパケットが送信されます。エコーパケットに応答する最初の 5 つのホストが、プローブのターゲットとして選択されます。マルチキャストプローブに応答したルーターまたはホストを検出できない場合、デーモンはプローブベースの障害を検出できません。この場合、ipmpstat -i コマンドはプローブの状態を unknown として報告します。

ホストルートを使用して、in.mpathd デーモンが使用するターゲットシステムのリストを明示的に構成できます。手順については、105 ページの「プローブベースの障害検出の構成」を参照してください。

検査用アドレスを使用しないプローブベースの障害検出

検査用アドレスを使用しないこの手法は、2 種類のプローブを使用して実装されています。

- **ICMP** プローブ

  ICMP プローブは、ルーティングテーブルに定義されたターゲットをプローブするために、IPMP グループ内のアクティブインタフェースによって送信されます。アクティブインタフェースとは、そのインタフェースのリンク層 (L2) アドレス宛てのインバウンド IP パケットを受信できるベースとなるインタフェースのことです。ICMP プローブは、データアドレスをそのプローブの発信元アドレスとして使用します。ICMP プローブがそのターゲットに到達し、ターゲットから応答が得られた場合、そのアクティブインタフェースは動作しています。

- 推移的プローブ

  推移的プローブは、アクティブインタフェースをプローブするために、IPMP グループ内の代替インタフェースによって送信されます。代替インタフェースとは、インバウンド IP パケットを能動的に受信しないベースとなるインタフェースのことです。

  たとえば、4 つのベースとなるインタフェースから成る IPMP グループを考えます。このグループでは、データアドレスは 1 つ構成されていますが、検査用アドレスは 1 つも構成されていません。この構成では、アウトバウンドパケットはベースとなるインタフェースをすべて使用できます。一方、インバウンドパケットは、データアドレスがバインドされたインタフェースによってのみ受信できます。インバウンドパケットを受信できない残り 3 つのベースとなるインタフェースが、「代替」インタフェースとなります。

代替インタフェースがアクティブインタフェースへのプローブの送信と応答の受信に成功した場合、そのアクティブインタフェースは機能しており、推論により、プローブを送信した代替インタフェースも機能しています。

注 – Oracle Solaris では、プローブベースの障害検出は検査用アドレスを使用して動作します。検査用アドレスなしでプローブベースの障害検出を選択するには、推移的プローブを手動で有効にする必要があります。手順については、106 ページの「使用する障害検出手法を選択する方法」を参照してください。

グループ障害

グループ障害は、IPMP グループ内のすべてのインタフェースが同時に故障したと思われる場合に発生します。この場合、ベースとなるインタフェースは一切使用できません。また、すべてのターゲットシステムが同時に故障したときに、プローブベースの障害検出が有効になっていた場合、`in.mpathd` デーモンはその現在のターゲットシステムをすべてフラッシュし、新しいターゲットシステムに対してプローブします。

検査用アドレスを持たない IPMP グループでは、アクティブインタフェースをプローブできる単一のインタフェースがプローバとして指定されます。この指定されたインタフェースには、`FAILED` フラグと `PROBER` フラグが両方とも設定されます。このインタフェースにデータアドレスがバインドされるため、このインタフェースは引き続き、ターゲットをプローブして回復を検出できます。

リンクベースの障害検出

リンクベースの障害検出は、インタフェースがその種の障害検出をサポートしている場合は、常に有効です。

サードパーティーのインタフェースがリンクベースの障害検出をサポートしているかどうかを判定するには、`ipmpstat -i` コマンドを使用します。ある特定のインタフェースの出力の `LINK` 列に `unknown` ステータスが含まれる場合、そのインタフェースはリンクベースの障害検出をサポートしません。デバイスに関するより具体的な情報については、製造元のドキュメントを参照してください。

リンクベースの障害検出をサポートするネットワークドライバは、インタフェースのリンク状態をモニターし、リンク状態が変わったときにネットワークサブシステムに通知します。変更を通知されると、ネットワークサブシステムは、インタフェースの `RUNNING` フラグを適宜設定または解除します。インタフェースの `RUNNING` フラグが解除されたことを検出すると、`in.mpathd` デーモンは即座にインタフェースに障害があるものとみなします。

障害検出と匿名グループ機能

IPMP は匿名グループでの障害検出をサポートしています。デフォルトでは、IPMP は IPMP グループに属するインタフェースのステータスのみをモニターします。ただし、どの IPMP グループにも属さないインタフェースのステータスも追跡するように IPMP デーモンを構成することもできます。したがって、これらのインタフェースは「匿名グループ」の一部とみなされます。`ipmpstat -g` コマンドを発行した場合、匿名グループは二重ダッシュ (`--`) として表示されます。匿名グループ内のインタフェースでは、データアドレスが検査用アドレスとしても機能します。これらのインタフェースは名前付きの IPMP グループに属していないため、これらのアドレスはアプリケーションから可視となります。IPMP グループの一部でないインタフェースの追跡を有効にする方法については、108 ページの「IPMP デーモンの動作を構成する方法」を参照してください。

物理インタフェースの回復検出

「修復検出時間」は障害検出時間の 2 倍です。障害検出のデフォルト時間は 10 秒です。したがって、修復検出のデフォルト時間は 20 秒です。故障したインタフェースが `RUNNING` フラグでふたたびマークされ、障害検出手法がそのインタフェースを修復済みとして検出すると、`in.mpathd` デーモンはそのインタフェースの `FAILED` フラグを解除します。修復されたインタフェースは、管理者が最初に設定したアクティブインタフェースの数に応じて再配備されます。

ベースとなるインタフェースが故障したときに、プローブベースの障害検出が使用されていた場合、`in.mpathd` デーモンは、検査用アドレスが構成されていない場合は指定されたプローバ経由で、またはそのインタフェースの検査用アドレスを使用して、プローブを継続します。インタフェース修復時の回復プロセスは、故障したインタフェースの元の構成に応じて次のように進みます。

- 故障したインタフェースが最初はアクティブインタフェースだった場合、修復されたインタフェースは元のアクティブステータスに戻ります。システム管理者によって定義されたように、十分なインタフェースがその IPMP グループでアクティブになっていれば、故障中に代用品として機能していたスタンバイインタフェースは元のスタンバイステータスに切り替えられます。

  注 – 例外は、修復されたアクティブインタフェースが `FAILBACK=no` モードでも構成されていた場合です。詳細は、86 ページの「`FAILBACK=no` モード」を参照してください

- 故障したインタフェースが最初はスタンバイインタフェースだった場合、IPMP グループにアクティブインタフェースの元の数が反映されていれば、修復されたインタフェースは元のスタンバイステータスに戻ります。それ以外の場合、スタンバイインタフェースはアクティブインタフェースになります。

インタフェースの故障や修復時の IPMP の動作方法のグラフィカル表現を確認するには、74 ページの「IPMP の動作方法」を参照してください。

FAILBACK=no モード

デフォルトでは、故障したあと修復されたアクティブインタフェースは自動的に、IPMP グループ内で元のアクティブインタフェースに戻ります。この動作は、`in.mpathd` デーモンの構成ファイル内の `FAILBACK` パラメータの値によって制御されます。ただし、管理者によっては、データアドレスが修復されたインタフェースに再マッピングされるときに発生する短い中断でも許容できない可能性もあります。そうした管理者は、起動されたスタンバイインタフェースが引き続きアクティブインタフェースとして機能できるようにすることを好む可能性があります。IPMP では、管理者がデフォルト動作をオーバーライドして、インタフェースが修復時に自動的にアクティブにならないようにすることができます。これらのインタフェースは `FAILBACK=no` モードで構成する必要があります。関連する手順については、108 ページの「IPMP デーモンの動作を構成する方法」を参照してください。

`FAILBACK=no` モードのアクティブインタフェースが故障したあと修復された場合、`in.mpathd` デーモンは IPMP の構成を次のように復元します。

- IPMP グループがアクティブインタフェースの元の構成を反映している場合、デーモンはこのインタフェースの `INACTIVE` ステータスを維持します。
- 修復時点での IPMP の構成が、グループのアクティブインタフェースの元の構成を反映していない場合、`FAILBACK=no` ステータスであるにもかかわらず、修復されたインタフェースがアクティブインタフェースとして再配備されます。

注 – `FAILBACK=NO` モードは IPMP グループ全体に対して設定されます。これは、インタフェース単位でチューニング可能なパラメータではありません。

IPMP と動的再構成

Oracle Solaris の動的再構成 (DR) 機能によって、システムの実行中にインタフェースなどのシステムハードウェアを再構成できます。DR は、この機能をサポートするシステムでのみ使用できます。DR をサポートするシステム上では、IPMP は RCM (Reconfiguration Coordination Manager) フレームワークに統合されています。したがって、NIC の接続、切り離し、または再接続を安全に行うことができ、RCM がシステムコンポーネントの動的再構成を管理します。たとえば、新しいインタフェースを接続して plumb したあと、それを既存の IPMP グループに追加できます。これらのインタフェースが構成されると、これらのインタフェースはすぐに IPMP で使用可能となります。

NIC を切断するすべてのリクエストは、まず接続性を保持できるかどうかチェックされます。たとえば、デフォルトでは、IPMP グループ内にない NIC を切断すること

はできません。IPMP グループ内の機能中のインタフェースだけを含む NIC も切断できません。ただし、システムコンポーネントを削除する必要がある場合は、cfgadm(1M) のマニュアルページに説明されている cfgadm コマンドの -f オプションを使用して、この動作をオーバーライドできます。

チェックが成功すると、in.mpathd デーモンはインタフェースに OFFLINE フラグを設定します。インタフェース上の検査用アドレスがすべて構成解除されます。次に、NIC はシステムを unplumb します。これらの手順のいずれかが失敗した場合、または同じシステムコンポーネントのその他のハードウェアの DR で障害が発生した場合は、前の構成が元の状態にリストアされます。このイベントに関するステータスメッセージが表示されます。それ以外の場合、切断要求は正常に完了しています。システムからコンポーネントを削除できます。既存の接続は切断されません。

注 – NIC を交換するときは、同じ種類 (Ethernet など) のカードであることを確認してください。NIC が交換されたら、持続的な IP インタフェース構成がその NIC に適用されます。

◆ ◆ ◆ 第 6 章

IPMP の管理 (タスク)

この章では、Oracle Solaris 11 リリースの IP ネットワークマルチパス (IPMP) でインタフェースグループを管理するためのタスクを紹介します。

この章で扱う内容は、次のとおりです。

注 – この章のタスクでは、`ipadm` コマンドを使用して IPMP を構成する方法について説明します。このコマンドは、Oracle Solaris 10 以前のリリースで使用される `ifconfig` コマンドに代わるものです。これらの 2 つのコマンドが相互にマップする方法の詳細は、『Oracle Solaris 11.1 での固定ネットワーク構成を使用したシステムの接続』の付録 A「比較マップ: ifconfig コマンドと ipadm コマンド」を参照してください。

Oracle Solaris 11 には、IPMP の新しい概念モデルも導入されています。詳細は、74 ページの「IPMP の動作方法」を参照してください。

IPMP の配備時にルーティングを維持

IPMP グループの構成時に、IPMP インタフェースは、ベースとなるインタフェースの IP アドレスを継承して、これらのアドレスをデータアドレスとして使用します。次に、ベースとなるインタフェースは、IP アドレス `0.0.0.0` を受信します。そのため、これらの IP インタフェースがそのあとで IPMP グループに追加された場合、特定のインタフェースを使用して定義されたルートは失われます。

IPMP の構成時にルーティングが失われる場合は通常、デフォルトルートが関与し、Oracle Solaris のインストールに関連して発生します。インストール時に、プライ

マリインタフェースなど、システムでインタフェースを使用する対象となるデフォルトルートを定義する必要があります。そのあとで、デフォルトルートを定義した同じインタフェースを使用して IPMP グループを構成します。IPMP の構成後に、インタフェースのアドレスは IPMP インタフェースに転送されたため、システムはネットワークパケットをルーティングできなくなります。

IPMP の使用中にデフォルトルートが保持されるようにするには、インタフェースを指定せずにルートを定義する必要があります。この方法で、IPMP インタフェースを含む任意のインタフェースをルーティングに使用できます。そのため、システムはトラフィックのルーティングを続行できます。

注 – このセクションでは、デフォルトルートが定義されているプライマリインタフェースを例として使用します。ただし、ルーティングが失われる事例は、ルーティングに使用され、あとで IPMP グループの一部になるすべてのインタフェースに適用されます。

▼ IPMP の使用時にルートを定義する方法

次の手順では、IPMP の構成時にデフォルトルートを保持する方法について説明します。

1. コンソールを使用してシステムにログインします。

   この手順を実行するには、コンソールを使用する必要があります。`ssh` または `telnet` コマンドを使用してログインする場合、後続の手順の実行時に接続は失われます。

2. (省略可能) ルーティングテーブルに定義されたルートを表示します。

   ```
   # netstat -nr
   ```

3. 特定のインタフェースにバインドされているルートを削除します。

   ```
   # route -p delete default gateway-address -ifp interface
   ```

4. インタフェースを指定せずにルートを追加します。

   ```
   # route -p add default gateway-address
   ```

5. (省略可能) 再定義されたルートを表示します。

   ```
   # netstat -nr
   ```

6. (省略可能) ルーティングテーブルで情報が変更されていない場合、ルーティングサービスを再起動し、ルーティングテーブル内の情報を再度調べて、ルートが正しく再定義されていることを確認します。

   ```
   # svcadm restart routing-setup
   ```

例 6–1 IPMP のルートの定義

この例では、インストール時に net0 のデフォルトルートが定義されたことを前提としています。

```
# netstat -nr
Routing Table: IPv4
 Destination     Gateway       Flags    Ref     Use       Interface
-------------  ------------  --------  -----  -----------  --------
default        10.153.125.1     UG      107    176682262   net0
10.153.125.0   10.153.125.222   U        22    137738792   net0

# route -p delete default 10.153.125.1 -ifp net0
# route -p add default 10.153.125.1

# netstat -nr
Routing Table: IPv4
 Destination     Gateway       Flags    Ref     Use       Interface
-------------  ------------  --------  -----  -----------  --------
default        10.153.125.1     UG      107    176682262
10.153.125.0   10.153.125.222   U        22    137738792   net0
```

IPMP グループの構成

このセクションでは、IPMP グループの計画および構成の手順について説明します。第 5 章「IPMP の概要」の概要では、IPMP グループの実装をインタフェースとして説明しています。したがって、この章では、用語「*IPMP* グループ」と「*IPMP* インタフェース」が同義として使用されます。

- 91 ページの「IPMP グループの計画を立てる方法」
- 94 ページの「DHCP を使用して IPMP グループを構成する方法」
- 96 ページの「アクティブ - アクティブ IPMP グループを手動で構成する方法」
- 97 ページの「アクティブ - スタンバイ IPMP グループを手動で構成する方法」

▼ IPMP グループの計画を立てる方法

次の手順には、必要となる計画タスクと IPMP グループを構成する前に収集する情報が含まれています。これらのタスクは、順番どおり行う必要はありません。

注 – 各サブネットまたは L2 ブロードキャストドメインに対して、IPMP グループを 1 つだけ構成する必要があります。詳細は、71 ページの「IPMP を使用するための規則」を参照してください。

---

1 ユーザーの要求に適した全般的な **IPMP** 構成を決定します。

IPMP の構成は、システム上でホストされるタイプのトラフィックを処理するためのネットワークの要件によって決まります。IPMP はアウトバウンドのネットワークパケットを IPMP グループのインタフェース間で分散するため、ネットワークのス

ループットを改善します。ただし、ある特定の TCP 接続では、インバウンドトラフィックは通常、アウトオブオーダーのパケットを処理するリスクを最小限に抑えるために、1 つの物理パスのみをたどります。

したがって、ネットワークが大量のアウトバウンドトラフィックを処理する場合、多数のインタフェースを 1 つの IPMP グループに構成すると、ネットワークのパフォーマンスを改善できます。代わりに、システムが大量のインバウンドトラフィックをホストする場合、グループ内のインタフェースの数を増やしても、トラフィックの負荷分散によってパフォーマンスが改善されるとはかぎりません。ただし、ベースとなるインタフェースの数を増やせば、インタフェースが故障した際のネットワークの可用性を保証しやすくなります。

2 グループ内の各インタフェースが一意の **MAC** アドレスを持っていることを確認します。

システム上のインタフェースごとに一意の MAC アドレスを構成するには、『Oracle Solaris 11.1 での固定ネットワーク構成を使用したシステムの接続』の「各インタフェースの MAC アドレスが一意であることを確認する方法」を参照してください。

3 **STREAMS** モジュールの同じセットが構成され、**IPMP** グループ内のすべてのインタフェースにプッシュされていることを確認します。

同じグループのすべてのインタフェースは、同じ順番で構成された STREAMS モジュールを持っていなければなりません。

a. 予想される **IPMP** グループのすべてのインタフェースの **STREAMS** モジュールの順番を確認します。

`ifconfig` *interface* `modlist` コマンドを使用して、STREAMS モジュールの一覧を出力できます。たとえば、`net0` インタフェースの `ifconfig` 出力は次のようになります。

```
# ifconfig net0 modlist
    0 arp
    1 ip
    2 e1000g
```

この出力が示しているように、インタフェースは通常、IP モジュールのすぐ下のネットワークドライバとして存在します。これらのインタフェースでは、追加の構成は必要ありません。

ただし、特定のテクノロジは、IP モジュールとネットワークドライバの間に STREAMS モジュールとしてプッシュされます。STREAMS モジュールがステートフルである場合、グループ内のすべてのインタフェースに同じモジュールをプッシュしている場合でも、フェイルオーバーで予期しない動作が発生する可能性があります。ただし、IPMP グループのすべてのインタフェースに同じ順番で転送している場合は、処理状態を把握できない STREAMS モジュールを使用できます。

b. 各インタフェースのモジュールを IPMP グループでの標準的な順番でプッシュします。

例:

```
# ifconfig net0 modinsert vpnmod@3
```

4 IPMP グループのすべてのインタフェースで同じ IP アドレス指定形式を使用します。

1 つのインタフェースが IPv4 向けに構成されている場合は、IPMP グループのすべてのインタフェースを IPv4 向けに構成する必要があります。たとえば、1 つのインタフェースに IPv6 アドレス指定を追加すると、IPMP グループ内のすべてのインタフェースを IPv6 をサポートするように構成する必要があります。

5 実装する障害検出のタイプを決定します。

たとえば、プローブベースの障害検出を実装する場合は、ベースとなるインタフェースで検査用アドレスを構成する必要があります。関連情報については、82 ページの「IPMP での障害検出」を参照してください。

6 IPMP グループ内のすべてのインタフェースが同じローカルネットワークに接続されていることを確認します。

たとえば、同じ IP サブネット上の Ethernet スイッチを 1 つの IPMP グループに構成できます。1 つの IPMP グループにいくつでもインタフェースを構成できます。

注 – たとえば、システムに物理インタフェースが 1 つだけ存在する場合は、単一インタフェースの IPMP グループを構成することもできます。関連情報については、73 ページの「IPMP インタフェース構成のタイプ」を参照してください。

---

7 IPMP グループに、別のネットワークメディアタイプのインタフェースが含まれていないことを確認します。

グループ化するインタフェースは、同じインタフェースタイプである必要があります。たとえば、1 つの IPMP グループに Ethernet インタフェースとトークンリングインタフェースを組み合わせることはできません。別の例としては、同じ IPMP グループに、トークンバスインタフェースと非同期転送モード (ATM) インタフェースを組み合わせることはできません。

8 ATM インタフェースを持つ IPMP の場合は、LAN エミュレーションモードで ATM インタフェースを構成します。

IETF RFC 1577 (`http://datatracker.ietf.org/doc/rfc1577/`) および IETF RFC 2225 (`http://datatracker.ietf.org/doc/rfc2225/`) で定義されているクラシカル IP over ATM テクノロジを使用するインタフェースでは、IPMP はサポートされていません。

▼ DHCP を使用して IPMP グループを構成する方法

複数インタフェースの IPMP グループは、アクティブ - アクティブインタフェースで構成することも、アクティブ - スタンバイインタフェースで構成することもできます。関連情報については、73 ページの「IPMP インタフェース構成のタイプ」を参照してください。次の手順では、DHCP を使用してアクティブ - スタンバイ IPMP グループを構成する方法について説明します。

始める前に　想定 IPMP グループに含める予定の IP インタフェースが、システムのネットワークデータリンク上で正しく構成されていることを確認します。リンクおよび IP インタフェースの構成手順については、『Oracle Solaris 11.1 での固定ネットワーク構成を使用したシステムの接続』の「IP インタフェースを構成する方法」を参照してください。IPMP インタフェースは、ベースとなる IP インタフェースがまだ作成されていなくても作成できます。ただし、ベースとなる IP インタフェースが作成されていない場合、この IPMP インタフェースでのその後の構成は失敗します。

さらに、SPARC ベースのシステムを使用する場合は、インタフェースごとに一意の MAC アドレスを構成します。手順については、『Oracle Solaris 11.1 での固定ネットワーク構成を使用したシステムの接続』の「各インタフェースの MAC アドレスが一意であることを確認する方法」を参照してください。

最後に、DHCP を使用する場合は、ベースとなるインタフェースのリースが無限になっていることを確認します。それ以外の場合、IPMP グループの障害が発生した場合に検査用アドレスが期限切れとなり、`in.mpathd` デーモンがプローブベースの障害検出を無効化し、リンクベースの障害検出が使用されます。リンクベースの障害検出によってインタフェースが機能していることがわかると、デーモンは誤ってインタフェースが修復されたと報告する可能性があります。DHCP の構成方法の詳細は、『Oracle Solaris 11.1 での DHCP の作業』を参照してください。

1. 管理者になります。

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

2. **IPMP インタフェースを作成します。**

   ```
   # ipadm create-ipmp ipmp-interface
   ```

   ここで、*ipmp-interface* は IPMP インタフェースの名前を指定します。IPMP インタフェースには、意味のある任意の名前を割り当てることができます。IP インタフェースと同様に、名前は `ipmp0` のように文字列と数字から構成されます。

3. **ベースとなる IP インタフェースがまだ存在しない場合はそれらを作成します。**

   ```
   # ipadm create-ip under-interface
   ```

   ここで、*under-interface* は、IPMP グループに追加する IP インタフェースを表します。

4 検査用アドレスを含むベースとなる **IP** インタフェースを、**IPMP** グループに追加します。

```
# ipadm add-ipmp -i under-interface1 [-i under-interface2 ...] ipmp-interface
```

IP インタフェースは、システムで使用可能な数だけ IPMP グループに追加できます。

5 **IPMP** インタフェース上のデータアドレスを **DHCP** に構成および管理させます。

```
# ipadm create-addr -T dhcp ipmp-interface
```

手順 5 は、DHCP サーバーによって提供されたアドレスをアドレスオブジェクトに関連付けます。アドレスオブジェクトは、たとえば `ipmp0/v4` のように、*interface*/*address-type* という形式を使用して IP アドレスを一意に識別します。アドレスオブジェクトの詳細は、『Oracle Solaris 11.1 での固定ネットワーク構成を使用したシステムの接続』の「IP インタフェースを構成する方法」を参照してください

6 検査用アドレスによるプローブベースの障害検出を使用する場合は、ベースとなるインタフェース上の検査用アドレスを **DHCP** に管理させます。

IPMP グループのベースとなるインタフェースごとに、次のコマンドを発行します。

```
# ipadm create-addr -T dhcp under-interface
```

手順 6 で自動的に作成されるアドレスオブジェクトは、たとえば `net0/v4` のように、*under-interface*/*address-type* という形式を使用します。

例 6–2 DHCP を使用した IPMP グループの構成

この例は、DHCP を使用してアクティブ - スタンバイ IPMP グループを構成する方法を示しますが、次のシナリオに基づいています。

- 3 つのベースとなるインタフェース `net0`、`net1`、および `net2` が IPMP グループに構成されます。
- IPMP インタフェース `ipmp0` は、同じ名前を IPMP グループと共有します。
- `net2` が指定されたスタンバイインタフェースになります。
- ベースとなるインタフェースのすべてに検査用アドレスが割り当てられます。

まず、管理者は IPMP インタフェースを作成します。

```
# ipadm create-ipmp ipmp0
```

次に、管理者はベースとなる IP インタフェースを作成し、それらを IPMP インタフェースに追加します。

```
# ipadm create-ip net0
# ipadm create-ip net1
# ipadm create-ip net2

# ipadm add-ipmp -i net0 -i net1 -i net2 ipmp0
```

次に、管理者は DHCP によって管理される IP アドレスを IPMP インタフェースに割り当てます。IPMP インタフェースに割り当てられる IP アドレスはデータアドレスです。この例では、IPMP インタフェースには 2 つのデータアドレスがあります。

```
# ipadm create-addr -T dhcp ipmp0
ipadm: ipmp0/v4
# ipadm create-addr -T dhcp ipmp0
ipadm: ipmp0/v4a
```

次に、管理者は DHCP によって管理される IP アドレスを、IPMP グループのベースとなる IP インタフェースに割り当てます。ベースとなるインタフェースに割り当てられる IP アドレスは、プローブベースの障害検出に使用される検査用アドレスです。

```
# ipadm create-addr -T dhcp net0
ipadm: net0/v4
# ipadm create-addr -T dhcp net1
ipadm: net1/v4
# ipadm create-addr -T dhcp net2
ipadm net2/v4
```

最後に、管理者は net2 がスタンバイインタフェースになるように構成します。

```
# ipadm set-ifprop -p standby=on net2
```

▼ アクティブ-アクティブ IPMP グループを手動で構成する方法

次の手順では、アクティブ - アクティブ IPMP グループを手動で構成する方法について説明します。この手順の手順 1 から 4 では、リンクベースのアクティブ - アクティブ IPMP グループを構成する方法について説明します。手順 5 では、リンクベースの構成をプローブベースにする方法について説明します。

始める前に　想定 IPMP グループに含める予定の IP インタフェースが、システムのネットワークデータリンク上で正しく構成されていることを確認します。リンクおよび IP インタフェースの構成手順については、『Oracle Solaris 11.1 での固定ネットワーク構成を使用したシステムの接続』の「IP インタフェースを構成する方法」を参照してください。IPMP インタフェースは、ベースとなる IP インタフェースがまだ存在していなくても作成できます。ただし、この IPMP インタフェースでのその後の構成は失敗します。

さらに、SPARC ベースのシステムを使用する場合は、インタフェースごとに一意の MAC アドレスを構成します。手順については、『Oracle Solaris 11.1 での固定ネットワーク構成を使用したシステムの接続』の「各インタフェースの MAC アドレスが一意であることを確認する方法」を参照してください。

1 管理者になります。

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

2 IPMP インタフェースを作成します。

```
# ipadm create-ipmp ipmp-interface
```

ここで、*ipmp-interface* は IPMP インタフェースの名前を指定します。IPMP インタフェースには、意味のある任意の名前を割り当てることができます。IP インタフェースと同様に、名前は `ipmp0` のように文字列と数字から構成されます。

3 ベースとなる IP インタフェースをグループに追加します。

```
# ipadm add-ipmp -i under-interface1 [-i underinterface2 ...] ipmp-interface
```

ここで、*under-interface* は、IPMP グループのベースとなるインタフェースを表します。IP インタフェースは、システムで使用可能な数だけ追加できます。

注 – デュアルスタック環境では、特定のグループにインタフェースの IPv4 インスタンスを配置すると、IPv6 インスタンスが自動的に同じグループに配置されます。

---

4 IPMP インタフェースにデータアドレスを追加します。

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

ここで、*address* は CIDR 表記にすることができます。

注 – IPMP グループ名または IP アドレスの DNS アドレスのみが必要です。

---

5 検査用アドレスによるプローブベースの障害検出を使用する場合は、ベースとなるインタフェースに検査用アドレスを追加します。

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

ここで、*address* は CIDR 表記にすることができます。IPMP グループのすべての検査用 IP アドレスは、1 つの IP サブネットに属していなければならず、したがって同じネットワーク接頭辞を使用する必要があります。

▼ アクティブ-スタンバイ IPMP グループを手動で構成する方法

スタンバイインタフェースについては、73 ページの「IPMP インタフェース構成のタイプ」を参照してください。次の手順では、1 つのインタフェースがスタンバイとして確保される IPMP グループを構成する方法を説明します。このインタフェースは、グループ内のアクティブインタフェースが故障した場合にのみ配備されます。

1. 管理者になります。

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

2. **IPMP インタフェースを作成します。**

   ```
   # ipadm create-ipmp ipmp-interface
   ```

   ここで、*ipmp-interface* は IPMP インタフェースの名前を指定します。IPMP インタフェースには、意味のある任意の名前を割り当てることができます。IP インタフェースと同様に、名前は `ipmp0` のように文字列と数字から構成されます。

3. **ベースとなる IP インタフェースをグループに追加します。**

   ```
   # ipadm add-ipmp -i under-interface1 [-i underinterface2 ...] ipmp-interface
   ```

   ここで、*under-interface* は、IPMP グループのベースとなるインタフェースを表します。IP インタフェースは、システムで使用可能な数だけ追加できます。

   注 – デュアルスタック環境では、特定の IPMP グループにインタフェースの IPv4 インスタンスを配置すると、IPv6 インスタンスが自動的に同じグループに配置されます。

   ---

4. **IPMP インタフェースにデータアドレスを追加します。**

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

   ここで、*address* は CIDR 表記にすることができます。

5. **検査用アドレスによるプローブベースの障害検出を使用する場合は、ベースとなるインタフェースに検査用アドレスを追加します。**

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

   ここで、*address* は CIDR 表記にすることができます。IPMP グループのすべての検査用 IP アドレスは、1 つの IP サブネットに属していなければならず、したがって同じネットワーク接頭辞を使用する必要があります。

6. **ベースとなるインタフェースの 1 つをスタンバイインタフェースとして構成します。**

   ```
   # ipadm set-ifprop -p standby=on under-interface
   ```

例 6–3 アクティブ - スタンバイ IPMP グループの構成

この例は、アクティブ - スタンバイ IPMP 構成を手動で作成する方法を示します。

まず、管理者は IPMP インタフェースを作成します。

```
# ipadm create-ipmp ipmp0
```

次に、管理者はベースとなる IP インタフェースを作成し、それらを IPMP インタフェースに追加します。

```
# ipadm create-ip net0
# ipadm create-ip net1
# ipadm create-ip net2

# ipadm add-ipmp -i net0 -i net1 -i net2 ipmp0
```

次に、管理者は IPMP インタフェースに IP アドレスを割り当てます。IPMP インタフェースに割り当てられる IP アドレスはデータアドレスです。この例では、IPMP インタフェースには 2 つのデータアドレスがあります。

```
# ipadm create-addr -a 192.168.10.10/24 ipmp0
ipadm: ipmp0/v4
# ipadm create-addr -a 192.168.10.15/24 ipmp0
ipadm: ipmp0/v4a
```

この例の IP アドレスには、10 進数で表される `prefixlen` プロパティーが含まれています。IP アドレスの `prefixlen` 部分は、アドレスの左から何ビットがアドレスの IPv4 ネットマスクまたは IPv6 接頭部に該当するかを指定します。残りの下位ビットは、アドレスのホスト部を定義します。`prefixlen` プロパティーをアドレスのテキスト表現に変換すると、アドレスのネットワーク部として使用するビット位置には 1 が、ホスト部として使用するビット位置には 0 が含まれています。このプロパティーは、`dhcp` アドレスオブジェクトタイプではサポートされません。詳細は、ipadm(1M) のマニュアルページを参照してください。

次に、管理者は IPMP グループのベースとなる IP インタフェースに IP アドレスを割り当てます。ベースとなるインタフェースに割り当てられる IP アドレスは、プローブベースの障害検出に使用される検査用アドレスです。

```
# ipadm create-addr -a 192.168.10.30/24 net0
ipadm: net0/v4
# ipadm create-addr -a 192.168.10.32/24 net1
ipadm: net1/v4
# ipadm create-addr -a 192.168.10.34/24 net2
ipadm: net2/v4
```

最後に、管理者は `net2` がスタンバイインタフェースになるように構成します。

```
# ipadm set-ifprop -p standby=on net2
```

管理者は `ipmpstat` コマンドを使用して IPMP 構成を表示できます。

```
# ipmpstat -g
GROUP     GROUPNAME   STATE      FDT        INTERFACES
ipmp0     ipmp0       ok         10.00s     net0 net1 (net2)

# ipmpstat -t
INTERFACE  MODE     TESTADDR         TARGETS
net0       routes   192.168.10.30    192.168.10.1
```

```
net1        routes    192.168.10.32   192.168.10.1
net2        routes    192.168.10.34   192.168.10.5
```

IPMP の保守

このセクションでは、システム上に作成した IPMP グループを保守する手順について説明します。

- 100 ページの「IPMP グループにインタフェースを追加する方法」
- 101 ページの「IPMP グループからインタフェースを削除する方法」
- 101 ページの「IP アドレスを追加する方法」
- 102 ページの「IP アドレスを削除する方法 」
- 103 ページの「インタフェースを 1 つの IPMP グループから別の IPMP グループに移動する方法」
- 104 ページの「IPMP グループを削除する方法」

▼ IPMP グループにインタフェースを追加する方法

始める前に　グループに追加するインタフェースが、すべての要件を満たしていることを確認します。要件の一覧については、91 ページの「IPMP グループの計画を立てる方法」を参照してください。

1. 管理者になります。

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

2. ベースとなる IP インタフェースがまだ存在していない場合は、インタフェースを作成します。

   ```
   # ipadm create-ip under-interface
   ```

3. IPMP グループへ IP インタフェースを追加します。

   ```
   # ipadm add-ipmp -i under-interface ipmp-interface
   ```

   ここで、*ipmp-interface* はベースとなるインタフェースを追加する IPMP グループを表します。

例 6–4　IPMP グループへのインタフェースの追加

次の例では、インタフェース `net4` を IPMP グループ `ipmp0` に追加します。

```
# ipadm create-ip net4
# ipadm add-ipmp -i net4 ipmp0
# ipmpstat -g
GROUP    GROUPNAME   STATE      FDT       INTERFACES
ipmp0    ipmp0       ok         10.00s    net0 net1 net4
```

▼ IPMP グループからインタフェースを削除する方法

1 管理者になります。

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

2 IPMP グループから 1 つ以上のインタフェースを削除します。

```
# ipadm remove-ipmp -i under-interface[ -i under-interface ...] ipmp-interface
```

ここで、*under-interface* は IPMP グループから削除する IP インタフェースを表し、*ipmp-interface* はベースとなるインタフェースを削除する IPMP グループを表します。

単一のコマンド内で、ベースとなるインタフェースを必要な数だけ削除できます。ベースとなるインタフェースをすべて削除しても、IPMP インタフェースは削除されません。代わりに、空の IPMP インタフェースまたはグループとして存在します。

例 6–5 IPMP グループからのインタフェースの削除

次の例では、インタフェース `net4` を IPMP グループ `ipmp0` から削除します。

```
# ipadm remove-ipmp net4 ipmp0
# ipmpstat -g
GROUP    GROUPNAME   STATE      FDT       INTERFACES
ipmp0    ipmp0       ok         10.00s    net0 net1
```

▼ IP アドレスを追加する方法

IP アドレスを追加するには、`ipadm create-addr` サブコマンドを使用します。IPMP では、IP アドレスはデータアドレスまたは検査用アドレスのどちらかです。データアドレスは IPMP インタフェースに追加されます。検査用アドレスは、IPMP インタフェースのベースとなるインタフェースに追加されます。次の手順では、検査用アドレスとデータアドレスのいずれかである IP アドレスを追加する方法について説明します。

1 管理者になります。

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

2 IPMP グループにデータアドレスを追加するには、次のコマンドを入力します。

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

IP アドレスを作成すると、アドレスオブジェクトが自動的に割り当てられます。アドレスオブジェクトは、IP アドレスの一意識別子です。アドレスオブジェクトの名前には、*interface*/*random-string* という命名規則が使用されます。したがって、データアドレスのアドレスオブジェクトの名前には、IPMP インタフェースが含まれることになります。

3 **IPMP グループのベースとなるインタフェースに検査用アドレスを追加するには、次のコマンドを入力します。**

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

IP アドレスを作成すると、アドレスオブジェクトが自動的に割り当てられます。アドレスオブジェクトは、IP アドレスの一意識別子です。アドレスオブジェクトの名前には、*interface*/*random-string* という命名規則が使用されます。したがって、検査用アドレスのアドレスオブジェクトの名前には、ベースとなるインタフェースが含まれることになります。

▼ IP アドレスを削除する方法

IP アドレスを削除するには、`ipadm delete-addr` サブコマンドを使用します。IPMP では、データアドレスは IPMP インタフェースでホストされ、検査用アドレスはベースとなるインタフェースでホストされます。次の手順は、データアドレスと検査用アドレスのいずれかである IP アドレスを削除する方法を示しています。

1 管理者になります。

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

2 削除するアドレスを決定します。

- データアドレスの一覧を表示するには、次のコマンドを入力します。

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

- 検査用アドレスの一覧を表示するには、次のコマンドを入力します。

  ```
  # ipadm show-addr
  ```

  検査用アドレスを識別するアドレスオブジェクトの名前には、アドレスが構成されているベースとなるインタフェースが含まれています。

3 **IPMP グループからデータアドレスを削除するには、次のコマンドを入力します。**

```
# ipadm delete-addr addrobj
```

ここで、*addrobj* には IPMP インタフェースの名前が含まれている必要があります。入力したアドレスオブジェクトに IPMP インタフェース名が含まれていない場合、削除されるアドレスはデータアドレスではありません。

4 IPMP グループから検査用アドレスを削除するには、次のコマンドを入力します。

```
# ipadm delete-addr addrobj
```

ここで、正しい検査用アドレスを削除するには、*addrobj* にベースとなる正しいインタフェースの名前が含まれている必要があります。

例 6–6 インタフェースからの検査用アドレスの削除

次の例は、例 6–3 のアクティブ - スタンバイ IPMP グループ `ipmp0` の構成を使用しています。この例では、ベースとなるインタフェース `net1` から検査用アドレスを削除します。

```
# ipadm show-addr net1
ADDROBJ           TYPE      STATE     ADDR
net1/v4           static    ok        192.168.10.30

# ipadm delete-addr net1/v4
```

▼ インタフェースを 1 つの IPMP グループから別の IPMP グループに移動する方法

インタフェースが既存の IPMP グループに属している場合は、新しい IPMP グループにインタフェースを配置できます。この場合、現在の IPMP グループからインタフェースを削除する必要はありません。新しいグループに追加されたインタフェースは、既存の IPMP グループから自動的に削除されます。

1 管理者になります。

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

2 新しい IPMP グループへインタフェースを移動します。

```
# ipadm add-ipmp -i under-interface ipmp-interface
```

ここで、*under-interface* は移動するベースとなるインタフェースを表し、*ipmp-interface* はベースとなるインタフェースの移動先となる IPMP インタフェースを表します。

例 6–7 別の IPMP グループへのインタフェースの移動

この例で、IPMP グループのベースとなるインタフェースは `net0`、`net1`、および `net2` です。次のコマンドは、`net0` インタフェースを IPMP グループ `ipmp0` から削除したあと、`net0` を IPMP グループ `cs-link1` に配置します。

```
# ipadm add-ipmp -i net0 ca-link1
```

▼ IPMP グループを削除する方法

この手順は、特定の IPMP グループが不要になったときに使用します。

1 管理者になります。

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

2 削除する **IPMP** グループおよびベースとなる **IP** インタフェースを特定します。

```
# ipmpstat -g
```

3 **IPMP** グループに現在属している **IP** インタフェースをすべて削除します。

```
# ipadm remove-ipmp -i under-interface[, -i under-interface, ...] ipmp-interface
```

ここで、*under-interface* は削除するベースとなるインタフェースを表し、*ipmp-interface* はベースとなるインタフェースを削除する IPMP インタフェースを表します。

注 – IPMP インタフェースを正常に削除するには、その IPMP グループの一部として IP インタフェースが存在していてはいけません。

---

4 **IPMP** インタフェースを削除します。

```
# ipadm delete-ipmp ipmp-interface
```

IPMP インタフェースを削除すると、このインタフェースに関連付けられた IP アドレスはすべてシステムから削除されます。

例 6–8 IPMP インタフェースの削除

次の例では、ベースとなる IP インタフェース `net0` および `net1` を含むインタフェース `ipmp0` を削除します。

```
# ipmpstat -g
GROUP   GROUPNAME   STATE      FDT        INTERFACES
ipmp0   ipmp0       ok         10.00s     net0 net1

# ipadm remove-ipmp -i net0 -i net1 ipmp0

# ipadm delete-ipmp ipmp0
```

プローブベースの障害検出の構成

プローブベースの障害検出では、82 ページの「プローブベースの障害検出」で説明されているようにターゲットシステムを使用します。プローブベースの障害検出のターゲットを特定するときに、`in.mpathd` デーモンはルーターターゲットモード、マルチキャストターゲットモードの 2 つのモードで動作します。ルーターターゲットモードでは、デーモンはルーティングテーブルに定義されたターゲットをプローブします。ターゲットが 1 つも定義されていない場合、このデーモンはマルチキャストターゲットモードで動作します。この場合、LAN 上の近くのホストをプローブするためにマルチキャストパケットが送出されます。

できれば、`in.mpathd` デーモンがプローブするターゲットシステムを設定するようにしてください。一部の IPMP グループでは、デフォルトルーターはターゲットとして十分です。ただし、一部の IPMP グループでは、プローブベースの障害検出用に特定のターゲットを構成したほうが良いこともあります。ターゲットを指定するには、ルーティングテーブル内にホストのルートをプローブターゲットとして設定します。ルーティングテーブルに構成されているすべてのホストルートは、デフォルトルーターの前に一覧化されます。IPMP はターゲットを選択するために、明示的に定義されたホストルートを使用します。したがって、デフォルトルーターを使用するのではなく、ホストのルートを設定して特定のプローブターゲットを構成するようにしてください。

ホストのルートをルーティングテーブルに設定するには、`route` コマンドを使用します。このコマンドで `-p` オプションを使用して、永続的なルートを追加できます。たとえば、`route -p add` はシステムのリブート後もルーティングテーブル内に残るルートを追加します。したがって、`-p` オプションを使用すると、システムが起動するたびにこれらのルートを作成し直す特殊なスクリプトを一切必要とせずに、持続的なルートを追加できます。プローブベースの障害検出を最適なかたちで使用するには、プローブを受信するターゲットを必ず複数設定してください。

107 ページの「プローブベースの障害検出のターゲットシステムを手動で指定する方法」の手順は、プローブベースの障害検出でのターゲットへの持続的なルートを追加するための正確な構文を示しています。`route` コマンドのオプションの詳細については、`route(1M)` のマニュアルページを参照してください。

このセクションの内容は次のとおりです。

プローブベースの障害検出のターゲットを選択するための要件

ネットワーク上のどのホストが適切なターゲットになるのかの評価では、この基準リストに従ってください:

- 予想されるターゲットが使用可能で、実行されていることを確認します。IP アドレスの一覧を作成します。

- ターゲットインタフェースが、構成中の IPMP グループと同じネットワークにあることを確認します。
- ターゲットシステムのネットマスクとブロードキャストアドレスは、IPMP グループ内のアドレスと同じでなければなりません。
- ターゲットホストは、プローブベースの障害検出を使用しているインタフェースからの ICMP 要求に応答できなければなりません。

プローブベースの障害検出の構成 (タスクマップ)

次のタスクマップは、IPMP グループのプローブベースの障害検出を構成する手順を示しています。

タスク	説明	参照先
プローブベースの障害検出の動作を選択します。	プローブベースの障害検出で推移的プローブまたは検査用アドレスを使用するかどうかを決定します。	106 ページの「使用する障害検出手法を選択する方法」
プローブベースの障害検出に使用するターゲットシステムを選択します。	IPMP グループのベースとなるインタフェースのステータスを検査するためにプローブベースの障害検出で使用する、システムの IP アドレスを指定します。	107 ページの「プローブベースの障害検出のターゲットシステムを手動で指定する方法」
修復されたインタフェースを IPMP グループに再配備する方法を選択します。	修復されたインタフェースを IPMP グループ内で自動的に再アクティブ化するか、非アクティブなままにするかを決定します。	108 ページの「IPMP デーモンの動作を構成する方法」

▼ 使用する障害検出手法を選択する方法

デフォルトでは、プローブベースの障害検出は検査用アドレスを使用して動作します。NIC ドライバがリンクベースの障害検出をサポートしている場合、この障害検出も自動的に有効になります。

NIC ドライバによってリンクベースの障害検出がサポートされている場合、この手法を無効にすることはできません。ただし、実装するプローブベースの障害検出のタイプは選択可能です。

始める前に　105 ページの「プローブベースの障害検出のターゲットを選択するための要件」に示された要件をプローブのターゲットが満たしていることを確認してください。

1. 推移的プローブのみを使用する場合は、次の手順を実行します。

   a. 適切な SMF コマンドを使用して IPMP のプロパティー **transitive-probing** をオンに切り替えます。

   ```
   # svccfg -s svc:/network/ipmp setprop config/transitive-probing=true
   # svcadm refresh  svc:/network/ipmp:default
   ```

   このプロパティーの設定方法の詳細については、in.mpathd(1M) のマニュアルページを参照してください。

   b. IPMP グループ用に構成された既存の検査用アドレスをすべて削除します。

   ```
   # ipadm delete-addr address addrobj
   ```

   ここで、*addrobj* は検査用アドレスをホストしているベースとなるインタフェースを表す必要があります。

2. 検査用アドレスを使用して障害をプローブする場合は、次の手順を実行します。

   a. 必要に応じて、推移的プローブをオフにします。

   ```
   # svccfg -s svc:/network/ipmp setprop config/transitive-probing=false
   # svcadm refresh  svc:/network/ipmp:default
   ```

   b. IPMP グループのベースとなるインタフェースに検査用アドレスを割り当てます。

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

   ここで、*address* は CIDR 表記にすることができ、*under-interface* は IPMP グループのベースとなるインタフェースです。

▼ プローブベースの障害検出のターゲットシステムを手動で指定する方法

この手順では、検査用アドレスを使用してプローブベースの障害検出を実装する場合に、特定のターゲットを追加する方法について説明します。

始める前に　105 ページの「プローブベースの障害検出のターゲットを選択するための要件」に示された要件をプローブのターゲットが満たしていることを確認してください。

1. プローブベースの障害検出を構成するシステムにユーザーアカウントでログインします。

2. プローブベースの障害検出のターゲットとして使用される特定のホストにルートを追加します。

   ```
   $ route -p add -host destination-IP gateway-IP -static
   ```

ここで、*destination-IP* と *gateway-IP* は、ターゲットとして使用されるホストの IPv4 アドレスです。たとえば、IPMP グループ `ipmp0` のインタフェースと同じサブネット上のターゲットシステム `192.168.10.137` を指定するには、次のように入力します。

```
$ route -p add -host 192.168.10.137 192.168.10.137 -static
```

この新しいルートは、システムを再起動するたびに自動的に構成されます。プローブベースの障害検出用のターゲットシステムへの一時的なルートを定義するだけの場合は、`-p` オプションを使用しないでください。

3 ターゲットシステムとして使用されるネットワーク上の追加ホストにルートを追加します。

▼ IPMP デーモンの動作を構成する方法

IPMP グループに関連する次のシステム共通パラメータを設定するには、IPMP 構成ファイル `/etc/default/mpathd` を使用します。

- `FAILURE_DETECTION_TIME`
- `FAILBACK`
- `TRACK_INTERFACES_ONLY_WITH_GROUPS`

1 管理者になります。

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

2 **`/etc/default/mpathd`** ファイルを編集します。

3 つのパラメータの 1 つ以上のデフォルト値を変更します。

a. **`FAILURE_DETECTION_TIME`** パラメータの新しい値を入力します。

```
FAILURE_DETECTION_TIME=n
```

ここで、*n* は ICMP 検証がインタフェースの障害が発生していないかどうかを検出する時間 (秒単位) です。デフォルトは 10 秒です。

b. **`FAILBACK`** パラメータの新しい値を入力します。

```
FAILBACK=[yes | no]
```

- yes – 値 yes が、IPMP のフェイルバック動作のデフォルトです。障害が発生したインタフェースの修復が検出されると、ネットワークアクセスはこの修復されたインタフェースに復帰します。詳細は、85 ページの「物理インタフェースの回復検出」を参照してください。
- no – 値 no は、データトラフィックが修復されたインタフェースに戻らないことを示します。障害が発生したインタフェースの回復が検出されると、そのインタフェースに `INACTIVE` フラグが設定されます。このフラグは、現時点でそ

のインタフェースをデータトラフィックに使用すべきでないことを示します。ただし、そのインタフェースを検査信号トラフィックに使用することはできます。

たとえば、IPMP グループ ipmp0 が 2 つのインタフェース net0 と net1 から構成されているとします。/etc/default/mpathd ファイルで FAILBACK=no パラメータが設定されています。net0 が故障すると、FAILED としてフラグが付けられて使用不可能になります。修復後、このインタフェースは INACTIVE としてフラグが付けられ、値 FAILBACK=no が設定されているため、使用不可能なままとなります。

net1 が故障し、net0 のみが INACTIVE 状態である場合には、net0 の INACTIVE フラグが解除され、このインタフェースが使用可能になります。IPMP グループに同じく INACTIVE 状態のインタフェースがほかにも含まれている場合、net1 の故障時にそれらの INACTIVE インタフェースのいずれか 1 つ (必ずしも net0 とはかぎらない) がクリアーされ、使用可能となります。

c. **TRACK_INTERFACES_ONLY_WITH_GROUPS** パラメータの新しい値を入力します。

```
TRACK_INTERFACES_ONLY_WITH_GROUPS=[yes | no]
```

注 – このパラメータと匿名グループ機能については、85 ページの「障害検出と匿名グループ機能」を参照してください。

- yes – 値 yes が、IPMP の動作のデフォルトです。この値を指定した場合、IPMP は、IPMP グループ内に構成されていないネットワークインタフェースを無視します。
- no - 値 no は、IPMP グループ内に構成されているネットワークインタフェースかどうかにかかわらず、すべてのネットワークインタフェースの障害と修復の検出を設定します。ただし、IPMP グループ内に構成されていないインタフェースで障害や修復が検出されても、IPMP では、そのインタフェースのネットワーク機能を維持するためアクションは一切起動されません。したがって、値 no を指定することは、障害の報告には役立ちますが、ネットワークの可用性を直接向上させることはありません。

3 **in.mpathd** デーモンを再起動します。

```
# pkill -HUP in.mpathd
```

デーモンは、新しいパラメータ値が有効になった状態で再起動します。

IPMP 情報のモニタリング

このセクションの例では、ipmpstat コマンドを使用して、システム上の IPMP グループのさまざまな側面をモニターできるようにします。IPMP グループ全体のステータスやそのベースとなる IP インタフェースのステータスを監視できます。IPMP グループのデータアドレスや検査用アドレスの構成を確認することもできます。ipmpstat コマンドを使用すると、障害検出に関する情報も取得されます。ipmpstat コマンドやそのオプションの詳細については、ipmpstat(1M) のマニュアルページを参照してください。

ipmpstat コマンドを使用する場合はデフォルトで、80 列に収まるもっとも意味のあるフィールドが表示されます。出力では、ipmpstat -p 構文の場合を除き、ipmpstat コマンドで使用したオプションに固有のフィールドがすべて表示されます。

デフォルトでは、ホスト名が存在する場合、数値 IP アドレスではなくホスト名が出力に表示されます。出力に数値 IP アドレスを表示するには、-n オプションを、IPMP グループの特定の情報を表示するためのほかのオプションとともに使用します。

注 – 次の各例では、特に明記していないかぎり、ipmpstat コマンドの使用時にシステム管理者の特権は必要ありません。

表示する情報を決定するには、ipmpstat コマンドに次のオプションを使用します。

- -g は、システム上の IPMP グループに関する情報を表示します。例 6–9 を参照してください。
- -a は、IPMP グループに構成されているデータアドレスを表示します。例 6–10 を参照してください。
- -i は、IPMP 構成に関連する IP インタフェースに関する情報を表示します。例 6–11 を参照してください。
- -t は、障害検出に使用されるターゲットシステムに関する情報を表示します。このオプションは、IPMP グループで使用される検査用アドレスも表示します。例 6–12 を参照してください。
- -p は、障害検出に使用されているプローブに関する情報を表示します。例 6–13 を参照してください。

次の例では、ipmpstat コマンドで取得できるシステムの IPMP 構成に関する情報を示します。

例 6–9　IPMP グループ情報の取得

-g オプションは、システム上のさまざまな IPMP グループのステータスを、そのベースとなるインタフェースのステータスも含めて一覧表示します。ある特定のグループでプローブベースの障害検出が有効になっていると、コマンドはそのグループの障害検出時間も含めます。

例 6–9 IPMP グループ情報の取得 (続き)

```
$ ipmpstat -g
GROUP    GROUPNAME   STATE      FDT        INTERFACES
ipmp0    ipmp0       ok         10.00s     net0 net1
acctg1   acctg1      failed     --         [net3 net4]
field2   field2      degraded   20.00s     net2 net5 (net7) [net6]
```

出力フィールドでは、次の情報が提供されます。

GROUP IPMP インタフェースの名前を指定します。匿名グループの場合、このフィールドは空になります。匿名グループの詳細は、in.mpathd(1M) のマニュアルページを参照してください。

GROUPNAME IPMP グループの名前を指定します。匿名グループの場合、このフィールドは空になります。

STATE IPMP グループの現在のステータスを示します。ステータスは次のいずれかになります。

- ok は、IPMP グループのベースとなるインタフェースがすべて使用可能であることを示します。
- degraded は、グループ内のベースとなるインタフェースの一部が使用不可能であることを示します。
- failed は、グループのインタフェースがすべて使用不可能であることを示します。

FDT 障害検出が有効になっている場合に、その障害検出時間を指定します。障害検出が無効になっている場合、このフィールドは空になります。

INTERFACES IPMP グループに属するベースとなるインタフェースを指定します。このフィールドでは、まずアクティブなインタフェースが表示され、次にアクティブでないインタフェースが表示され、最後に使用不可能なインタフェースが表示されます。インタフェースのステータスはその表示形式によって示されます。

- *interface* (丸括弧や角括弧なし) は、アクティブなインタフェースを示します。アクティブなインタフェースは、システムでデータトラフィックの送受信に使用されています。
- *(interface)* (丸括弧付き) は、機能しているがアクティブではないインタフェースを示します。このインタフェースは、管理ポリシーの定義に従って未使用になっています。
- *[interface]* (角括弧付き) は、インタフェースに障害が発生しているかオフラインになっているために使用不可能であることを示します。

例 6–10 IPMP のデータアドレス情報の取得

-a オプションは、データアドレスと各アドレスの所属先 IPMP グループを表示します。表示される情報には、アドレスが ipadm [up-addr/down-addr] コマンドによって切り替えられたかどうかに基づいてどのアドレスが使用可能であるかも含まれます。また、あるアドレスがどのインバウンドまたはアウトバウンドインタフェース上で使用できるかも判定できます。

```
$ ipmpstat -an
ADDRESS         STATE    GROUP      INBOUND     OUTBOUND
192.168.10.10   up       ipmp0        net0    net0 net1
192.168.10.15   up       ipmp0        net1    net0 net1
192.0.0.100     up       acctg1       --         --
192.0.0.101     up       acctg1       --         --
128.0.0.100     up       field2        net2    net2 net7
128.0.0.101     up       field2        net7    net2 net7
128.0.0.102     down     field2       --         --
```

出力フィールドでは、次の情報が提供されます。

ADDRESS -n オプションが -a オプションとともに使用された場合は、ホスト名またはデータアドレスを示します。

STATE IPMP インタフェースのアドレスが、up したがって使用可能、down したがって使用不可能、のいずれであるかを示します。

GROUP 特定のデータアドレスをホストする IPMP インタフェースを示します。通常、Oracle Solaris では、IPMP グループの名前は IPMP インタフェースです。

INBOUND 特定のアドレスのパケットを受信するインタフェースを識別します。このフィールドの情報は、外部のイベントに応じて変わる可能性があります。たとえば、データアドレスが停止している場合や、IPMP グループ内にアクティブな IP インタフェースが 1 つも残っていない場合、このフィールドは空になります。空のフィールドは、この特定のアドレス宛ての IP パケットをシステムが受け入れていないことを示します。

OUTBOUND 特定のアドレスを発信元アドレスとして使用したパケットを送信するインタフェースを識別します。INBOUND フィールドと同様に、OUTBOUND の情報も外部のイベントに応じて変わる可能性があります。空のフィールドは、システムがこの特定の発信元アドレスでパケットを送信していないことを示します。このフィールドが空である場合、それは、アドレスが停止しているからか、またはグループ内にアクティブな IP インタフェースが 1 つも残っていないからです。

例 6–11 IPMP グループのベースとなる IP インタフェースに関する情報の取得

-i オプションは、IPMP グループのベースとなる IP インタフェースに関する情報を表示します。

例 6–11 IPMP グループのベースとなる IP インタフェースに関する情報の取得 (続き)

```
$ ipmpstat -i
INTERFACE   ACTIVE   GROUP      FLAGS      LINK       PROBE      STATE
net0        yes      ipmp0      --mb---    up         ok         ok
net1        yes      ipmp0      -------    up         disabled   ok
net3        no       acctg1     -------    unknown    disabled   offline
net4        no       acctg1     is-----    down       unknown    failed
net2        yes      field2     --mb---    unknown    ok         ok
net6        no       field2     -i-----    up         ok         ok
net5        no       filed2     -------    up         failed     failed
net7        yes      field2     --mb---    up         ok         ok
```

出力フィールドでは、次の情報が提供されます。

INTERFACE　各 IPMP グループのベースとなる各インタフェースを示します。

ACTIVE　このインタフェースが機能していて使用中 (yes) であるか、それともそうではない (no) かを示します。

GROUP　IPMP インタフェースの名前を指定します。匿名グループの場合、このフィールドは空になります。匿名グループの詳細は、in.mpathd(1M) のマニュアルページを参照してください。

FLAGS　ベースとなる各インタフェースのステータスを示し、これは次のいずれか、またはその任意の組み合わせになります。

- i は、このインタフェースに INACTIVE フラグが設定されていることを示します。したがって、このインタフェースはデータトラフィックの送受信に使用されません。
- s は、このインタフェースがスタンバイインタフェースとして構成されていることを示します。
- m は、このインタフェースがこの IPMP グループの IPv4 マルチキャストトラフィックの送受信用として、システムによって指定されていることを示します。
- b は、このインタフェースがこの IPMP グループのブロードキャストトラフィックの受信用として、システムによって指定されていることを示します。
- M は、このインタフェースがこの IPMP グループの IPv6 マルチキャストトラフィックの送受信用として、システムによって指定されていることを示します。
- d は、このインタフェースが停止しており、したがって使用不可能であることを示します。
- h は、このインタフェースが重複する物理ハードウェアアドレスを別のインタフェースと共有しており、オフラインになっていることを示します。h フラグは、このインタフェースが使用不可能であることを示します。

例 6–11 IPMP グループのベースとなる IP インタフェースに関する情報の取得 (続き)

LINK リンクベースの障害検出のステータスを示し、これは次のいずれかになります。

- up または down は、リンクの使用可能または使用不可能を示します。
- unknown は、リンクが up、down のいずれであるかの通知をドライバがサポートしておらず、したがってドライバがリンクの状態変化を検出しないことを示します。

PROBE 検査用アドレスが構成されたインタフェースについて、次のようにプローブベースの障害検出の状態を示します。

- ok は、プローブが機能しており、アクティブであることを示します。
- failed は、プローブベースの障害検出が、このインタフェースが動作していないことを検出したことを示します。
- unknown は、適切なプローブターゲットが見つからなかったため、プローブを送信できないことを示します。
- disabled は、このインタフェースでは IPMP 検査用アドレスが構成されていないことを示します。したがって、プローブベースの障害検出は無効になっています。

STATE 次のように、このインタフェース全体の状態を指定します。

- ok は、このインタフェースがオンラインになっており、障害検出手法の構成に基づいて正常に動作していることを示します。
- failed は、このインタフェースのリンクが停止しているか、またはこのインタフェースはトラフィックを送受信できないとプローブ検出が判定したために、このインタフェースが動作していないことを示します。
- offline は、このインタフェースが使用不可能であることを示します。通常、次の状況の下ではインタフェースがオフラインになります。
  - インタフェースがテスト中である。
  - 動的再構成が実行中である。
  - このインタフェースが、重複するハードウェアアドレスを別のインタフェースと共有している。
- unknown は、プローブベースの障害検出のプローブターゲットが見つからなかったために IPMP インタフェースの状態を判定できないことを示します。

例 6–12　IPMP のプローブターゲット情報の取得

-t オプションは、IPMP グループ内の各 IP インタフェースに関連付けられたプローブターゲットを特定します。最初の出力は、プローブベースの障害検出に検査用アドレスを使用する IPMP 構成の例です。

```
$ ipmpstat -nt
INTERFACE   MODE          TESTADDR        TARGETS
net0        routes        192.168.85.30   192.168.85.1 192.168.85.3
net1        disabled      --              --
net3        disabled      --              --
net4        routes        192.1.2.200     192.1.2.1
net2        multicast     128.9.0.200     128.0.0.1 128.0.0.2
net6        multicast     128.9.0.201     128.0.0.2 128.0.0.1
net5        multicast     128.9.0.202     128.0.0.1 128.0.0.2
net7        multicast     128.9.0.203     128.0.0.1 128.0.0.2
```

2 番目の出力は、推移的プローブまたはプローブベースの障害検出を検査用アドレスなしで使用する IPMP 構成の例です。

```
$ ipmpstat -nt
INTERFACE   MODE         TESTADDR         TARGETS
net3        transitive   <net1>           <net1> <net2> <net3>
net2        transitive   <net1>           <net1> <net2> <net3>
net1        routes       172.16.30.100    172.16.30.1
```

出力フィールドでは、次の情報が提供されます。

INTERFACE　IPMP グループのベースとなる各インタフェースを示します。

MODE　プローブターゲットを取得するための方法を指定します。

- routes は、プローブターゲットの検索にシステムのルーティングテーブルが使用されることを示します。
- mcast は、ターゲットの検索にマルチキャスト ICMP プローブが使用されることを示します。
- disabled は、このインタフェースでプローブベースの障害検出が無効になっていることを示します。
- transitive は、2 番目の例に示したように、障害検出に推移的プローブが使用されることを示します。推移的プローブと検査用アドレスを同時に使用してプローブベースの障害検出を実装することはできない点に注意してください。検査用アドレスを使用しない場合は、推移的プローブを有効にする必要があります。推移的プローブを使用しない場合は、検査用アドレスを構成する必要があります。概要については、82 ページの「プローブベースの障害検出」を参照してください。

TESTADDR　ホスト名、または -n オプションが -t オプションとともに使用された場合は、プローブの送受信のためにこのインタフェースに割り当てられた IP アドレスを示します。

例 6–12 IPMP のプローブターゲット情報の取得 (続き)

推移的プローブが使用された場合、インタフェースの名前は、データ受信用としてアクティブに使用されていない、ベースとなる IP インタフェースを表します。また、これらの名前は、指定されたこれらのインタフェースの発信元アドレスを使用して推移的テストプローブが送信されていることも示しています。データを受信するアクティブなベースとなる IP インタフェースの場合、表示される IP アドレスは、送信 ICMP プローブの発信元アドレスを示します。

注 – IP インタフェースに IPv4 検査用アドレスと IPv6 検査用アドレスの両方が構成されている場合、プローブターゲットの情報は各検査用アドレスについて個別に表示されます。

TARGETS 現在のプローブターゲットを空白区切りリストとして一覧表示します。プローブターゲットは、ホスト名と IP アドレスのどちらかで表示されます。`-n` オプションが `-t` オプションとともに使用された場合は、IP アドレスが表示されます。

例 6–13 IPMP のプローブの監視

`-p` オプションを使用すると、進行中のプローブを監視できます。`ipmpstat` コマンドにこのオプションを使用すると、Control-C を押してコマンドを終了するまで、システム上のプローブのアクティビティーに関する情報が継続的に表示されます。このコマンドを実行するには、Primary Administrator 特権を持っている必要があります。

最初の出力は、プローブベースの障害検出に検査用アドレスを使用する IPMP 構成の例です。

```
# ipmpstat -pn
TIME    INTERFACE   PROBE     NETRTT   RTT      RTTAVG     TARGET
0.11s   net0        589       0.51ms   0.76ms   0.76ms     192.168.85.1
0.17s   net4        612       --       --       --         192.1.2.1
0.25s   net2        602       0.61ms   1.10ms   1.10ms     128.0.0.1
0.26s   net6        602       --       --       --         128.0.0.2
0.25s   net5        601       0.62ms   1.20ms   1.00ms     128.0.0.1
0.26s   net7        603       0.79ms   1.11ms   1.10ms     128.0.0.1
1.66s   net4        613       --       --       --         192.1.2.1
1.70s   net0        603       0.63ms   1.10ms   1.10ms     192.168.85.3
^C
```

2 番目の出力は、推移的プローブまたはプローブベースの障害検出を検査用アドレスなしで使用する IPMP 構成の例です。

```
# ipmpstat -pn
TIME    INTERFACE   PROBE      NETRTT   RTT      RTTAVG     TARGET
1.39S   net4        t28        1.05ms   1.06ms   1.15ms     <net1>
```

例 6–13 IPMP のプローブの監視 *(続き)*

```
1.39s   net1        i29        1.00ms   1.42ms   1.48ms     172.16.30.1
^C
```

出力フィールドでは、次の情報が提供されます。

TIME ipmpstat コマンドが発行された時点を基準にしたプローブの送信時間を指定します。ipmpstat が開始される前にプローブが起動された場合、コマンドが発行された時点を基準にした負の値で時間が表示されます。

INTERFACE プローブの送信先となるインタフェースを指定します。

PROBE プローブを表す識別子を指定します。障害検出に推移的プローブが使用される場合は、この識別子に、推移的プローブの場合は t という接頭辞が、ICMP プローブの場合は i という接頭辞が付きます。

NETRTT プローブの合計ネットワーク往復時間を示します (ミリ秒で測定)。NETRTT は、IP モジュールがプローブを送信した時点から、IP モジュールがターゲットから ack パケットを受け取った時点までの時間をカバーします。プローブが失われたと in.mpathd デーモンが判定した場合、このフィールドは空になります。

RTT プローブの合計往復時間を示します (ミリ秒で測定)。RTT は、in.mpathd デーモンがプローブを送信するコードを実行した時点から、デーモンがターゲットからの ack パケットの処理を完了した時点までの時間が対象になります。プローブが失われたとデーモンが判定した場合、このフィールドは空になります。NETRTT に存在しないスパイクが RTT で発生する場合、それは、ローカルシステムが過負荷状態であることを示している可能性があります。

RTTAVG ローカルシステムとターゲットの間における、このインタフェース経由でのプローブの平均往復時間を示します。平均往復時間は、低速なターゲットの特定に役立ちます。データが不十分で平均を計算できない場合、このフィールドは空になります。

TARGET ホスト名、または -n オプションが -p とともに使用された場合は、プローブの送信先となるターゲットアドレスを示します。

ipmpstat コマンドの出力のカスタマイズ

ipmpstat コマンドの -o オプションを使用すると、出力をカスタマイズできます。前述した ipmpstat のほかのオプションとともにこのオプションを使用して、メインオプションで通常表示される合計フィールドの中から、表示する特定のフィールドを選択します。

たとえば、-g オプションでは次の情報が提供されます。

- IPMP グループ
- IPMP グループ名
- グループのステータス
- 障害検出時間
- IPMP グループのベースとなるインタフェース

システム上の IPMP グループのステータスだけを表示するとします。次の例に示すように、-o オプションと -g オプションを組み合わせ、フィールド groupname および state を指定します。

```
$ ipmpstat -g -o groupname,state
GROUPNAME  STATE
ipmp0      ok
accgt1     failed
field2     degraded
```

特定の種類の情報に関してのみ、ipmpstat コマンドのすべてのフィールドを表示するには、構文に -o all を含めます。たとえば、グループ情報を提供するすべてのフィールドを表示するには、ipmpstat -g -o all と入力します。

スクリプト内での ipmpstat コマンドの使用

-o オプションは、コマンドをスクリプトから発行したりコマンドエイリアスを使用して発行したりするときに、特にマシンによる解析が可能な出力を生成する場合に便利です。

マシンによる解析が可能な情報を生成するには、-P オプションと -o オプションを ipmpstat のほかのメインオプションの 1 つに組み合わせ、表示する特定のフィールドを指定します。マシンによる解析が可能な出力は、次のように通常の出力とは異なります。

- 列ヘッダーは省略されます。
- 各フィールドがコロン (:) で区切られます。
- 空の値を含むフィールドは、二重ダッシュ (--) が設定されるのではなく、空になります。
- 複数のフィールドをリクエストするとき、あるフィールドにリテラルのコロン (:) またはバックスラッシュ (\) が含まれている場合は、それらの文字の前にバックスラッシュ (\) を付加することで、それらの文字をエスケープつまり除外できます。

ipmpstat -P 構文を正しく使用するには、次の規則に従います。

- `-o` *option fields* を `-P` オプションとともに使用する。複数のオプションフィールドはコンマで区切る。
- `-o all` は `-P` オプションでは決して使用しない。

これらの規則のいずれかを無視した場合は、`ipmpstat -P` が失敗します。

次の例は、`-P` オプションを使用した場合の情報の形式を示しています。

```
$ ipmpstat -P -o -g groupname,fdt,interfaces
ipmp0:10.00s:net0 net1
acctg1::[net3 net4]
field2:20.00s:net2 net7 (net5) [net6]
```

グループ名、障害検出時間、およびベースとなるインタフェースは、グループ情報フィールドです。したがって、`-o -g` オプションを `-P` オプションとともに使用します。

`-P` オプションは、スクリプト内で使用するためのものです。次の例は、ipmpstat コマンドをスクリプトから発行する方法を示しています。このスクリプトは、IPMP グループの障害検出時間を表示します。

```
getfdt() {
         ipmpstat -gP -o group,fdt | while IFS=: read group fdt; do
             [[ "$group" = "$1" ]] && { echo "$fdt"; return; }
         done
     }
```

◆ ◆ ◆ 第 7 章

7 LLDP によるネットワーク接続情報の交換

どのローカルエリアネットワークでも、システムやスイッチなどの個々のコンポーネントが分離された状態で構成されることはありません。ネットワークトラフィックを効率よくホストするには、ネットワーク上のシステムの構成を相互に調整する必要があります。したがって、パケット交換は、コンポーネントの機能、リンクプロパティー構成、帯域幅制限などに基づいて行われます。各システムやスイッチなどのコンポーネントの間で互換性を確保するために、それぞれを手動で構成することができます。ただし、この方法にはリスクが伴い、特に複数の管理者が異なるシステムを独立して操作する場合は、構成ミスが発生しやすくなります。より優れた方法は、システムがそれぞれの構成情報をピアシステムに送信できるようなテクノロジを使用することです。このテクノロジを使用すると、リスクが軽減され、ネットワーク管理がより簡単になります。Oracle Solaris は、この特定の目的のためにリンク層検出プロトコル (LLDP) をサポートしています。

この章では、システムが LLDP を使用して、ローカルネットワーク全体にわたってシステムおよびネットワーク接続情報を交換できるようにする方法について説明します。この章の内容は次のとおりです。

Oracle Solaris での LLDP の概要

LLDP は、トポロジの検出を目的として、LAN 全体にわたって情報を通知します。このプロトコルを使用すると、システムは、接続や管理の情報をネットワーク上のほかのシステムに通知できます。これらの情報には、ネットワーク操作に関連するシステムの機能、管理アドレス、およびその他の情報を含めることができます。またこのプロトコルにより、そのシステムは、同じローカルネットワーク上にあるほかのシステムに関する同様の情報を受信できるようになります。

Oracle Solaris では、LLDP はデータセンターブリッジング交換プロトコル (DCBX) の TLV ユニットの交換にも使用されます。DCBX は、優先順位ベースのフロー制御 (PFC) や拡張伝送選択 (ETS) などの DCB 機能に関する構成情報を提供します。DCB の詳細については、第 8 章「Oracle Solaris におけるデータセンターブリッジング機能の操作」を参照してください。

LLDP を使用すると、システム管理者は、特に仮想ローカルエリアネットワーク (VLAN) やリンクアグリゲーションなどを含む複雑なネットワークで、誤ったシステム構成を容易に検出できます。ネットワークを構成しているサーバー、スイッチ、その他のデバイス間の物理的な接続を追跡しなくても、トポロジに関する情報を簡単に取得できます。

LLDP 実装のコンポーネント

LLDP は、次のコンポーネントを使用して実装されています。

- LLDP 機能を有効にするには、LLDP パッケージがインストールされている必要があります。このパッケージは、LLDP デーモン、コマンド行ユーティリティー、サービスマニフェストとスクリプトのほか、LLDP が動作するために必要なその他のコンポーネントを提供します。
- `lldp` サービスは、`svcadm` コマンドによって有効になります。このサービスは LLDP デーモンを管理するとともに、このデーモンの起動、停止、再起動、またはリフレッシュを行います。LLDP パッケージをインストールすると、このサービスは自動的に有効になります。
- `lldpadm` コマンドは個々のリンク上の LLDP を管理し、たとえば LLDP の動作モードを構成したり、送信される TLV (Type-Length-Value) ユニットを指定したり、DCBX TLV ユニットを構成したりするために使用されます。特に、このコマンドは、エージェントごとの LLDP プロパティーやグローバルな LLDP プロパティーを設定するために使用されます。`lldpadm` コマンドの一般的なサブコマンドは、`dladm` および `ipadm` コマンドの各サブコマンドに対応しています。
  - `lldpadm set-*` は、指定された LLDP プロパティーに 1 つ以上の値を設定するアクションの実行を指定します。
  - `lldpadm show-*` は、指定された LLDP プロパティーに設定されている値を表示します。
  - `lldpadm reset-*` は、指定された LLDP プロパティーの構成をデフォルト値にリセットします。

  これらのサブコマンドの使用は、以降のセクションに示されています。`lldpadm` コマンドの詳細は、`lldpadm(1M)` のマニュアルページを参照してください。
- LLDP ライブラリ (`liblldp.so`) は、リンク上の LLDP 情報を取得したり、LLDP パケットを解析したり、その他の機能を実行したりするために使用できる API を提供します。

- LLDP エージェントは、LLDP が有効になっているネットワークインタフェースカードに関連付けられた LLDP インスタンスです。LLDP エージェントは、関連付けられた NIC 上の LLDP の動作を制御します。LLDP エージェントは、NIC または物理リンク上でのみ構成でき、これらのリンクのポートを使用して情報を通知します。したがって、この LLDP ドキュメントでは、LLDP エージェントの名前、それが有効になっている物理リンク、およびポートは同一です。
- LLDP デーモン (`lldpd`) は、システム上の LLDP エージェントを管理します。また、SNMP (Simple Network Management Protocol) を経由してシステム上で受信される LLDP 情報を取得するために、SNMP のためのデーモンである `snmpd` とも相互作用します。さらに、このデーモンは LLDP `sysevents` 情報を送信したり、LLDP ライブラリからのクエリーに応答したりします。

LLDP エージェントの情報源

LLDP エージェントは、LLDP データユニット (LLDPDU) を送信したり、受信したりします。このエージェントは、次の 2 つのタイプのデータストア内で、これらの LLDPDU に含まれている情報を管理および格納します。

- ローカル管理情報ベース (ローカル **MIB**) - このデータストアには、LLDP エージェントが有効になっているシステムの特定のリンクに関連するネットワーク情報が含まれています。ローカル MIB には、一般的な情報と固有の情報の両方が含まれます。たとえば、シャーシ ID は、システム上のすべての LLDP エージェントの間で共有されている一般的な情報です。ただし、システムのデータリンクのポート ID は異なります。そのため、各エージェントは、独自のローカル MIB を管理します。
- リモート **MIB** - このデータストア内の情報は、ピアホストの LLDP エージェントから受信します。

LLDP エージェントの動作モード

LLDP エージェントは、次のモードで動作します。

- 送信のみ (`txonly`): このモードでは、LLDP エージェントは受信 LLDPDU を処理しません。そのため、リモート MIB は空です。
- 受信のみ (`rxonly`): このモードでは、エージェントは受信 LLDPDU のみを処理し、情報をリモート MIB 内に格納します。ただし、ローカル MIB からの情報は送信されません。
- 送受信 (`both`): このモードでは、エージェントはローカル情報を送信し、受信 LLDPDU を処理し、そのためにローカル MIB とリモート MIB の両方を維持します。ベースとなるリンクが DCB 機能をサポートしている場合は、サポートされる DCB 機能の DCBX TLV ユニットが自動的に有効になります。
- 無効 (`disable`): このモードでは、エージェントは存在しません。

LLDP の SMF プロパティー

サービス管理機能 (SMF) プロパティー `auto-enable-agents` は、システム上で LLDP を有効にする方法を制御します。このプロパティーでは、LLDP をすべての物理リンクにわたってグローバルに有効にするか、一度に 1 つの物理リンクでのみ有効にするかを選択できます。このプロパティーには、次の 3 つの取り得る値のいずれかを指定できます。

- `yes` は、この SMF プロパティーのデフォルト値です。この値を指定すると、ポートに以前の LLDP 構成が存在していない場合は、LLDP がすべてのポートでグローバルに、受信モードと送信モードの両方で有効になります。ポートに構成が存在している場合は、そのポートの構成が保持されます。たとえば、ポートに受信モードのみの LLDP がすでに構成されている場合、LLDP サービスはエージェントを送受信モードでの実行に切り替えません。そのポートの LLDP は引き続き受信モードになります。
- `force` は、LLDP をすべてのポートで受信モードと送信モードの両方で有効にし、ポートの既存の LLDP 構成をすべてオーバーライドします。たとえば、ポートの以前の LLDP 構成が受信モードのみで動作している場合、LLDP エージェントは、受信モードと送信モードの両方で動作するデフォルトの LLDP モードに切り替えられます。
- `no` は、既存の LLDP 構成があるポートを除くすべてのポートで、LLDP の自動有効化を無効にします。これらのポートでは、既存の LLDP 構成が保持されます。

`auto-enable-agents` プロパティーをカスタマイズするたびに、新しい値を有効にするために LLDP SMF サービスを再起動する必要があります。

LLDP エージェントが通知する情報

LLDP エージェントは、LLDP パケットまたは LLDPDU でシステムおよび接続情報を送信します。このようなパケットには、TLV (Type-Length-Value) 形式で個別にフォーマットされた情報ユニットが含まれています。そのため、これらの情報ユニットは *TLV* ユニットとも呼ばれます。特定の TLV ユニットは必須であり、LLDP が有効になったときにデフォルトで LLDP パケットに含まれます。`lldpadm` コマンドを使用して、このようなユニットのいずれかを除外することはできません。必須の TLV ユニットは次のとおりです。

- シャーシ ID
- ポート ID
- TTL (生存期間)
- PDU の終了

シャーシ ID は `hostid` コマンドによって生成される情報と同じです。ポート ID は物理 NIC の MAC アドレスです。リンクの数に応じて、1 つのシステム内で複数の

LLDP エージェントを有効にすることができます。シャーシ ID とポート ID の組み合わせによってエージェントが一意に識別され、システム上のほかのエージェントから区別されます。

オプションの TLV ユニットを LLDP パケットに追加できます。これらのオプションの TLV ユニットは、ベンダーが、通知されるベンダー固有の TLV ユニットを挿入する手段になります。TLV ユニットは個々の組織一意識別子 (OUI) によって識別され、これらの OUI が IEEE 802.1 仕様または IEEE 802.3 仕様のどちらに従っているかに応じて入力されます。LLDP エージェントのプロパティーを構成して、このようなオプションの TLV ユニットの送信を有効または無効にすることができます。

次の表は、各 TLV のタイプまたはグループ、それに対応するプロパティー名、プロパティーごとの TLV ユニット、およびそれらの説明を示しています。LLDP が有効になったときにパケットに含まれる TLV ユニットを指定するには、これらのプロパティーのいずれかを構成します。

表 7–1 LLDP エージェントに対して有効にできる TLV ユニット

TLV のタイプ	プロパティー名	TLV ユニット	説明
基本的な管理	`basic-tlv`	`sysname`、`portdesc`、`syscapab`、`sysdesc`、`mgmtaddr`	通知されるシステム名、ポートの説明、システムの機能、システムの説明、および管理アドレスを指定します。
802.1 OUI	`dot1-tlv`	`vlanname`、`pvid`、`linkaggr`、`pfc`、`appln`、`evb`、`etscfg`	通知される次の項目を指定します: VLAN 名、ポートの VLAN ID、リンクアグリゲーション、優先順位ベースのフロー制御の TLV ユニット、アプリケーション、拡張伝送選択、およびエッジ仮想ブリッジング。
802.3 OUI	`dot3-tlv`	`max-framesize`	通知される最大フレームサイズを指定します。
Oracle 固有の OUI (`0x0003BA` として定義されている)	`virt-tlv`	`vnic`	仮想ネットワークが構成されている場合は、通知される VNIC を指定します。

TLV ユニットとそのプロパティー

各 TLV ユニットにはプロパティーがあり、これらのプロパティーは特定の値を使用してさらに構成できます。TLV ユニットが LLDP エージェントのプロパティーとして有効になると、その TLV ユニットは、指定された値のみを使用してネットワーク内で通知されます。たとえば、システムの機能を通知する TLV 値 `syscapab` を考えてみます。これらの機能には、ルーター、ブリッジ、リピータ、電話などのデバイスに対するサポートが含まれる可能性があります。ただし、ルーターやブリッジなど、特定のシステム上で実際にサポートされている機能のみが通知されるように `syscapab` を設定できます。

TLV ユニットを構成するための手順は、グローバルな *TLV* ユニットまたはエージェントごとの *TLV* ユニットのどちらを構成するかによって異なります。

グローバルな TLV ユニットは、システム上のすべての LLDP エージェントに適用されます。次の表は、グローバルな TLV 値とそれに対応する、取り得る構成を示しています。

表 7–2 グローバルな TLV ユニットとそのプロパティー

TLV 名	TLV プロパティー名	取り得るプロパティー値	値の説明
`syscapab`	`supported`	`other`、`repeater`、`bridge`、`wlan-ap`、`router`、`telephone`、`docsis-cd`、`station`、`cvlan`、`sylvan`、`tpmr`	システムのサポートされている主要な機能を表します。デフォルト値は、`router`、`station`、および `bridge` です。
	`enabled`	`supported` に対して示されている値のサブセット	システムの有効になっている機能を表します。
`mgmtaddr`	`ipaddr`	`ipv4` または `ipv6`	ローカルの LLDP エージェントに関連付けられる IP アドレスのタイプを指定します。これらのアドレスは、上位階層エンティティーに到達するために使用され、ネットワーク管理による検出に役立ちます。指定できるのは 1 つのタイプだけです。

グローバルな値を取ることのできない TLV ユニットは、LLDP エージェントのレベルで管理されます。エージェントごとの TLV ユニットでは、指定した値は、特定の LLDP エージェントがその TLV ユニットの転送を有効にしたときに使用されます。

次の表は、LLDP エージェントの TLV 値とそれに対応する、取り得る構成を示しています。

表 7–3 エージェントごとの TLV ユニットとそのプロパティー

TLV 名	TLV プロパティー名	取り得るプロパティー値	値の説明
pfc	willing	on、off	優先順位ベースのフロー制御に関連するリモートマシンからの構成情報を受け入れるか、または拒否するように LLDP エージェントを設定します。
appln	apt	値は、アプリケーション優先順位表で定義されている情報から取得されます。	アプリケーション優先順位表を構成します。この表には、アプリケーション TLV ユニットとそれに対応する優先順位の一覧が含まれています。アプリケーションは、id/selector のペアで識別されます。この表の内容では、次の形式が使用されます。 id/selector/priority
etscfg	willing	on、off	拡張伝送選択に関連するリモートマシンからの構成情報を受け入れるか、または拒否するように LLDP エージェントを設定します。

エージェントごとの TLV ユニットについては、第 8 章「Oracle Solaris におけるデータセンターブリッジング機能の操作」を参照してください。

システムでの LLDP の有効化

次の手順では、システム情報をネットワーク上のほかのホストまたはピアと交換するように LLDP を構成する方法について説明します。

- 128 ページの「LLDP を配備する方法」
- 131 ページの「エージェントの LLDP パケットの TLV ユニットを指定する方法」
- 132 ページの「TLV 値を定義する方法」
- 134 ページの「LLDP の無効化」

▼ LLDP を配備する方法

次の手順では、システム上で LLDP を使用してシステム機能の通知を開始する方法について説明します。デフォルトでは、LLDP パッケージのインストールが完了すると、LLDP が有効で使用できる状態になります。デフォルトの LLDP 構成で十分な場合は、ほとんどの手順がオプションになります。

始める前に　LLDP を使用するには、LLDP パッケージをインストールする必要があります。パッケージをインストールするには、次のコマンドを入力します。

```
# pkg install lldp
```

1　LLDP サービスが開始されたことを確認します。

```
# svcs lldp
STATE          STIME    FMRI
online         Jul_10   svc:/network/lldp:default
```

LLDP サービスが無効になっている場合は、次のコマンドでサービスを起動します。

```
# svcadm enable svc:/network/lldp:default
```

2　次の手順のいずれかを実行します。

- LLDP サービスをシステム上でグローバルに有効にする場合は、LLDP エージェントで通知する TLV ユニットを指定します。

  ```
  # lldpadm set-agentprop -p property=value agent
  ```

  ここで、*agent* は LLDP エージェントであり、エージェントが有効になっている物理リンクによって識別されます。したがって、LLDP が `net0` で有効になっている場合、エージェントは `net0` です。

  注 – `lldpadm` のサブコマンドを発行するときは省略形を使用できます。たとえば、`lldpadm set-agentprop` の代わりに、`lldpadm set-ap` と入力できます。サブコマンドとその省略形については、`lldpadm(1M)` のマニュアルページを参照してください。

  LLDP エージェントのプロパティーの説明については、124 ページの「LLDP エージェントが通知する情報」を参照してください。

  LLDP エージェントのプロパティーを一覧表示するには、`lldpadm show-agentprop` と入力します。または、表 7–1 を参照してください。

  手順については、131 ページの「エージェントの LLDP パケットの TLV ユニットを指定する方法」を参照してください。

- 選択したポートでのみ LLDP サービスを有効にする場合は、次の手順を実行します。

a. `auto-enable-agents` SMF プロパティーを `no` に変更します。

```
# svccfg -s svc:/network/lldp:default setprop lldp/auto-enable-agents = "no"
```

この SMF プロパティーは、システム上で LLDP を有効にする方法を決定します。取り得る値は、`yes`、`force`、および `no` の 3 つです。デフォルトでは、このプロパティーは `yes` に設定されます。これらの値の説明、およびこれらの値から生じる LLDP エージェントの動作については、124 ページの「LLDP の SMF プロパティー」を参照してください。

b. LLDP サービスを再起動します。

```
# svcadm restart svc:/network/lldp:default
```

c. 選択したポートまたはリンク上で LLDP エージェントを有効にします。

```
# lldpadm set-agentprop -p mode=value agent
```

ここで、*agent* は LLDP エージェントであり、エージェントが有効になっている物理リンクによって識別されます。したがって、LLDP を `net0` で有効にする場合、エージェントは `net0` です。

プロパティー `mode` は、LLDP エージェントの動作モードを表す 4 つの取り得る値 `tx`、`rx`、`both`、`disable` のいずれかに設定できます。これらの値の説明については、123 ページの「LLDP エージェントの動作モード」を参照してください。

d. LLDP エージェントで通知する TLV ユニットを指定します。

```
# lldpadm set-agentprop -p property=value agent
```

LLDP エージェントのプロパティーの説明については、124 ページの「LLDP エージェントが通知する情報」を参照してください。

`mode` プロパティーに加え、LLDP エージェントのその他のプロパティーを一覧表示するには、`lldpadm show-agentprop` と入力します。または、表 7–1 を参照してください。

手順については、131 ページの「エージェントの LLDP パケットの TLV ユニットを指定する方法」を参照してください。

3 必要に応じて、グローバルな **TLV** ユニットをカスタマイズします。

```
# lldpadm set-tlvprop -p property=value global-tlv
```

ここで、*property* は、グローバルな TLV ユニットのプロパティーを表します。

グローバルな TLV ユニットの説明については、126 ページの「TLV ユニットとそのプロパティー」を参照してください。

グローバルな TLV を一覧表示するには、`lldpadm show-tlvprop` と入力します。または、表 7–2 を参照してください。

手順については、132 ページの「TLV 値を定義する方法」を参照してください。

4 必要に応じて、エージェントごとの **TLV** ユニットをカスタマイズします。

```
# lldpadm set-agenttlvprop -p property=value -a agent per-agent-tlv
```

ここで、*property* は、エージェントごとの TLV ユニットのプロパティーを表します。

エージェントごとの TLV ユニットの説明については、126 ページの「TLV ユニットとそのプロパティー」を参照してください。

エージェントごとの TLV を一覧表示するには、`lldpadm show-tlvprop` と入力します。または、表 7–2 を参照してください。

手順については、132 ページの「TLV 値を定義する方法」を参照してください。

例 7–1 auto-enable-agents SMF プロパティーのカスタマイズ

次の例では、この SMF プロパティーの値を変更すると LLDP が異なる方法で有効になることを示します。4 つのポートを備えたシステムで、LLDP が 2 つのポート上で次のように構成されているとします。

- `net0`: 受信モードと送信モード
- `net1`: 受信のみ
- `net2` および `net3`: なし

この SMF プロパティーをデフォルト値 `yes` に設定すると、`net2` と `net3` で LLDP が自動的に有効になります。LLDP 構成は次のように表示されます。

```
# lldpadm show-agentprop -p mode
AGENT    PROPERTY   PERM   VALUE    DEFAULT   POSSIBLE
net0     mode       rw     both     disable   txonly,rxonly,both,
                                               disable
net1     mode       rw     rxonly   disable   txonly,rxonly,both,
                                               disable
net2     mode       rw     both     disable   txonly,rxonly,both,
                                               disable
net3     mode       rw     both     disable   txonly,rxonly,both,
                                               disable
```

この SMF プロパティーを `no` に切り替えると、サービスの再起動時に構成が変更されます。

```
# svccfg -s svc:/network/lldp:default setprop lldp/auto-enable-agents = "no"
# svcadm restart svc:/network/lldp:default
# lldpadm show-agentprop -p mode
AGENT    PROPERTY   PERM   VALUE    DEFAULT   POSSIBLE
net0     mode       rw     both     disable   txonly,rxonly,both,
                                               disable
net1     mode       rw     rxonly   disable   txonly,rxonly,both,
                                               disable
net2     mode       rw     disable  disable   txonly,rxonly,both,
                                               disable
net3     mode       rw     disable  disable   txonly,rxonly,both,
                                               disable
```

出力例では、それまで LLDP モードが自動的に有効になっていた net2 と net3 に、現在は無効としてとしてフラグが付けられています。ただし、LLDP エージェントがあらかじめ構成されていた net0 と net1 には、変更は発生しません。

例 7–2 複数のデータリンク上の LLDP を有効にする

この例では、LLDP を選択的に有効にする方法を示します。システムに 2 つのデータリンク net0 と net1 が存在します。net0 では、エージェントで LLDP パケットの送信と受信を行うようにします。net1 では、エージェントで LLDP パケットの送信のみを行うようにします。入力するコマンドは次のようになります。

```
# svccfg -s svc:/network/lldp:default setprop lldp/auto-enable-agents = "no"
# svcadm restart svc:/network/lldp:default
# lldpadm set-agentprop -p mode=both net0
# lldpadm set-agentprop -p mode=txonly net1
```

▼ エージェントの LLDP パケットの TLV ユニットを指定する方法

この手順では、エージェントが送信する LLDP パケット内で通知する、TLV ユニットを指定する方法について説明します。TLV ユニットを指定するには、lldpadm set-agentprop サブコマンドを使用します。

1. 必要な場合は、追加する **TLV** ユニットを含めることのできる **LLDP** エージェントプロパティーを識別します。

   このサブコマンドではまた、プロパティーごとにすでに設定されている TLV ユニットも表示されます。

   ```
   # lldpadm show-agentprop agent
   ```

   プロパティーを指定しない場合は、すべての LLDP エージェントプロパティーとそれらの TLV 値が表示されます。

2. プロパティーに **TLV** ユニットを追加します。

   ```
   # lldpadm set-agentprop -p property[+|-]=value[,...] agent
   ```

   複数の値を受け入れるプロパティーには、+|- の修飾子が使用されます。これらの修飾子を使用すると、一覧に値を追加する (+) か、または削除する (-) ことができます。これらの修飾子を使用しない場合は、設定する値により、以前にそのプロパティーに対して定義されていたすべての値が置き換えられます。

3. (オプション) プロパティーの新しい値を表示します。

   ```
   # lldpadm show-agentprop -p property agent
   ```

例 7–3 LLDP パケットへのオプションの TLV ユニットの追加

この例では、LLDP エージェント net0 がすでに、LLDP パケットで VLAN 情報を通知するように構成されています。それに加えて、システムの機能、リンクアグリゲーション、およびネットワーク仮想化の情報も通知されるようにします。ただし、パケットから VLAN の説明は削除します。

```
# lldpadm show-agentprop net0
AGENT    PROPERTY    PERM    VALUE           DEFAULT    POSSIBLE
net0     mode        rw      both            disable    txonly,rxonly,both,
                                                        disable
net0     basic-tlv   rw      sysname,         none      none,portdesc,
                             sysdesc                    sysname,sysdesc,
                                                        syscapab,mgmtaddr,
                                                        all
net0     dot1-tlv    rw      vlanname,        none      none,vlanname,pvid,
                             pvid,pfc                   linkaggr,pfc,appln,
                                                        evb,etscfg,all
net0     dot3-tlv    rw      max-framesize    none      none, max-framesize,
                                                        all
net0     virt-tlv    rw      none             none      none,vnic,all


# lldpadm set-agentprop -p basic-tlv+=syscapab,dot1-tlv+=linkaggr,virt-tlv=vnic net0
# lldpadm set-agentprop -p dot1-tlv-=vlanname net0
# lldpadm show-agentprop -p net0
AGENT    PROPERTY    PERM    VALUE           DEFAULT    POSSIBLE
net0     mode        rw      both            disable    txonly,rxonly,both,
                                                        disable
net0     basic-tlv   rw      sysname,         none      none,portdesc,
                             sysdesc,                   sysname,sysdesc,
                             syscapab                   syscapab,mgmtaddr,
                                                        all
net0     dot1-tlv    rw      pvid,            none      none,vlanname,pvid,
                             linkaggr                   linkaggr,pfc,appln,
                                                        evb,etscfg,all
net0     dot3-tlv    rw      max-framesize    none      none, max-framesize,
                                                        all
net0     virt-tlv    rw      vnic             none      none,vnic,all
```

▼ TLV 値を定義する方法

この手順では、特定の TLV ユニットに値を指定する方法について説明します。次のいずれかのサブコマンドを使用します。

- グローバルな TLV ユニットを構成する場合は `lldpadm set-tlvprop`。
- エージェントごとの TLV ユニットを構成する場合は `lldpadm set-agenttlvprop`。

1 グローバルな **TLV** ユニットを構成するのか、エージェントごとのユニットを構成するのかに応じて、次のいずれかの手順を実行します。

- グローバルな TLV ユニットを構成する場合は、該当する TLV プロパティーを、通知しようとする値を含むように設定します。

```
# lldpadm set-tlvprop -p tlv-property=value[,value,value,...] tlv-name
```

ここで、*tlv-name* はグローバルな TLV ユニットの名前、*tlv-property* はその TLV ユニットのプロパティーです。プロパティーに複数の値を割り当てることができます。参考のため、表 7–2 を参照してください。

- エージェントごとの TLV ユニットを構成する場合は、LLDP エージェントの該当する TLV プロパティーを、エージェントで通知しようとする値を含むように構成します。

```
# lldpadm set-agenttlvprop -p tlv-property[+|-]=value[,value,value,...] -a agent tlv-name
```

ここで、*tlv-name* はエージェントの TLV ユニットの名前、*tlv-property* はその TLV ユニットのプロパティーです。プロパティーに複数の値を割り当てることができます。参考のため、表 7–3 を参照してください。

2 (オプション) 次のいずれかの手順を実行して、先ほど構成した **TLV** プロパティーの値を表示します。

- グローバルな TLV プロパティーの値を表示するには、次のコマンドを使用します。

```
# lldpadm show-tlvprop
```

- エージェントの TLV プロパティーの値を表示するには、次のコマンドを使用します。

```
# lldpadm show-agenttlvprop
```

例 7–4 システムの機能と管理 IP アドレスの指定

この例では、次の 2 つの目的を達成します。

- LLDP パケットで通知されるシステムの機能に関する特定の情報を指定します。この目的を達成するには、`syscapab` TLV ユニットの `supported` プロパティーと `enabled` プロパティーの両方を構成する必要があります。
- 通知で使用される管理 IP アドレスを指定します。

```
# llpdadm set-tlvprop -p supported=bridge,router,repeater syscapab
# llpdadm set-tlvprop -p enabled=router syscapab
# llpdadm set-tlvprop -p ipaddr=192.168.1.2 mgmtaddr
# llpdadm show-tlvprop
TLVNAME    PROPERTY   PERM  VALUE        DEFAULT         POSSIBLE
syscapab   supported  rw    bridge,      bridge,router,  other,router,
                            router,      station         repeater,bridge,
                            repeater                     wlan-ap,telephone,
                                                         docis-cd,station,
                                                         cvlan,svlan,tpmr
syscapab   enabled    rw    router       none            bridge,router,
                                                         repeater
mgmtaddr   ipaddr     rw    192.162.1.2  none            --
```

LLDP の無効化

LLDP を個々のポートで選択的に無効にするには、次のいずれかのコマンドを使用します。

- `lldpadm set-agentprop -p mode=disable` *agent*

  ここで、*agent* は LLDP エージェントであり、エージェントが有効になっている物理リンクによって識別されます。したがって、LLDP を `net0` で有効にする場合、エージェントは `net0` です。このコマンドは、エージェントのモードを変更することによって LLDP を無効にします。

- `lladpadm reset-agentprop`

  このコマンドでは、`mode` プロパティーの値を設定しません。このコマンドは、ポートから LLDP 構成を削除することによって LLDP を無効にします。

注意 – サブコマンド `lldpadm reset–agentprop` は、ポートから LLDP 構成を完全に削除します。`no` に設定されている `auto-enable-agents` が切り替えられて `yes` に戻った場合、LLDP の動作は、そのポートでエージェントのモードが単に無効になっていた場合とは異なります。

LLDP をシステムのすべてのインタフェースでグローバルに無効にするには、次の手順を実行します。

1. SMF LLDP プロパティーを `no` に変更します。

   ```
   # svccfg -s svc:/network/lldp:default setprop lldp/auto-enable-agents = "no"
   ```

2. LLDP サービスを再起動します。

   ```
   # svcadm restart svc:/network/lldp:default
   ```

3. 以前の LLDP 構成が保持されている各ポートで、LLDP を無効にします。

   ```
   # lldpadm set-agentprop -p mode=disable agent
   ```

LLDP エージェントのモニタリング

`lldpadm show-agent` サブコマンドでは、LLDP エージェントによって通知される完全な情報が表示されます。特定のシステムに応じて、通知は、ネットワークのほかの部分に送信される、そのローカルシステムに関する情報である場合があります。あるいは、通知は、そのシステムによって受信される、同じネットワーク上のほかのシステムからの情報である場合があります。

このセクションでは、次の 2 つの手順について説明します。

- 135 ページの「通知を表示する方法」
- 136 ページの「LLDP 統計情報を表示する方法」

▼ 通知を表示する方法

この手順は、LLDP エージェントによって通知されている情報を表示する方法を示しています。情報はローカルまたはリモートのいずれかにできます。ローカル情報は、ローカルシステムから得られます。リモート情報は、ネットワーク上のほかのシステムから得られ、ローカルシステムによって受信されます。

- **lldpadm show-agent** サブコマンドを適切なオプションとともに使用して、必要な情報を表示します。
  - **LLDP** エージェントによって通知されるローカル情報を表示するには、次のコマンドを入力します。

    # lldpadm show-agent -l *agent*
  - **LLDP** エージェントによって受信されるリモート情報を表示するには、次のコマンドを入力します。

    # lldpadm show-agent -r *agent*
  - ローカル情報またはリモート情報のどちらかを詳細に表示するには、次のコマンドを入力します。

    # lldpadm show-agent -[l|r]v *agent*

例 7–5 通知される LLDP エージェント情報の取得

次の例は、LLDP エージェントによってローカルに、またはリモートから通知されている情報を表示する方法を示しています。デフォルトでは、情報は省略形式で表示されます。-v オプションを使用すると、詳細情報を取得できます。

```
# lldpadm show-agent -l net0
AGENT   CHASSISID   PORTID
net0    004bb87f    00:14:4f:01:77:5d

# lldpadm show-agent -lv net0
                      Agent: net0
         Chassis ID Subtype: Local(7)
            Port ID Subtype: MacAddress(3)
                    Port ID: 00:14:4f:01:77:5d
          Port Description: net0
              Time to Live: 81 (seconds)
               System Name: hosta.example.com
        System Description: SunOS 5.11 dcb-clone-x-01-19-11 i86pc
     Supported Capabilities: bridge,router
       Enabled Capabilities: router
         Management Address: 192.168.1.2
         Maximum Frame Size: 3000
               Port VLAN ID: --
               VLAN Name/ID: vlan25/25
        VNIC PortID/VLAN ID: 02:08:20:72:71:31
    Aggregation Information: Capable, Not Aggregated
```

```
                PFC Willing: --
                    PFC Cap: --
                    PFC MBC: --
                 PFC Enable: --
                PFC Pending: --
Application(s) (ID/Sel/Pri): --
    Information Valid Until: 117 (seconds)

# lldpdadm show-agent -r net0
AGENT   SYSNAME   CHASSISID   PORTID
net0    hostb     0083b390    00:14:4f:01:59:ab

# llpdadm show-agent -rv net0
                      Agent: net0
         Chassis ID Subtype: Local(7)
            Port ID Subtype: MacAddress(3)
                    Port ID: 00:14:4f:01:59:ab
           Port Description: net0
               Time to Live: 121 (seconds)
                System Name: hostb.example.com
         System Description: SunOS 5.11 dcb-clone-x-01-19-11 i86pc
     Supported Capabilities: bridge,router
       Enabled Capabilities: router
         Management Address: 192.168.1.3
         Maximum Frame Size: 3000
               Port VLAN ID: --
               VLAN Name/ID: vlan25/25
        VNIC PortID/VLAN ID: 02:08:20:72:71:31
    Aggregation Information: Capable, Not Aggregated
                PFC Willing: --
                    PFC Cap: --
                    PFC MBC: --
                 PFC Enable: --
Application(s) (ID/Sel/Pri): --
    Information Valid Until: 117 (seconds)
```

▼ LLDP 統計情報を表示する方法

ローカルシステムまたはリモートシステムによって通知されている LLDP パケットに関する情報を取得するために、LLDP 統計情報を表示できます。この統計情報は、LLDP パケットの送信および受信を含む重大なイベントを示します。

1 **LLDP** パケットの送信および受信に関するすべての統計情報を表示するには、次のコマンドを使用します。

```
# lldpadm show-agent -s agent
```

2 選択された統計情報を表示するには、**-o** オプションを使用します。

```
# lldpadm show-agent -s -o field[,field,...]agent
```

ここで、*field* は `show-agent -s` コマンドの出力にあるいずれかのフィールド名を示します。

例 7–6 LLDP パケット統計情報の表示

この例は、LLDP パケット通知に関する情報を表示する方法を示しています。

```
# lldpadm show-agent -s net0
AGENT IFRAMES IEER IDISCARD OFRAMES OLENERR TLVDISCARD TLVUNRECOG AGEOUT
net0        9    0        0      14       0          4          5      0
```

このコマンド出力では、次の情報が提供されます。

- AGENT は、LLDP エージェントの名前を指定します。これは、この LLDP エージェントが有効になっているデータリンクの名前と同じです。
- IFRAMES、IEER、および IDISCARD には、受信されているパケット、エラーを含む受信パケット、および破棄された受信パケットに関する情報が表示されます。
- OFRAMES と OLENERR は、送信パケットと、長さのエラーを含むパケットを示します。
- TLVDISCARD と TLVUNRECOG には、破棄された TLV ユニットと、認識されない TLV ユニットに関する情報が表示されます。
- AGEOUT は、タイムアウトしたパケットを示します。

この例は、システムで受信した 9 フレームのうち、おそらく標準に準拠していないために 5 つの TLV ユニットが認識されないことを示します。この例はまた、ローカルシステムによって 14 フレームがネットワークに送信されたことも示しています。

◆ ◆ ◆ 第 8 章

Oracle Solaris におけるデータセンターブリッジング機能の操作

第 7 章「LLDP によるネットワーク接続情報の交換」で説明されている LLDP と同様に、データセンターブリッジング (DCB) にもネットワーク上のピアとの情報交換が伴います。情報は、特にデータセンターなどトラフィックの重い環境における、ネットワークパケットの整合性に影響する構成に関連します。DCB は、このようなセンターの高速トラフィックに関連するコンポーネント構成を調整することにより、効率のよいネットワークトラフィック交換を可能にします。

この章で扱う内容は、次のとおりです。

データセンターブリッジング (DCB) の概要

データセンターブリッジングは、特にネットワークトラフィック量が多く転送速度の高い環境のトラフィックを管理するために、従来の Ethernet ネットワークの機能を拡張した一連の機能です。このようなトラフィックの伝送にはファイバチャネルを専用に割り当てることができます。ただし、専用リンクをファイバチャネルトラフィックだけに使用すると、コストが高くなる可能性があります。そのため、FCoE (fiber channel traffic over Ethernet) がより一般的に使用されます。ファイバチャネルは Ethernet ネットワークを通過する間のパケットロスに敏感ですが、DCB 機能はこれに対処します。

DCB では、ピアは優先順位に基づいてトラフィックを区別できます。優先順位を区別することにより、ホスト間で輻輳が発生した場合に、より高い優先順位を持つトラフィックに対してパケットの整合性を確実に維持することができます。DCB 交換プロトコル (DCBX) では、通信を行うホストが、高速ネットワークトラフィックに影響する構成情報を交換できます。ピアは次に、共通の構成についてネゴシエーションを行なって、トラフィックフローを維持しながら、高い優先順位を持つパケットのパケットロスを防止するようにします。

Oracle Solaris では、DCBX TLV ユニットの交換に LLDP が使用されます。ベースとなる NIC が DCB をサポートしている場合、優先順位ベースのフロー制御 (PFC) や拡張伝送選択 (ETS) などの DCB 機能を、ネットワーク上のピアホストと共有することができます。

- 優先順位ベースのフロー制御 (PFC) は、定義されたサービスクラス (CoS) 優先順位を持つパケットに対して、トラフィックフローを一時停止するメカニズムを実装することにより、パケットロスを防止します。141 ページの「優先順位ベースのフロー制御」を参照してください。CoS の詳細については、`dladm(1M)` のマニュアルページにある `cos` リンクプロパティーの説明を参照してください。
- 拡張伝送選択 (ETS) は、定義された CoS 優先順位に基づくパケット間での帯域幅共有を可能にします。146 ページの「拡張伝送選択」を参照してください。

LLDP を使用して交換されるすべてのシステム情報と同様に、ホストにはローカル DCB 情報とリモート DCB 情報の 2 種類の DCB 情報が存在します。PFC 機能が有効になるには、ホスト上の PFC に関するこれら 2 種類の DCB 情報が対称でなければなりません。通常、ローカルホストはピアから受信する DCB 情報を照合できる必要があります。DCB が有効になっている Oracle Solaris システムでは、DCB 情報をピアと同期するこの機能も有効になります。

注 – DCB 機能を Oracle Solaris 11 システムで使用できるのは、物理 NIC が DCB をサポートしている場合だけです。また、そのカードは DCB モードで動作するように構成されている必要があります。

▼ DCBX を有効にする方法

LLDP を有効にすると、DCBX のサポートは自動的に有効になります。この手順では、何らかの自動プロセスが失敗した場合に手動で行う代替手順を示します。この手順は `net0` に対して実行されるものとします。

1 **LLDP パッケージをインストールします。**

```
# pkg install lldp
```

2 **LLDP サービスが実行されていることを確認します。**

```
# svcs lldp
```

LLDP サービスが無効になっている場合は、次のコマンドでサービスを起動します。

```
# svcadm enable svc:/network/lldp:default
```

3 **LLDP エージェントが Rx および Tx モードで実行されていることを確認します。**

```
# lldpadm show-agentprop -p mode net0
```

LLDP エージェントが両方のモードで有効になっていない場合は、次のように入力します。

```
# lldpadm set-agentprop -p mode=both net0
```

詳細は、124 ページの「LLDP の SMF プロパティー」を参照してください。

ほかの取り得る LLDP エージェント構成については、127 ページの「システムでの LLDP の有効化」を参照してください。

4 ベースとなる **NIC** が **DCB** をサポートしていることを確認します。

```
# dladm show-linkprop -p ntcs net0
```

ゼロ (`0`) より大きいプロパティー値は、NIC が DCB をサポートしていることを示します。

優先順位ベースのフロー制御

PFC では、IEEE 802.1p の CoS 値を含むように標準 PAUSE フレームが拡張されます。PFC では、PAUSE フレームの送信時にリンク上のすべてのトラフィックを停止する代わりに、PFC フレーム内で有効になっている CoS 値のトラフィックだけを停止します。PFC フレームは、トラフィックを一時停止する必要のある、有効になっている優先順位に対して送信されます。送信側ホストがその優先順位のトラフィックを停止する一方で、ほかの無効になっている優先順位のトラフィックは影響を受けません。PFC フレームで指定された時間間隔のあと、または、送信側ホストが別の PFC フレームを受信したあと、それらのパケットの伝送が再開されます。優先順位に基づく一時停止により、その優先順位のパケットが破棄されることはなくなります。優先順位が定義されていないパケットに対しては、PAUSE フレームは送信されません。したがって、トラフィックフローは継続し、トラフィックの輻輳時にパケットが破棄される可能性があります。

優先順位は、`pfcmap` データリンクプロパティーの 8 ビットマスク (0–7) で表されます。最下位ビットは優先順位 0 を表し、最上位ビットは優先順位 7 を表します。このマスクの各ビットは、それに対応する優先順位に対して PFC が有効かどうかを示します。デフォルトでは、`pfcmap` は `1111111` に設定され、したがって、すべての優先順位に対して PFC が有効になります。受信側ホストで輻輳が発生した場合、リンク上で伝送される任意のパケットについて、送信側ホストに PFC フレームが送信されます。

PFC 関連のデータリンクプロパティー

`pfcmap` プロパティーのほかに、次のプロパティーにより、優先順位の定義とマッピングに関する情報が提供されます。

- `pfcmap-lcl-effective` は、ローカルホストで有効になっている PFC マッピングを示します。このプロパティーは読み取り専用です。このプロパティーは、`pfcmap` プロパティーの値または `pfcmap-rmt-effective` プロパティーの値のどちらかを反映します。
- `pfcmap-rmt-effective` は、リモートピアで有効になっている PFC マッピングを示します。このプロパティーも読み取り専用です。

PFC フレームが正しく送信されるには、通信を行うホストの DCB 構成情報が対称でなければなりません。Oracle Solaris 11 システムは、リモートピアの PFC 構成と一致するように自身の PFC 構成を自動的に調整できます。

これら 2 つのプロパティーは、ピア間で PFC 情報が同期されているかどうかを間接的に示します。ローカルピアとリモートピアの間で PFC 情報が一致しているデータリンクの場合、`pfcmap` に設定されている値にかかわらず、`pfcmap-lcl-effective` と `pfcmap-rmt-effective` の値は同じになります。ローカルホストで同期機能が無効になっている場合、`pfcmap-lcl-effective` にはローカルホストの `pfcmap` プロパティーの値が反映されます。

これらのプロパティー構成によって提供される PFC 情報の例については、143 ページの「PFC 構成情報の取得」を参照してください。

優先順位ベースのフロー制御 TLV ユニット

PFC TLV ユニットは、ホストがピアホストから受信した情報に関してどのように動作するかを制御します。この TLV ユニットには構成可能なプロパティーが 1 つだけあり、それは `willing` です。デフォルトでは、このプロパティーは `on` に設定され、ローカルホストは自身の PFC 優先順位定義をリモートピアの PFC 定義と同期することができます。特定のエージェントの情報の自動同期を防止するには、次のようにプロパティーを `off` に切り替えます。

```
# lldpadm set-agenttlvprop -p willling=off -a agent pfc
```

この *agent* は、エージェントが有効になっているデータリンクによって識別されます。

▼ DCB の優先順位ベースのフロー制御をカスタマイズする方法

ほとんどの場合、PFC のデフォルトの構成で十分です。この構成は LLDP が有効になっている場合に自動的に設定されます。ただし、PFC の構成時に使用できる各種オプションを示すために、この手順では手動の PFC 構成手順を示します。この手順では、自動構成は存在しないものとします。手順を理解しやすくするために、構成はすべて `net0` に対して実行します。

1. **DCBX** が有効になっていることを確認します。
   140 ページの「DCBX を有効にする方法」を参照してください。

2. **(オプション)** 有効にする **DCB** 機能をカスタマイズします。
   デフォルトでは、PFC、ETS、およびエッジ仮想ブリッジング (EVB) が有効になります。PFC だけを使用すると仮定します。したがって、ほかの 2 つの値を LLDP エージェントの `dot1-tlv` プロパティーから削除する必要があります。`dot1-tlv` の取り得る値のリストについては、表 7–3 を参照してください。

   ```
   # lldpadm set-agenttlvprop -p dot1-tlv-=etscfg,evb net0
   ```

3. データリンクの **`flowctrl`** プロパティーが **`pfc`** に設定されていることを確認します。

   ```
   # dladm show-linkprop -p flowctrl net0
   ```

   プロパティーの値のリストに `pfc` が含まれていない場合は、次のコマンドを発行します。

   ```
   # dladm set-linkprop -p flowctrl=pfc net0
   ```

4. **`pfcmap`** プロパティーのデフォルト値 **`11111111`** を使用したくない場合は、適宜設定します。
   たとえば、CoS 優先順位 6 だけを有効にするには、次のコマンドを入力します。

   ```
   # dladm set-linkprop -p pfcmap=01000000 net0
   ```

5. ホストが **PFC** 情報をリモートピアの **PFC** 情報と同期できることを確認します。

   ```
   # lldpadm show-agenttlvprop -p willing -a net0 pfc
   ```

   PFC TLV プロパティー `willing` が `off` に設定されている場合は、次のコマンドを発行します。

   ```
   # lldpadm set-agenttlvprop -p willing=on -a net0 pfc
   ```

PFC 構成情報の取得

このセクションでは、LLDP と DCB の構成後の PFC に関連する情報の例をいくつか示します。

次のコマンドは、PFC に関連する情報を表示します。

- `dladm show-linkprop -p pfcmap,pfc-lcl-effective,pfc-rmt-effective` *datalink*
  このコマンドは、優先順位の定義およびデータリンク上で有効になっている PFC マッピングを表示します。
- `dladm show-phys -D pfc` *datalink*
  このコマンドは、NIC で有効になっている優先順位に関連する、物理リンクの PFC 情報を表示します。

- `lldpadm show-agenttlvprop -a` *agent* `pfc`

  この *agent* は、LLDP が有効になっているデータリンクによって識別されます。したがって、LLDP エージェントの名前はデータリンクの名前と同じです。このコマンドは、ホストが PFC マッピングをピアと同期する機能を制御する PFC TLV プロパティーを表示します。

- `lldpadm show-agent -lv -o "PFC Pending"` *agent*

  このコマンドは、ローカルホストとピアの間で PFC マッピング情報が一致しないことをユーザーに警告します。

次の例は、前述のコマンドで表示される情報の種類を示します。

例 8–1 PFC 関連のデータリンクプロパティーの表示

この例では、優先順位ベースのフロー制御に関連するデータリンクプロパティーのステータスを表示する方法を示します。

```
# dladm show-linkprop -p pfcmap,pfc-lcl-effective,pfc-rmt-effective net0
LINK   PROPERTY               PERM   VALUE      DEFAULT    POSSIBLE
net0   pfcmap                 rw     11111111   11111111   00000000-11111111
net0   pfcmap-lcl-effective   r-     11111111   --         --
net0   pfcmap-rmt-effective   r-     01000000   --         --
```

出力は、ローカルホストの PFC マッピングがデフォルト値で、8 つの優先順位がすべて有効になっていることを示しています。`pfcmap-lcl-effective` と `pfcmap-rmt-effective` の値が一致していないことは、ローカルホストが PFC 情報をリモートピアと同期していないことを示しています。この不一致の原因として考えられるのは、同期を有効にするプロパティーがオフになっていることです。あるいは、ピアが PFC TLV ユニットをネットワークに送信していません。次のコマンドを入力して、この構成を確認できます。

例 8–2 ローカルホストで PFC 情報を同期する機能の表示

この例では、ホストがピアの PFC 構成に合わせて調整する機能の、現在のステータスを表示する方法を示します。

```
# lldpadm show-agenttlvprop -a net0 pfc
AGENT   TLVNAME   PROPERTY   PERM   VALUE   DEFAULT   POSSIBLE
net0    pfc       willing    rw     off     on        on,off
```

同期を有効にするには、次のコマンドを発行します。

```
# lldpadm set-agenttlvprop -p willing=on -a net0 pfc

# dladm show-linkprop -p pfcmap,pfc-lcl-effective,pfc-rmt-effective net0
LINK   PROPERTY               PERM   VALUE      DEFAULT    POSSIBLE
net0   pfcmap                 rw     11111111   11111111   00000000-11111111
net0   pfcmap-lcl-effective   r-     01000000   --         --
net0   pfcmap-rmt-effective   r-     01000000   --         --
```

例 8–2 ローカルホストで PFC 情報を同期する機能の表示 *(続き)*

2 番目の出力で、ローカルホストは自身の PFC マッピング (11111111) を破棄しています。代わりに、ローカルホストはピアと同期して、有効な PFC マッピングはピアの PFC マッピングと同じになっています。この値の収束により、ホストは PFC PAUSE フレームを正常に交換できます。

例 8–3 ホストとピアの間で PFC 情報が対称であることの確認

この例では、実際の実行時にホストとピアの間で PFC 情報が同期されているか、あるいは不一致が発生しているかを確認する方法を示します。

```
# lldpadm show-agent -lv -o "PFC Pending" net0
PFC Pending: True
```

`PFC Pending` は、ホストとピアの間で PFC 情報が収束しない場合に `True` ステータスを返します。不一致が解決されたあとは、`PFC Pending` のステータスは `False` に戻ります。

エージェントによって通知されるすべての情報を表示するには、`lldpadm show-agent` コマンドの `verbose` オプションを使用します。

```
# lldpadm show-agent -v agent
```

例 8–4 CoS 優先順位定義の表示

この例では、特定のデータリンクの現在の CoS 優先順位定義を `pfcmap` プロパティーの値に基づいて表示する方法を示します。たとえば、`pfcmap` が `01000000` に構成されているとします。対応する物理リンクの優先順位マッピングを表示するには、次のように操作します。

```
# dladm show-phys -D pfc net0
LINK        COS   PFC   PFC_EFFECT   CLIENTS
ixgbe0      0     YES           NO   net0,vnic1
            1     YES          YES   vnic2
            2     YES           NO   vnic3
            3     YES           NO   vnic4
            4     YES           NO   vnic5
            5     YES           NO   vnic6
            6     YES           NO   vnic7
            7     YES           NO   vnic8
```

物理リンク `net0` では、データリンク上に構成されているすべての VNIC クライアントに対して優先順位が有効になっています。ただし、ローカルホストは自身の PFC マッピングをピアの PFC マッピングに合わせて調整するため、`PFC_EFFECT` フィールドの値で示されているように、CoS `0` および `2-7` では優先順位が無効になっています。したがって、`vnic2` を除くすべての VNIC のトラフィックに対して、リソースが使用可能かどうかにかかわらず、PFC フレームの交換は行われません。この構成では、`vnic2` 以外のすべての VNIC を通過するトラフィックで、パケットの破棄が許可されます。`vnic2` のトラフィックに対しては、トラフィックの輻輳が発生すると、こ

例 8–4 CoS 優先順位定義の表示 (続き)

のクライアントでのパケットロスを防止するために PFC PAUSE フレームが送信されます。

アプリケーション TLV ユニット

アプリケーション TLV ユニットには、ホスト上のアプリケーションに使用する優先順位の情報が含まれています。優先順位はアプリケーション優先順位表で定義されます。表の各エントリには、アプリケーションの名前およびアプリケーションに割り当てられた優先順位が含まれます。アプリケーション TLV はこの表をほかのホストとのアプリケーション優先順位情報の伝送に使用します。

表のエントリには次の形式が使用されます。

protocol-id/*selector*/ *priority*

protocol-id と *selector* のペアによってアプリケーションが識別されます。*Priority* には、対応するアプリケーションの優先順位を示す 0 から 7 までの値が含まれます。

アプリケーションの優先順位に関するこの情報をほかのホストと交換するには、アプリケーション TLV を次のように設定します。

```
# lldpadm set-agenttlvprop -p property=value -a agent appln
```

たとえば、FCoE トラフィックの場合、プロトコル ID は `0x8906`、セレクタ ID は `1` です。このアプリケーションに優先順位 `4` が割り当てられているとします。アプリケーション TLV の設定用パラメータを示す 表 7–3 に基づいて、次のコマンドを入力します。

```
# lldpadm set-agenttlvprop -p apt=8906/1/4 -a net0 appln
# lldpadm show-agenttlvprop -a net0 appln
AGENT   TLVNAME   PROPERTY   PERM   VALUE      DEFAULT   POSSIBLE
net0    appln     apt        rw     8906/1/4   --        --
```

拡張伝送選択

ETS は、アプリケーションの DCB 優先順位に基づいて NIC 上の帯域幅をアプリケーションに割り当てることができる DCB 機能です。DCB 優先順位は、3 ビットの優先順位フィールドを持つ VLAN ヘッダーです。優先順位フィールドの値により、ネットワークの Ethernet パケットが差別化されます。DCB は、802.1p 優先順位とも呼ばれるこの優先順位値を使用して、PFC 構成やリンク帯域幅といったほかの DCB プロパティーにトラフィックを関連付けます。DCB を構成して、パケットの優先順位値に応じてパケットに割り当てる帯域幅を設定します。

ETS を使用するには、NIC が DCB をサポートしており、DCB モードで動作している必要があります。

ETS 関連のデータリンクプロパティー

PFC 情報に関連するデータリンクプロパティーは、パケットに定義されている CoS 優先順位に基づくパケットロス防止に適用されます。ETS 情報に関連するプロパティーは、同じ CoS 優先順位に基づくパケットへの共有帯域幅割り当てに適用されます。次のデータリンクプロパティーで ETS を構成します。

- `cos` は、データリンクのサービスクラスを指定します。このプロパティーは Ethernet 優先順位を表します。このプロパティーは 0 から 7 までの値を取り、データリンクのアウトバウンドパケットに適用されます。値はアウトバウンドパケットの VLAN タグに設定されます。このプロパティーが物理リンク自体に設定された場合、優先順位はそのリンクのプライマリクライアントのトラフィックだけに適用されます。VNIC など、ほかのセカンダリクライアントには優先順位は設定されません。NIC が DCB モードで動作している場合、またはリンクが VLAN の場合、`cos` はデフォルトで `0` に設定されます。
- `etsbw-lcl` は、データリンクの TX 側に割り当てられた ETS 帯域幅を示します。ベースとなる物理 NIC が DCB 機能を持ち、ETS をサポートしている場合のみ、このプロパティーは構成可能です。値を設定するには、セカンダリデータリンクまたはクライアントに割り当てる帯域幅を、ベースとなる NIC の合計帯域幅の割合で指定します。リンクの `cos` がゼロ (`0`) に設定されていない場合に、このプロパティーを設定できます。

注 – アグリゲーションに構成されている DCB モードの物理リンクでは、ETS は現在サポートされていません。

`etsbw-lcl` に定義される帯域幅の割合は、そのセカンダリクライアントだけに予約された量ではありません。割り当てられた帯域幅が使用されていない場合は、同様に構成されているほかのクライアントがそれを使用できます。また、帯域幅割り当ては、ホストのトラフィックの送信側だけに適用されます。

上記のリストのプロパティーに加え、次の読み取り専用プロパティーにより、ローカルホストとそのピアの間で交換される帯域幅データに関する情報が提供されます。

- `etsbw-lcl-advice` は、推奨される共有帯域幅を指定します。データリンクのこの推奨帯域幅は、リモートピアからローカルホストに送信されます。
- `etsbw-lcl-effective` は、ローカルホストのデータリンクに実装されている実際の共有帯域幅を示します。このプロパティーは、`etsbw-lcl` プロパティーの値または `etsbw-lcl-advice` プロパティーの値のどちらかを反映します。
- `estbw-rmt-effective` は、リモートピアに構成されている共有帯域幅を示します。

特定の優先順位を持つパケットに適切な帯域幅が使用されるようにするには、通信を行うホスト間で ETS 情報が対称になっているか同期されている必要があります。特に、ローカルシステムには etsbw-lcl-advice の値に合わせて自身の共有帯域幅を調整する機能があることが望ましいです。Oracle Solaris 11 システムは、リモートピアの ETS 構成と一致するように自身の ETS 構成を自動的に調整できます。

estbw-lcl-effective プロパティーは、ローカルホストが ETS 情報をピアと一致させる機能が有効になっているかどうかを間接的に示します。このプロパティーの値が etsbw-lcl-advice の値と一致している場合、機能は有効になっています。それ以外の場合、etsbw-lcl-effective プロパティーと etsbw-lcl プロパティーの値は同じになります。

拡張伝送選択 TLV ユニット

ETS TLV ユニット etscfg は、ホストがピアホストから受信した情報に関してどのように動作するかを制御します。この TLV ユニットには構成可能なプロパティーが 1 つだけあり、それは willing です。デフォルトでは、このプロパティーは on に設定され、ローカルホストは自身の ETS 構成をリモートピアの ETS 構成と同期することができます。特定のエージェントの情報の同期を防止する必要がある場合は、次のように willing プロパティーを off に設定します。

```
# lldpadm set-agenttlvprop -p willing=off -a agent etscfg
```

この *agent* は、エージェントが有効になっているデータリンクによって識別されます。

▼ DCB の拡張伝送選択をカスタマイズする方法

ほとんどの場合、システムの ETS のデフォルトの構成で十分です。LLDP が有効になっている場合で、ベースとなるリンクが DCB をサポートしており、DCB モードで動作しているとき、この構成は自動的に設定されます。ただし、ETS の構成時に使用できる各種オプションを示すために、この手順では手動の ETS 構成手順を示します。この手順では、自動構成は存在しないものとし、仮想クライアント vnic1 に対して構成を実行するものとします。仮想クライアントは LLDP エージェントである net0 経由で構成されます。

1 **DCBX が有効になっていることを確認します。**

140 ページの「DCBX を有効にする方法」を参照してください。

2 (オプション) 有効にする DCB 機能をカスタマイズします。

デフォルトでは、PFC、ETS、およびエッジ仮想ブリッジング (EVB) が有効になります。EVB を無効にすると仮定します。したがって、ほかの 2 つの値を LLDP エージェントの `dot1–tlv` プロパティーから削除します。

```
# lldpadm set-agenttlvprop -p dot1-tlv-=evb net0
```

3 VNIC に CoS 優先順位定義を設定します。

```
# dladm set-linkprop -p cos=value vnic1
```

4 物理リンクの合計帯域幅と共有する VNIC の帯域幅を設定します。

```
# dladm set-linkprop -p etsbw-lcl=value vnic1
```

`etsbw-lcl` プロパティーに割り当てる値は、ベースとなるリンクの合計帯域幅容量に対する割合を表します。クライアントに割り当てるすべての割り当て帯域幅値の合計は、100 パーセントを超えてはいけません。

5 ホストが ETS 情報をリモートピアの ETS 情報と同期できることを確認します。

```
# lldpadm show-agenttlvprop -p willing -a net0 etscfg
```

`willing` プロパティーが `off` に設定されている場合は、次のコマンドを発行します。

```
# lldpadm set-agenttlvprop -p willing=on -a net0 etscfg
```

ETS 構成情報の取得

このセクションでは、LLDP と DCB の構成後の ETS 構成に関連する情報の例をいくつか示します。

次のコマンドは、ETS 構成に関する情報を表示します。

- `dladm show-linkprop -p etsbw-lcl,etsbw-advise,etsbw-lcl-effective,etsbw-rmt-effective` *datalink*

  このコマンドは、帯域幅割り当ての定義およびデータリンク上で有効になっている割り当てを表示します。

- `dladm show-phys -D ets` *datalink*

  このコマンドは、リンク上の帯域幅の割り当てと分配に関連する、物理リンクの ETS 構成を表示します。

- `lldpadm show-agenttlvprop -a` *agent* `etscfg`

  この *agent* は、LLDP が有効になっているデータリンクによって識別されます。このコマンドは、ホストが ETS 情報をピアと同期する機能を制御する ETS TLV プロパティーを表示します。

次の例は、前述のコマンドで表示される情報の種類を示します。

例 8–5 ETS 関連のデータリンクプロパティーの表示

この例では、拡張伝送選択に関連するデータリンクプロパティーのステータスを表示する方法を示します。

```
# dladm show-linkprop -p cos,etsbw-lcl,etsbw-lcl-advise, \
etsbw-lcl-effective,etsbw-rmt-effective vnic1
LINK    PROPERTY              PERM  VALUE      DEFAULT    POSSIBLE
vnic1   cos                   rw    2          0          0-7
vnic1   etsbw-lcl             rw    20         0          --
vnic1   etsbw-lcl-advise      r-    --         --         --
vnic1   etsbw-lcl-effective   r-    --         --         --
vnic1   etsbw-rmt-effective   r-    --         --         --
```

出力は、物理リンクで使用できる合計帯域幅の 20% が vnic1 の共有帯域幅になるように構成されていることを示しています。cos プロパティーで示される VNIC の 802.1p 優先順位は、2 に設定されています。

例 8–6 ローカルホストで ETS 情報を同期する機能の表示

この例では、ホストがピアの ETS 構成に合わせて調整する機能の、現在のステータスを表示する方法を示します。

```
# lldpadm show-agenttlvprop -a net0 etscfg
AGENT   TLVNAME   PROPERTY   PERM   VALUE   DEFAULT   POSSIBLE
net0    etscfg    willing    rw     off     on        on,off
```

同期を有効にするには、次のコマンドを発行します。

```
# lldpadm set-agenttlvprop -p willing=on -a net0 etscfg

# dladm show-linkprop -p etsbw-lcl,etsbw-lcl-advise, \
etsbw-lcl-effective,etsbw-rmt-effective vnic0
LINK    PROPERTY              PERM  VALUE      DEFAULT    POSSIBLE
vnic1   cos                   rw    2          0          0-7
vnic1   etsbw-lcl             rw    20         0          --
vnic1   etsbw-lcl-advise      r-    15         --         --
vnic1   etsbw-lcl-effective   r-    15         --         --
vnic1   etsbw-rmt-effective   r-    25         --         --
```

vnic1 の estbw-lcl は 20% に設定されていましたが、VNIC で有効になっている共有帯域幅は、ピアから通知された帯域幅と一致する 15% になっています。この調整は、etscfg TLV ユニットの willing プロパティーを on に切り替えた結果として発生します。

次の例は、物理リンク上の優先順位マッピングを表示します。

```
# dladm show-phys -D ets net0
LINK        COS   ETSBW   ETSBW_EFFECT   CLIENTS
ixgbe0      0     20      20             <default,mcast>,net0
            1     15      15             vnic2
            2     20      20             vnic1
```

```
3     30      30              vnic5
4     15      15              vnic3
5     0       0               vnic4
6     0       0               vnic6
7     0       0               vnic7
```

この例では、各 VNIC にそれぞれ対応する cos 値が設定されています。上記の出力に基づくと、vnic1 の cos プロパティーは 2 に設定されています。ETSBW フィールドでは、クライアント vnic1 で有効になっている共有帯域幅は、ピアから通知された値と一致する 15% になっており、これは ETSBW_EFFECT フィールドで示されています。この例では、最大の共有帯域幅が vnic5 に割り当てられていることも示されています。vnic4、vnic6、および vnic7 には 0% が割り当てられていますが、これは、これらのクライアントが帯域幅をまったく共有しないことを意味するわけではありません。ほかのクライアントが割り当てられた帯域幅を使用している場合、これらのクライアントは帯域幅を受け取りません。

◆ ◆ ◆ 第 9 章

9 Oracle Solaris でのエッジ仮想ブリッジング

この章では、エッジ仮想ブリッジング (EVB) の機能について説明します。EVB は、次の章で説明されている情報交換機能をさらに拡張します。

- 第 7 章「LLDP によるネットワーク接続情報の交換」
- 第 8 章「Oracle Solaris におけるデータセンターブリッジング機能の操作」

EVB により、システムの仮想リンクに関連する情報の交換が可能になります。この章の内容は次のとおりです。

エッジ仮想ブリッジングの概要

エッジ仮想ブリッジングは、ホストが仮想リンクの情報を外部スイッチと交換できるようにする、発展中の IEEE 規格です。EVB では、DCB 機能で提供される物理リンクの共有帯域幅や優先順位定義などのほかに、仮想リンクの構成に関するより詳細な情報をネットワーク上に通知することができます。

一般に、EVB は次の操作に使用できます。

- 外部ブリッジポートの反射型リレーを有効にします。154 ページの「反射型リレー機能」を参照してください。
- ブリッジの仮想ポート構成を自動化します。154 ページの「ブリッジの自動仮想ポート構成」を参照してください。

EVB のメカニズムを理解するには、EVB で使用される次の用語に注意してください。

- ステーションは、システムまたはホストを指します。
- ブリッジは、ステーションに接続されている外部スイッチを指します。

- 仮想ステーションインスタンス **(VSI)** は、ステーション上に構成されている VNIC を指します。
- 仮想マシン **(VM)** は、ゾーンや Oracle VM VirtualBox など、ソフトウェアによってシステムに実装されるマシンを指す一般的な用語です。

次のセクションでは、EVB の機能について詳細に説明します。

反射型リレー機能

ネットワークの仮想化では、ステーションの単一の物理 NIC 上に複数の仮想ネットワークインタフェースカード (VNIC) を構成できます。VNIC はステーション上の仮想マシンに割り当てられます。このセットアップでは、パケットをステーションの外部に送信せずに仮想マシン間の通信を行うことができます。代わりに、物理リンクの仮想スイッチによって、VM から別の VM にパケットがルーティングされます。VNIC、仮想スイッチ、および仮想マシンを含むシステム構成の図については、『Oracle Solaris 11.1 での仮想ネットワークの使用』の「ネットワーク仮想化のコンポーネント」を参照してください。

場合によっては、VM 間の内部通信に外部ブリッジが必要になることがあります。たとえば、外部ブリッジに構成されているアクセス制御リスト (ACL) に従って内部通信を行う必要がある場合などです。したがって、送信側 VM からのパケットは、ポートを通してステーションから外部ブリッジに送信されます。次に、これらのパケットはブリッジからステーションに戻され、受信側 VM に送信されます。

デフォルトでは、ブリッジはパケットを受信した同じポートでパケットを送信することはできません。したがって、外部ブリッジを使用する VM 間の通信では、ブリッジに反射型リレー機能が必要です。この機能により、ブリッジは送信側 VM からのパケットを、それらのパケットを受信した同じリンクで、受信側 VM に中継することができます。

EVB は、反射型リレー機能についてネットワークピアに通知するために、組織固有の新しい LLDP TLV ユニットを定義します。一方、EVB TLV ユニットは情報の伝達媒体として機能します。情報交換では、ブリッジで反射型リレーがサポートされている場合、まずステーションがブリッジに反射型リレーを有効にするようリクエストします。ブリッジは、その機能をサポートしていればそれを有効にし、リクエスト側のホストにその機能について通知します。ブリッジが反射型リレーをサポートしていない場合は、無効ステータスがステーションに返送されます。その場合、仮想マシン間の通信には、単に物理リンクの仮想スイッチが使用されます。

ブリッジの自動仮想ポート構成

LLDP と DCBX により、ステーションはもっとも近いブリッジと構成情報を交換できます。ブリッジは、この交換で、ステーション上でトラフィッククラスに対して定

義されている優先順位などを検出できます。この情報に基づいて、パケットの802.1p優先順位値に従ってパケットを処理するようにブリッジが自動的に構成されます。

情報交換と自動構成がないとしたら、ステーションとは別にブリッジを手動で構成して、ステーション構成をトラフィッククラスの優先順位に反映させる必要があります。手動構成では、ブリッジの構成を誤るリスクがあり、ステーションとブリッジの間に不整合が生じる可能性があります。

EVBを使用すると、交換メカニズムが拡張され、ブリッジに対してステーションのVSIに関する情報も含まれるようになります。このような方法で、VNICの構成をブリッジに伝えることができます。たとえば、VNICに特定の帯域幅制限が構成されているとします。EVBでは、ブリッジはそのVNIC宛てのパケットに帯域幅制限を適用できます。

VSI情報交換のための**EVB**コンポーネント

次のEVBコンポーネントにより、ステーションはVSI情報をブリッジに通知できます。

- **VSI**プロファイルは、特定のVNICに構成されているリンクプロパティーから成ります。したがって、ステーションは構成済みVNICと同じ数のVSIプロファイルを持つことができます。
- VSI識別子は、VSIプロファイルを一意に識別する、**VSI**タイプ**ID**と**VSI**バージョン**ID**のペアから成ります。
- **VSI**マネージャーは、VSIタイプID-VSIバージョンID識別子を特定のVNICプロパティーセットとマッピングすることにより、ステーション上の複数のVSIプロファイルを管理します。
- **VSI**マネージャー**ID**は、特定のVSIタイプID - VSIバージョンペアに関連するVSIマネージャーを識別します。VSIマネージャーIDはIPv6アドレスとして表されます。

VSIマネージャーID、VSIタイプID、およびVSIバージョンを組み合わせると、特定のVNICのプロパティーセットを識別するタプルになります。

VSI情報はVDP (VSI discovery and configuration protocol) を使用して交換され、VDP TLVユニットは情報の伝達媒体として機能します。ブリッジはステーションからVDP TLVユニットを受信します。ブリッジはTLVユニットに含まれているタプルを使用して、VSIに関連付けられているプロパティーセットを取得します。ブリッジは、VSIプロファイルまたはタイプのプロパティーを取得したあと、そのプロパティー構成をそのVSI宛てのパケットに適用できます。

VSI情報をブリッジに通知するには、まず次の要件を満たす必要があります。

- ステーションは、VSI発見プロトコルリクエストの送信時に使用するVSIマネージャーIDを知っている必要があります。

- ブリッジは、ステーションから送信される VSI マネージャー ID を認識でき、サポートしている必要があります。

Oracle Solaris での EVB のサポート

プロファイルにどのプロパティーを含めるべきかなど、VSI プロファイルの定義に関して定義済みの標準は現在存在しません。また、VSI タイプの定義は VSI マネージャー ID と密接に関連していますが、多くの場合 VSI マネージャー ID はベンダー固有です。

Oracle Solaris では、VSI マネージャーは 3 バイトエンコーディング `oracle_v1` を使用して定義されます。この VSI マネージャーは次のデータリンクプロパティーをサポートします。

- 帯域幅の制限
- ベースとなるリンクのリンク速度
- トラフィッククラス
- VNIC の最大転送単位 (MTU)

`oracle_v1` エンコーディングは次のように定義されています。

ビット	プロパティー
0-4	リンク帯域幅の制限 00000-10100 : リンク速度の 0-100% (5% 刻み)。 残り: 予約済み
5–7	リンク速度 000 - 不明 001 - 10M ビット/秒 010 - 100M ビット/秒 011 - 1G ビット/秒 100 - 10G ビット/秒 101 - 40G ビット/秒 110 - 100G ビット/秒 111 - 予約済み
8–12	予約済み
13-15	トラフィッククラス **(0-7)**

ビット	プロパティー
16-17	リンク **MTU** 00 - 1500 バイト 01 - 9000 バイト 10 - カスタム 11 - 予約済み

この 3 バイトエンコードは、Oracle Solaris で VSI タイプ ID として直接使用されます

したがって、Oracle Solaris では、Oracle VSI マネージャーおよび組み合わされた VSI タイプ ID-VSI バージョン ID ペアが、ブリッジに通知されるタプルになります。VSI 情報の交換メカニズムは、155 ページの「VSI 情報交換のための EVB コンポーネント」で説明されているものと同じプロセスに従います。ブリッジは Oracle VSI マネージャーを認識するように構成されます。ブリッジは次に、Oracle VSI マネージャー ID および組み合わされた VSI タイプ ID-VSI バージョン ID ペアを使用して、VSI プロファイルに関連付けられているプロパティーセットを取得します。ブリッジは、プロパティー情報を取得したあと、そのプロパティー構成をその VNIC 宛てのパケットに適用できます。

VSI マネージャー ID TLV の送信後、Oracle 組織固有の OUI TLV ユニットが送信されます。OUI TLV は、それに先行する VSI マネージャー ID にいずれかのエンコーディングが使用されていれば、それを示します。ブリッジは、この Oracle 定義の VSI マネージャー ID を認識した場合、リクエスト側のステーションに応答するときにその TLV ユニットを含めます。ブリッジの応答に Oracle 固有の TLV ユニットが含まれていない場合、そのスイッチでは Oracle VSI マネージャーが認識もサポートもされていないことを意味します。

EVB 関連のデータリンクプロパティー

次は、EVB に関連する構成可能なデータリンクプロパティーのリストです。

- `vsi-mgrid` は、物理リンクまたは VNIC のどちらかに設定される VSI マネージャー ID を指定します。Oracle Solaris では、このプロパティーはデフォルトの VSI マネージャー ID `ORACLE_VSIMGR_V1` に関連付けられます。

  IPv6 アドレスを使用する場合は、VSI タイプ ID と VSI バージョン ID も定義する必要があります。それ以外の場合、タプルは Oracle Solaris で認識されません。さらに、VSI タイプ ID-VSI バージョン ID/VSI マネージャー ID タプルに対応する適切なデータリンクプロパティーを手動で構成する必要があります。

  できれば、EVB を使用するときはデフォルトの Oracle VSI マネージャー ID を使用するようにしてください。そうすれば、ステーションの VSI プロファイルの VSI タイプ ID と VSI バージョン ID を Oracle VSI マネージャーで自動的に生成できます。

- vsi-mgrid-enc は、VSI マネージャー ID に関連付けられているエンコーディングを示します。デフォルトでは、このプロパティーは oracle_v1 に設定されます。VSI マネージャー ID に oracle_v1 を関連付けない場合は、このプロパティーを値 none に設定してください。
- vsi-typeid は VSI タイプ ID を指定します。VSI タイプ ID は、VSI バージョン ID とのペアで VSI プロファイルに関連付けられます。vsi-mgrid と vsi-mgrid-enc にデフォルト値を使用している場合、この 3 バイト値は自動的に生成されます。それ以外の場合は、このプロパティーの値を明示的に指定する必要があります。
- vsi-vers は VSI バージョン ID を指定します。VSI バージョン ID は、VSI タイプ ID とのペアで VSI プロファイルに関連付けられます。vsi-mgrid と vsi-mgrid-enc にデフォルト値を使用している場合、この 1 バイト値は自動的に生成されます。それ以外の場合は、このプロパティーの値を明示的に指定する必要があります。

注 – これらすべてのプロパティーは、すべての VNIC に手動で構成できますが、物理リンクに構成できるのは vsi-mgrid および vsi-mgrid-enc プロパティーだけです。

上記のリストのプロパティーに加え、次の読み取り専用プロパティーにより、システムで有効になっている実際の EVB 構成に関する情報が提供されます。

- vsi-mgrid-effective は、仮想リンクまたは VNIC 上の VSI マネージャー ID を指定します。
- vsi-mgrid-enc-effective は、仮想リンクまたは VNIC に使用されている、VSI マネージャー ID の基となる VSI マネージャー ID エンコーディングを示します。
- vsi-typeid-effective は、仮想リンクまたは VNIC 上で有効になっている VSI タイプ ID を指定します。
- vsi-vers-effective は、リンク上で有効になっている VSI バージョンを指定します。

ステーションでの EVB の使用

ステーションで EVB を使用するには、EVB パッケージをインストールする必要があります。次のコマンドを入力します。

```
# pkg install evb
```

できれば、パッケージのインストール後に自動的に有効になるデフォルトの EVB 構成を受け入れるようにしてください。EVB 構成では、EVB の有効化に Oracle VSI マネージャーを使用することを基にしています。デフォルトの EVB 構成を受け入れると、ステーションはただちに、ステーション上に構成されている VNIC に関する VSI 情報をブリッジと交換できるようになります。

次の例は、物理リンク上の EVB 関連のプロパティーを示しています。

```
# dladm show-linkprop -p vsi-mgrid,vsi-mgrid-enc
LINK     PROPERTY            PERM VALUE        DEFAULT      POSSIBLE
net4     vsi-mgrid           rw   --           ::           --
net4     vsi-mgrid-enc       rw   --           oracle_v1    none,oracle_v1
```

出力には、Oracle Solaris 11 でのデフォルトの EVB 構成が表示されています。`oracle_v1` エンコーディングを使用すると、Oracle VSI マネージャーが認識できサポートしている VSI とそのデータリンクプロパティーが、Oracle VSI マネージャーで管理されます。

デフォルトの構成を使用しない場合は、エンコーディングを none に設定してください。

```
# dladm set-linkprop -p vsi-mgrid-enc=none net4
```

その後、VSI マネージャー ID として使用する IPv6 アドレスを手動で指定し、VSI タイプ ID とその他すべての EVB 関連コンポーネントおよびそれらのプロパティーを定義する必要があります。

次の例は、VSI または VNIC 上の EVB 関連のプロパティーを示しています。

```
# dladm show-linkprop vnic0
LINK     PROPERTY                PERM VALUE      DEFAULT  POSSIBLE
...
vnic0    vsi-typeid              rw   --         --       --
vnic0    vsi-typeid-effective    r-   65684      --       --
vnic0    vsi-vers                rw   --         --       --
vnic0    vsi-vers-effective      r-   0          --       --
vnic0    vsi-mgrid               rw   --         --       --
vnic0    vsi-mgrid-effective     r-   ::         --       --
vnic0    vsi-mgrid-enc-effective r-   oracle_v1  --       --
...
```

出力には、Oracle VSI マネージャーの使用に基づく値が表示されています。VSI の VSI マネージャー ID で有効になっているエンコーディングは `oracle_v1` です。次に、`vnic0` にはタイプ ID `65684` が自動的に生成され、有効になっています。

次の例は、ステーションで EVB が有効になっている場合の物理 Ethernet リンクの VDP 状態に関する情報を示しています。単一のリンクに関する情報だけを表示するには、そのリンクをコマンドで指定します。それ以外の場合、すべての Ethernet リンクに関する VDP 情報が表示されます。

```
# dladm show-ether -P vdb
VSI      LINK   VSIID             VSI-TYPEID VSI-STATE   CMD-PENDING
vnic0    net4   2:8:20:2c:ed:f3   65684/0    TIMEDOUT    NONE
vnic1    net4   2:8:20:df:73:77   65684/0    TIMEDOUT    NONE
```

出力は、リンク net4 上に 2 つの VSI が構成されていることを示しています。それぞれ固有の VSI ID は、それぞれに対応する MAC アドレスを示しています。vsi-mgrid のデフォルト値に基づき、どちらの VSI も同じ VSI タイプ ID、すなわち 65684 になっています。

発信または着信 VDP パケットに関する統計を取得するには、次のコマンドを使用します。

```
# dlstat show-ether -P vdb
```

◆ ◆ ◆ 第 10 章

10 統合ロードバランサ (概要)

この章では、Oracle Solaris の機能である統合ロードバランサ (ILB) について説明します。ILB は、SPARC および x86 ベースのシステムにインストールされている Oracle Solaris にレイヤー 3 およびレイヤー 4 の負荷分散機能を提供します。ILB はクライアントからの受信リクエストを傍受し、リクエストを処理するバックエンドサーバーを負荷分散規則に基づいて決定し、選択されたサーバーにリクエストを転送します。ILB はオプションの健全性検査を実行し、選択されたサーバーが受信リクエストを処理できるかどうかを確認するために負荷分散アルゴリズムのデータを提供します。ILB は上記の機能を実行することによって、サーバーに向けられた作業負荷を複数のサーバーに分散します。これによって信頼性を向上させ、応答時間を最小限に抑え、サーバーのパフォーマンスを全体的に向上させることができます。

次の内容について説明します。

ILB の機能

ILB の主な機能は次のとおりです。

- IPv4 および IPv6 について、ステートレス Direct Server Return (DSR) およびネットワークアドレス変換 (NAT) の動作モードをサポートします
- コマンド行インタフェース (CLI) による ILB 管理を可能にします
- 健全性検査によるサーバーモニタリング機能を提供します

ILB の追加機能は次のとおりです。

- クライアントが仮想 **IP (VIP)** アドレスを **ping** できる – ILB は、仮想サービスの IP アドレスに対するクライアントからのインターネット制御メッセージプロトコル (ICMP) エコーリクエストに応答できます。ILB は、DSR および NAT の動作モードについてこの機能を提供します。
- サービスを中断せずにサーバーをサーバーグループに追加および削除できる – バックエンドサーバーと確立されている既存の接続を中断させずにサーバーをサーバーグループに動的に追加および削除できます。ILB は、NAT 動作モードについてこの機能を提供します。
- セッション持続性 (スティッキネス) を構成できる – 多くのアプリケーションでは、同一のクライアントからの一連の接続、パケット、あるいはその両方が同一のバックエンドサーバーに送信されることが重要です。`ilbadm create-rule` サブコマンドで `-p` オプションを使用し、`pmask` を指定することによって、仮想サービスのセッション持続性 (つまり、発信元アドレスの持続性) を構成できます。詳細は、193 ページの「ILB 規則を作成する方法」を参照してください。持続性マッピングが作成されると、その後に続く仮想サービスへの接続またはパケット、あるいはその両方のリクエストは、クライアントの発信元 IP アドレスが一致する場合、同一のバックエンドサーバーに転送されます。クラスレスドメイン間ルーティング (CIDR) 表記の接頭辞長は、IPv4 では `0-32`、IPv6 では `0-128` の間の値です。セッション持続性のサポートは、DSR と NAT の両方の動作モードで利用できます。
- 接続排出を実行できる – ILB は、NAT ベースの仮想サービスのサーバーについてのみこの機能をサポートします。この機能は、無効にされているサーバーに対して新しい接続が送信されることを回避します。この機能は、アクティブな接続またはセッションを中断せずにサーバーをシャットダウンする場合に役立ちます。サーバーへの既存の接続は動作を継続します。そのサーバーへのすべての接続が終了したあとに、サーバーは保守のためにシャットダウンできます。サーバーがリクエストを処理するための準備が整ったら、サーバーを有効にすることで、ロードバランサは新しい接続をサーバーに転送できるようになります。この機能によって、アクティブな接続またはセッションを中断せずに、サーバーを保守のためにシャットダウンできます。
- **TCP** および **UDP** ポートの負荷分散ができる – ILB は、各ポートに明示的な規則を設定しなくても、特定の IP アドレス上のすべてのポートを異なる一連のサーバーで負荷分散できます。ILB は、DSR および NAT の動作モードについてこの機能を提供します。
- 同一のサーバーグループ内で仮想サービス用の個別ポートを指定できる – この機能により、NAT 動作モードの場合、同一のサーバーグループ内の異なるサーバーについて異なる着信先ポートを指定できます。
- 単純なポート範囲の負荷分散ができる – ILB では、VIP 上のポート範囲を、特定のサーバーグループで負荷分散できます。便宜上、同一 VIP 上の異なるポート範囲を異なる一連のバックエンドサーバーで負荷分散することによって、IP アドレスを節約できます。また、NAT モードでセッション持続性が有効なときは、ILB は同一のクライアント IP アドレスから、範囲内のさまざまなポートへのリクエストを、同一のバックエンドサーバーに送信します。

- ポート範囲の移動および収縮ができる – ポート範囲の移動および収縮は、負荷分散規則内のサーバーのポート範囲に依存します。したがって、サーバーのポート範囲が VIP ポート範囲と異なる場合、ポート移動が自動的に実装されます。サーバーのポート範囲が単一ポートの場合、ポート収縮が実装されます。これらの機能は NAT 動作モードについて提供されます。

ILB のコンポーネント

ILB には 3 つの主要コンポーネントがあります。

- `ilbadm` CLI – このコマンド行インタフェースを使用して、負荷分散規則の構成、オプションの健全性検査の実行、および統計の表示が可能です。
- `libilb` 構成ライブラリ – `ilbadm` およびサードパーティーアプリケーションは、`libilb` 内に実装されている ILB 管理用の機能を使用できます。
- `ilbd` デーモン – このデーモンは次のタスクを実行します。
  - リブート後やパッケージ更新後にも持続する構成を管理します。
  - 構成情報を処理し、その情報を ILB カーネルモジュールに送信して実行することによって、ILB カーネルモジュールへの逐次アクセスを提供します
  - 健全性検査を実行し、結果を ILB カーネルモジュールに送信することで、負荷分散が正しく調整されます

ILB の動作モード

ILB は、1 脚または 2 脚のトポロジにおいて、IPv4 および IPv6 に対するステートレス Direct Server Return (DSR) およびネットワークアドレス変換 (NAT) の動作モードをサポートします。

- ステートレス DSR トポロジ
- NAT モード (フル NAT およびハーフ NAT) トポロジ

Direct Server Return トポロジ

DSR モードでは、ILB は受信リクエストをバックエンドサーバーに分散しますが、サーバーからクライアントへの戻りトラフィックは ILB をバイパスします。ただし、ILB をバックエンドサーバーのルーターとして使用するように設定することもできます。この場合、バックエンドサーバーからクライアントへの応答は、ILB を実行しているシステムを通るようにルーティングされます。ILB の現在の DSR 実装は TCP 接続追跡を提供しません (つまりステートレスです)。ステートレス DSR では、ILB は基本的な統計情報を除き、処理されるパケットの状態情報を保存しません。このモードでは ILB が状態を保存しないため、パフォーマンスは通常の IP 転送のパフォーマンスに匹敵します。このモードはコネクションレスプロトコルに最適です。

利点:

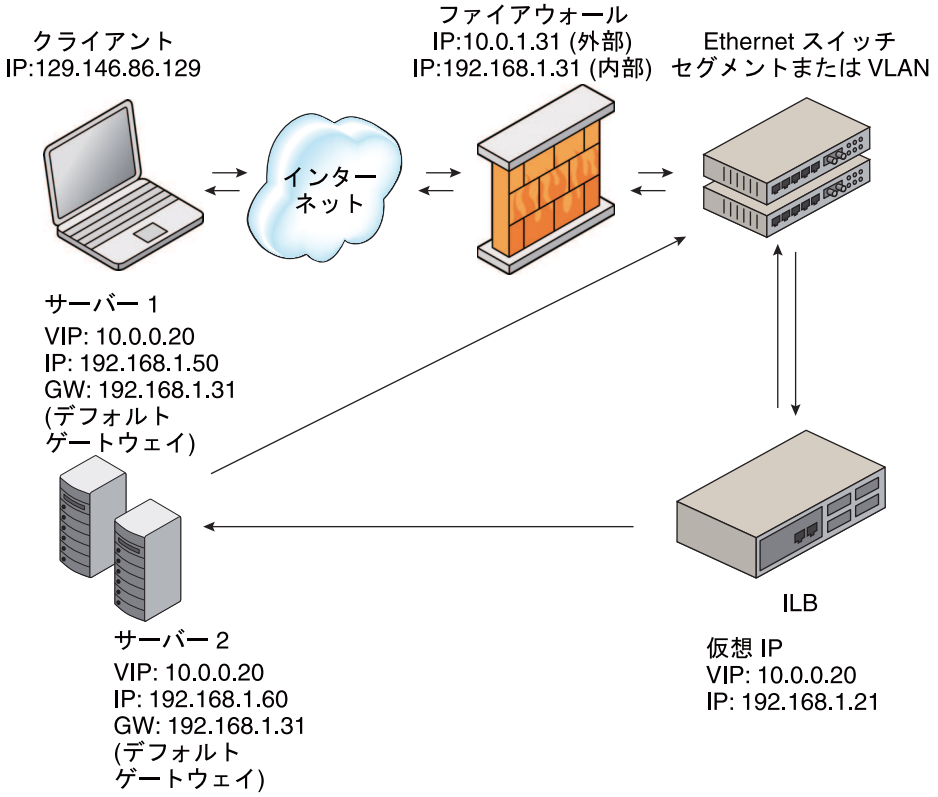
- パケットの着信先 MAC アドレスのみが変更され、サーバーがクライアントに直接応答するため、NAT よりもパフォーマンスが優れています。
- サーバーとクライアントの間に完全な透過性があります。サーバーはクライアント IP アドレスから接続を直接認識し、デフォルトゲートウェイを介してクライアントに応答します。

欠点:

- バックエンドサーバーは、それ固有の IP アドレス (健全性検査用) および仮想 IP アドレス (負荷分散トラフィック用) の両方に応答する必要があります。
- ロードバランサは接続状態を維持しない (つまりステートレスである) ため、サーバーを追加または削除すると接続が中断されます。

次の図に、DSR トポロジを使用した ILB の実装を示します。

図 10–1　Direct Server Return トポロジ



この図で、バックエンドサーバーはどちらも ILB ボックスと同じサブネット (192.168.1.0/24) 内にあります。また、サーバーはルータに接続されているため、ILB ボックスから転送されたリクエストを受け取ったあと、クライアントに直接応答することができます。

ネットワークアドレス変換トポロジ

ILB は負荷分散機能のためだけに、NAT をスタンドアロンモードで使用します。このモードでは、ILB はヘッダー情報を書き換え、受信トラフィックと送信トラフィックを処理します。ILB はハーフ NAT モードとフル NAT モードの両方で動作します。ただし、フル NAT は発信元 IP アドレスも書き換えるため、サーバーには、すべての接続がロードバランサから発信されているように見えます。NAT は TCP 接続追跡を提供します (つまりステートフルです)。NAT モードは、追加のセキュリティーを提供し、ハイパーテキスト転送プロトコル (HTTP) (または Secure Sockets Layer (SSL)) トラフィックに最適です。

利点:

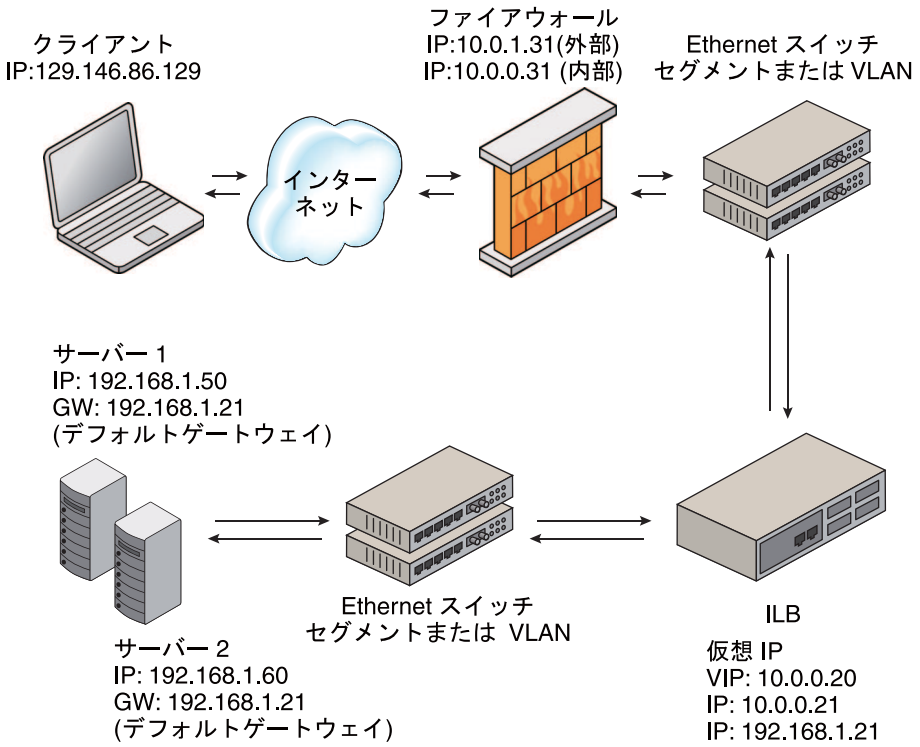
- ロードバランサを指定するようにデフォルトゲートウェイを変更することによって、すべてのバックエンドサーバーで動作します。
- ロードバランサが接続状態を維持するため、接続を中断せずにサーバーを追加または削除できます。

欠点:

- 処理に IP ヘッダーの操作を伴い、サーバーがロードバランサに応答を送信するため、DSR よりもパフォーマンスが低くなります。
- すべてのバックエンドサーバーがロードバランサをデフォルトゲートウェイとして使用する必要があります。

次の図に、NAT トポロジの一般的な実装を示します。

図 10–2 ネットワークアドレス変換トポロジ



この場合、VIP へのリクエストはすべて ILB ボックスを経由し、バックエンドサーバーに転送されます。バックエンドサーバーからの応答はすべて、NAT のために ILB を経由します。

注意 – ILB に実装されている NAT コードパスは、Oracle Solaris の IP フィルタ機能に実装されているコードパスとは異なります。これらのコードパスを同時に使用しないでください。

ハーフ NAT 負荷分散トポロジ

ILB のハーフ NAT 動作モードでは、ILB はパケットのヘッダー内の着信先 IP アドレスのみを書き換えます。ハーフ NAT 実装を使用している場合、サーバーが存在する同一のサブネットから、サービスの仮想 IP (VIP) アドレスに接続することはできません。次の表に、クライアントと ILB の間、および ILB とバックエンドサーバーの間を流れるパケットの IP アドレスを示します。

表 10–1 サーバーとクライアントが異なるネットワーク上にある場合のハーフ NAT 実装のリクエストフローと応答フロー

	リクエストフロー	発信元 IP アドレス	着信先 IP アドレス
1.	クライアント –> ILB	クライアント	ILB の VIP
2.	ILB –> サーバー	クライアント	サーバー
	応答フロー		
3.	サーバー –> ILB	サーバー	クライアント
4.	ILB –> クライアント	ILB の VIP	クライアント

クライアントシステムをサーバーと同じネットワークに接続した場合、意図したサーバーはクライアントに直接応答します。4 番目の手順は発生しません。したがって、クライアントに対するサーバー応答の発信元 IP アドレスは無効です。クライアントが接続リクエストをロードバランサに送信すると、応答は意図したサーバーから発生します。したがって、クライアントの IP スタックはすべての応答を適切に破棄します。

この場合、リクエストフローと応答フローは次の表に示すとおりに進行します。

表 10–2 サーバーとクライアントが同じネットワーク上にある場合のハーフ NAT 実装のリクエストフローと応答フロー

	リクエストフロー	発信元 IP アドレス	着信先 IP アドレス
1.	クライアント –> ILB	クライアント	ILB の VIP
2.	ILB –> サーバー	クライアント	サーバー
	応答フロー		
3.	サーバー –> クライアント	サーバー	クライアント

フル NAT 負荷分散トポロジ

フル NAT 実装では、発信元 IP アドレスと着信先 IP アドレスが書き換えられることで、トラフィックがロードバランサを両方向で通過します。フル NAT トポロジでは、サーバーが存在する同一のサブネットから VIP に接続できるようになります。

次の表に、フル NAT トポロジの使用時にクライアントと ILB の間、および ILB とバックエンドサーバーの間を流れるパケットの IP アドレスを示します。サーバーには、ILB ボックスを使用する特別なデフォルトルートは不要です。ただし、フル NAT トポロジでは、ILB がバックエンドサーバーとの通信に発信元アドレスとして使用する 1 つの IP アドレスまたは IP アドレス範囲を、管理者が別途設定する必要があります。使用するアドレスがサブネット C に属しているとします。このシナリオでは、ILB はプロキシとして動作します。

表 10–3 フル NAT 実装のリクエストフローと応答フロー

	リクエストフロー	発信元 IP アドレス	着信先 IP アドレス
1.	クライアント –> ILB	クライアント	ILB の VIP
2.	ILB –> サーバー	ロードバランサのインタフェースアドレス (サブネット C)	サーバー
	応答フロー		
3.	サーバー –> ILB	サーバー	ILB のインタフェースアドレス (サブネット C)
4.	ILB –> クライアント	ILB の VIP	クライアント

ILB の動作

このセクションでは、ILB の動作、ILB がクライアントから VIP へのリクエストをどのように処理し、リクエストをバックエンドサーバーにどのように転送し、応答をどのように処理するかについて説明します。

クライアントからサーバーへのパケット処理:

1. ILB は、クライアントから VIP アドレスに送信された受信リクエストを受け取り、リクエストを負荷分散規則と照合します。
2. ILB は一致する負荷分散規則を見つけると、負荷分散アルゴリズムを使用して、動作モードに応じてリクエストをバックエンドサーバーに転送します。
   - DSR モードでは、ILB は受信リクエストの MAC ヘッダーを、選択されたバックエンドサーバーの MAC ヘッダーに置換します。

- ハーフ NAT モードでは、ILB は受信リクエストの着信先 IP アドレスおよびトランスポートプロトコルのポート番号を、選択されたバックエンドサーバーのものに置換します。
- フル NAT モードでは、ILB は受信リクエストの発信元 IP アドレスとトランスポートプロトコルのポート番号を、負荷分散規則の NAT 発信元アドレスに置換します。また、ILB は受信リクエストの着信先 IP アドレスおよびトランスポートプロトコルのポート番号を、選択されたバックエンドサーバーのものに置換します。

3. ILB は変更された受信リクエストを、選択されたバックエンドサーバーに転送します。

サーバーからクライアントへのパケット処理:

1. バックエンドサーバーはクライアントからの受信リクエストに応答して、返信を ILB に送信します。
2. バックエンドサーバーから応答を受け取ったあとの ILB のアクションは、動作モードに基づきます。
   - DSR モードでは、バックエンドサーバーからの応答は ILB をバイパスし、クライアントに直接届きます。ただし、ILB がバックエンドサーバーのルーターとしても使用される場合、バックエンドサーバーからクライアントへの応答は、ILB を実行しているシステムを通るようにルーティングされます。
   - ハーフ NAT モードとフル NAT モードでは、ILB はバックエンドサーバーからの応答を受信リクエストと照合し、変更された IP アドレスおよびトランスポートプロトコルのポート番号を元の受信リクエストのものに置換します。その後、ILB はクライアントに応答を転送します。

ILB のアルゴリズム

ILB のアルゴリズムはトラフィック分散を制御し、負荷分散およびサーバー選択のためのさまざまな特性を提供します。ILB は 2 つの動作モードに対して次のアルゴリズムを提供します。

- ラウンドロビン – ラウンドロビンアルゴリズムでは、ロードバランサはサーバーグループに対してローテーションベースでリクエストを割り当てます。サーバーにリクエストが割り当てられると、そのサーバーはリストの末尾に移動します。
- *src IP* ハッシュ – 発信元 IP ハッシュ方式では、ロードバランサは受信リクエストの発信元 IP アドレスのハッシュ値に基づいてサーバーを選択します。
- *src-IP, port* ハッシュ – 発信元 IP、ポートハッシュ方式では、ロードバランサは受信リクエストの発信元 IP アドレスおよび発信元ポートのハッシュ値に基づいてサーバーを選択します。

- *src-IP, VIP* ハッシュ – 発信元 IP、VIP ハッシュ方式では、ロードバランサは受信リクエストの発信元 IP アドレスおよび着信先 IP アドレスのハッシュ値に基づいてサーバーを選択します。

サービス管理機能

ILB はサービス管理機能 (SMF) サービス `svc:/network/loadbalancer/ilb:default` によって管理されています。SMF の概要については、『Oracle Solaris 11.1 でのサービスと障害の管理』の第 1 章「サービスの管理 (概要)」を参照してください。SMF に関連する詳細な手順については、『Oracle Solaris 11.1 でのサービスと障害の管理』の第 2 章「サービスの管理 (タスク)」を参照してください。

ILB のコマンド行インタフェース

ILB のコマンド行インタフェースは `/usr/sbin/ilbadm` ディレクトリにあります。CLI には、負荷分散規則、サーバーグループ、および健全性検査を構成するサブコマンドが含まれています。また、統計情報や構成の詳細を表示するサブコマンドも含まれています。サブコマンドは、次の 2 つのカテゴリに分けられます。

- 構成サブコマンド – これらのサブコマンドでは次のタスクを実行できます。
  - 負荷分散規則の作成および削除
  - 負荷分散規則の有効化および無効化
  - サーバーグループの作成および削除
  - サーバーグループへのサーバーの追加および削除
  - バックエンドサーバーの有効化および無効化
  - 負荷分散規則内のサーバーグループに関するサーバー健全性検査の作成および削除

注 – 構成サブコマンドを管理するには、特権が必要です。特権は、Oracle Solaris の役割に基づくアクセス制御 (RBAC) を通じて取得します。適切な役割の作成およびユーザーへの役割の割り当てについては、『Oracle Solaris 11.1 の管理: セキュリティーサービス』の「RBAC の初期構成 (タスクマップ)」を参照してください。

- 表示サブコマンド – これらのサブコマンドでは次のタスクを実行できます。
  - 構成された負荷分散規則、サーバーグループ、および健全性検査の表示
  - パケット転送統計の表示
  - NAT 接続テーブルの表示
  - 健全性検査結果の表示
  - セッション持続性マッピングテーブルの表示

注 – 表示サブコマンドを管理するために特権は不要です。

ilbadm サブコマンドの一覧については、171 ページの「ILB のコマンドおよびサブコマンド」を参照してください。ilbadm サブコマンドの詳細については、ilbadm(1M) のマニュアルページを参照してください。

ILB のコマンドおよびサブコマンド

ilbadm およびそのサブコマンドを使用して、負荷分散規則を操作できます。ilbadm サブコマンドの詳細については、ilbadm(1M) のマニュアルページを参照してください。

表 10–4 負荷分散規則の操作に使用される ILB のサブコマンド

ILB のサブコマンド	説明
ilbadm create-rule	特定の特性を持つ rule name を作成します。
ilbadm show-rule	指定された規則の特性を表示するか、規則が指定されていない場合はすべての規則を表示します。
ilbadm delete-rule	rule name に関連するすべての情報を削除します。rule name が存在しない場合、このサブコマンドは失敗します。
ilbadm enable-rule	指定された規則を有効にするか、名前が指定されていない場合はすべての規則を有効にします。
ilbadm disable-rule	指定された規則を無効にするか、名前が指定されていない場合はすべての規則を無効にします。
ilbadm show-statistics	ILB の統計情報を表示します。たとえば、このサブコマンドに -t を付けると、各ヘッダーにタイムスタンプが含められます。
ilbadm show-hc-result	指定された規則の名前 rule-name に関連付けられたサーバーの健全性検査結果を表示します。rule-name が指定されていない場合、すべての規則についてサーバーの健全性検査結果を表示します。
ilbadm show-nat	NAT テーブル情報を表示します。
ilbadm create-servergroup	1 つ以上のサーバーを含むサーバーグループを作成します。追加のサーバーは、ilbadm add-server を使用して追加できます。
ilbadm delete-servergroup	サーバーグループを削除します。
ilbadm show-servergroup	1 つのサーバーグループを一覧表示するか、サーバーグループが指定されていない場合はすべてのサーバーグループを一覧表示します。

表 10–4 負荷分散規則の操作に使用される ILB のサブコマンド (続き)

ILB のサブコマンド	説明
`ilbadm enable-server`	無効にされたサーバーを有効にします。
`ilbadm disable-server`	指定されたサーバーを無効にします。
`ilbadm add-server`	指定されたサーバーをサーバーグループに追加します。
`ilbadm show-server`	指定された規則に関連付けられたサーバーを表示するか、規則名が指定されていない場合はすべてのサーバーを表示します。
`ilbadm remove-server`	サーバーグループから 1 つ以上のサーバーを削除します。
`ilbadm create-healthcheck`	規則の設定に使用できる健全性検査情報を設定します。
`ilbadm show-healthcheck`	構成されている健全性検査の詳細情報を表示します。
`ilbadm delete-healthcheck`	健全性検査情報を削除します。
`ilbadm show-persist`	セッション持続性マッピングテーブルを表示します。
`ilbadm export-config` *filename*	`ilbadm import` を使用してインポートに適した形式で既存の ILB 構成ファイルをエクスポートします。*filename* が指定されていない場合、`ilbadm export` は `stdout` に書き込みます。
`ilbadm import-config -p` *filename*	ファイルをインポートし、このインポートされたファイルの内容で既存の ILB 構成を置換します。*filename* が指定されていない場合、`ilbadm import` は `stdin` から読み取ります。

◆ ◆ ◆ 第 11 章

11 統合ロードバランサの構成

この章では、統合ロードバランサ (ILB) のインストールについて説明し、簡単な ILB 構成の設定例を示します。次の内容について説明します。

ILB のインストール

ILB には、カーネルとユーザーランドの 2 つの部分があります。カーネル部分は、Oracle Solaris 11 インストールの一部として自動的にインストールされます。ILB のユーザーランド部分を入手するには、`pkg install ilb` コマンドを使用して `ilb` パッケージを手動でインストールする必要があります。

ILB の有効化

このセクションでは、ILB の有効化に使用する手順について説明します。

▼ ILB を有効にする方法

始める前に　システムの役割に基づくアクセス制御 (RBAC) の属性ファイルに、次のエントリがあることを確認します。エントリが存在しない場合は、手動で追加します。

- ファイル名: `/etc/security/auth_attr`
  - `solaris.network.ilb.config:::Network ILB Configuration::help=NetworkILBconf.html`

    - `solaris.network.ilb.enable:::Network ILB Enable Configuration::help=NetworkILBenable.html`
    - `solaris.smf.manage.ilb:::Manage Integrated Load Balancer Service States::help=SmfILBStates.html`
  - ファイル名: `/etc/security/prof_attr`
    - `Network ILB:::Manage ILB configuration via ilbadm:auths=solaris.network.ilb.config,solaris.network.ilb.enable; help=RtNetILB.html`
    - ファイル内の Network Management エントリに `solaris.smf.manage.ilb` が含まれている必要があります。
  - ファイル名: `/etc/user_attr`
    - `daemon::::auths=solaris.smf.manage.ilb,solaris.smf.modify.application`

ILB 構成サブコマンドのユーザー承認を設定する必要があります。171 ページの「ILB のコマンドおよびサブコマンド」に示されている ILB 構成サブコマンドを実行するには、`solaris.network.ilb.config` RBAC 承認が必要です。

- 既存のユーザーに承認を割り当てるには、『Oracle Solaris 11.1 の管理: セキュリティーサービス』の第 9 章「役割に基づくアクセス制御の使用 (タスク)」を参照してください。
- システムに新しいユーザーアカウントを作成するときに承認を与えることもできます。

  次の例では、ユーザー `ilbadm` をグループ ID `10`、ユーザー ID `1210` で作成し、システムの ILB を管理する承認を与えます。

  ```
  # useradd -g 10 -u 1210 -A solaris.network.ilb.config ilbadmin
  ```

  `useradd` コマンドは、`/etc/passwd`、`/etc/shadow`、および `/etc/user_attr` ファイルに新しいユーザーを追加します。`-A` オプションは、ユーザーに承認を割り当てます。

1. **ILB Management 権利プロファイルを含んでいる役割になるか、スーパーユーザーになります。**

   作成する役割に ILB Management 権利プロファイルを割り当てることができます。役割の作成およびユーザーへの割り当てについては、『Oracle Solaris 11.1 の管理: セキュリティーサービス』の「RBAC の初期構成 (タスクマップ)」を参照してください。

2. **適切な転送サービス (IPv4 と IPv6 のいずれか、またはその両方) を有効にします。**

   このコマンドは正常に終了した場合は出力を生成しません。

   ```
   # ipadm set-prop -p forwarding=on ipv4
   # ipadm set-prop -p forwarding=on ipv6
   ```

3 **ILB サービスを有効にします。**

```
# svcadm enable ilb
```

4 **ILB サービスが有効になっていることを確認します。**

```
# svcs ilb
```

ILB の構成

このセクションでは、ハーフ NAT トポロジを使用して 2 つのサーバー間でトラフィックの負荷分散を行うように ILB を設定する手順について説明します。163 ページの「ILB の動作モード」の NAT トポロジの実装を参照してください。

▼ ILB を構成する方法

1 **ILB Management 権利プロファイルを含んでいる役割になるか、スーパーユーザーになります。**

作成する役割に ILB Management 権利プロファイルを割り当てることができます。役割の作成およびユーザーへの割り当てについては、『Oracle Solaris 11.1 の管理: セキュリティーサービス』の「RBAC の初期構成 (タスクマップ)」を参照してください。

2 **バックエンドサーバーを設定します。**

このシナリオでは、バックエンドサーバーは ILB をデフォルトルーターとして使用するように設定されます。これを行うには、次に示すコマンドを両方のサーバーで実行します。

```
# route add -p default 192.168.1.21
```

このコマンドを実行したあと、両方のサーバーでサーバーアプリケーションを起動します。ポート 5000 で待機している TCP アプリケーションであるとします。

3 **ILB でサーバーグループを設定します。**

2 つのサーバー `192.168.1.50` および `192.169.1.60` があります。これら 2 つのサーバーから成るサーバーグループ `srvgrp1` は、次のコマンドを入力すると作成できます。

```
# ilbadm create-sg -s servers=192.168.1.50,192.168.1.60 srvgrp1
```

4 **`hc-srvgrp1` (次のコマンドを入力して作成できます) という単純な健全性検査を設定します。**

単純な TCP レベルの健全性検査を使用して、サーバーアプリケーションが到達可能かどうかを検出します。この確認は 60 秒ごとに行われます。最大 3 秒の待機時間を

おいて最大 3 回試行し、サーバーが正常かどうかを確認します。3 回の試行がすべて失敗した場合は、サーバーを dead としてマークします。

```
# ilbadm create-hc -h hc-test=tcp,hc-timeout=3, \
hc-count=3,hc-inerval=60  hc-srvgrp1
```

5 次のコマンドを入力して、**ILB** 規則を設定します。

この規則では、持続性 (32 ビットマスク) が使用されます。負荷分散アルゴリズムは round robin です。使用されるサーバーグループは srvgrp1、使用される健全性検査メカニズムは hc-srvgrp1 です。この規則を作成するには、次のコマンドを入力します。

```
# ilbadm create-rule -e -p -i vip=10.0.2.20,port=5000 -m \
 lbalg=rr,type=half-nat,pmask=32 \
-h hc-name=hc-srvgrp1 -o servergroup=srvgrp1  rule1_rr
```

ILB の無効化

次のセクションでは、ILB を無効にする手順について説明します。

▼ ILB を無効にする方法

1 **ILB Management** 権利プロファイルを含んでいる役割になるか、スーパーユーザーになります。

作成する役割に ILB Management 権利プロファイルを割り当てることができます。役割の作成およびユーザーへの割り当てについては、『Oracle Solaris 11.1 の管理: セキュリティーサービス』の「RBAC の初期構成 (タスクマップ)」を参照してください。

2 **ILB** サービスを無効にします。

```
# svcadm disable ilb
```

3 **ILB** サービスが無効になっていることを確認します。

```
# svcs ilb
```

構成のインポートとエクスポート

ilbadm export サブコマンドは、現在の ILB 構成をユーザー指定のファイルにエクスポートします。この情報は、あとで ilbadm import サブコマンドへの入力として使用できます。

ilbadm import サブコマンドは、既存の構成を保持するよう特に指示されていないかぎり、インポートの前に既存の構成を削除します。ファイル名が省略されている場合、コマンドは stdin から読み取るか、stdout に書き込みます。

ILB 構成をエクスポートするには、export-config コマンドを使用します。次の例では、import サブコマンドによるインポートに適した形式で、現在の構成をファイル /var/tmp/ilb_config にエクスポートします。

```
# ilbadm export-config /var/tmp/ilb_config
```

ILB 構成をインポートするには、import-config コマンドを使用します。次の例では、ファイル /var/tmp/ilb_config の内容を読み取り、既存の構成をオーバーライドします。

```
# ilbadm import-config /var/tmp/ilb_config
```

ILB の高可用性構成 (アクティブパッシブモードのみ)

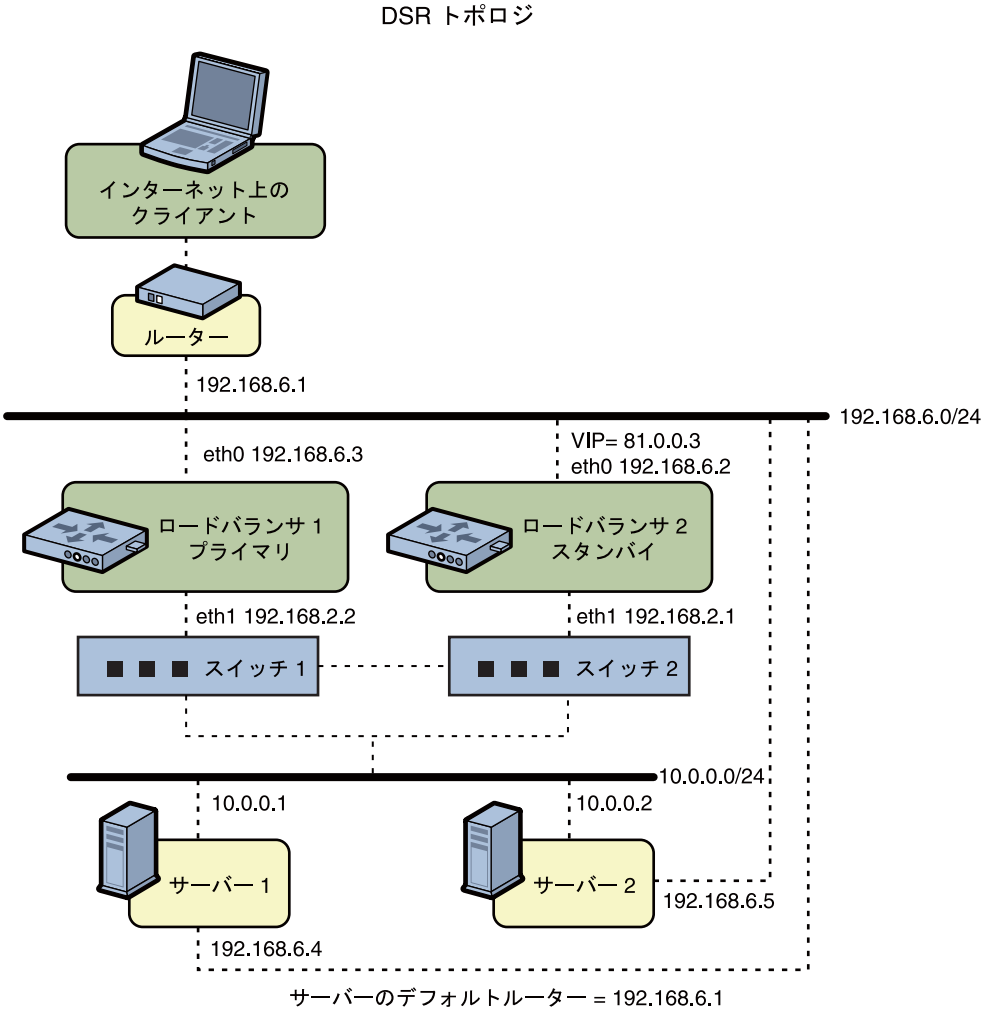
このセクションでは、DSR、ハーフ NAT トポロジを使用した ILB の高可用性 (HA) 構成について説明します。

DSR トポロジを使用した ILB の高可用性構成

このセクションでは、DSR トポロジを使用して高可用性 (HA) を実現するために ILB 接続を設定する方法について説明します。2 つのロードバランサを設定する必要があり、1 つはプライマリロードバランサ、もう 1 つはスタンバイロードバランサになります。プライマリロードバランサに障害が発生すると、スタンバイロードバランサがプライマリロードバランサの役割を引き受けます。

次の図は、ILB 接続を構成して HA を実現するための DSR トポロジを示しています。

図 11–1　DSR トポロジを使用した ILB の HA 構成



ロードバランサのすべての VIP はサブネット 192.168.6.0/24 に面したインタフェース上に構成されています。

▼ DSR トポロジを使用して高可用性を実現するために ILB を構成する方法

1 **ILB Management** 権利プロファイルを含んでいる役割になるか、スーパーユーザーになります。

作成する役割に ILB Management 権利プロファイルを割り当てることができます。役割の作成およびユーザーへの割り当てについては、『Oracle Solaris 11.1 の管理: セキュリティーサービス』の「RBAC の初期構成 (タスクマップ)」を参照してください。

2 プライマリロードバランサとスタンバイロードバランサの両方を構成します。

```
# ilbadm create-servergroup -s server=10.0.0.1,10.0.0.2 sg1
# ilbadm create-rule -i vip=81.0.0.3,port=9001 \
-m lbalg=hash-ip-port,type=DSR -o servergroup=sg1 rule1
```

3 すべてのサーバーの **lo0** インタフェースに **VIP** が構成されていることを確認します。

```
Server1# ipadm create-addr -d -a 81.0.0.3/24 lo0
Server2# ipadm create-addr -d -a 81.0.0.3/24 lo0
```

4 ロードバランサ **1** がプライマリロードバランサとして機能するように構成します。

```
LB1# dladm create-vnic -m vrrp -V 1 -A inet -l eth0 vnic1
LB1# vrrpadm create-router -V 1 -A inet -l eth0 -p 255 vrrp1
LB1# ipadm create-addr -d -a 81.0.0.3/24 vnic1
```

5 ロードバランサ **2** がスタンバイロードバランサとして動作するように構成します。

```
LB2# dladm create-vnic -m vrrp -V 1 -A inet -l eth0 vnic1
LB2# vrrpadm create-router -V 1 -A inet -l eth0 -p 100 vrrp1
LB2# ipadm create-addr -d -a 81.0.0.3/24 vincl
```

前述の構成は、次の障害シナリオに対する保護を提供します。

- ロードバランサ 1 に障害が発生すると、ロードバランサ 2 がプライマリロードバランサになります。ロードバランサ 2 は VIP 81.0.0.3 のアドレス解決を引き継ぎ、着信先 IP アドレス 81.0.0.3 を持つクライアントからのすべてのパケットを処理します。

  ロードバランサ 1 が回復すると、ロードバランサ 2 はスタンバイモードに戻ります。

- ロードバランサ 1 の 1 つまたは両方のインタフェースに障害が発生すると、ロードバランサ 2 はプライマリロードバランサとして引き継ぎます。ロードバランサ 2 は VIP 81.0.0.3 のアドレス解決を引き継ぎ、着信先 IP アドレス 81.0.0.3 を持つクライアントからのすべてのパケットを処理します。

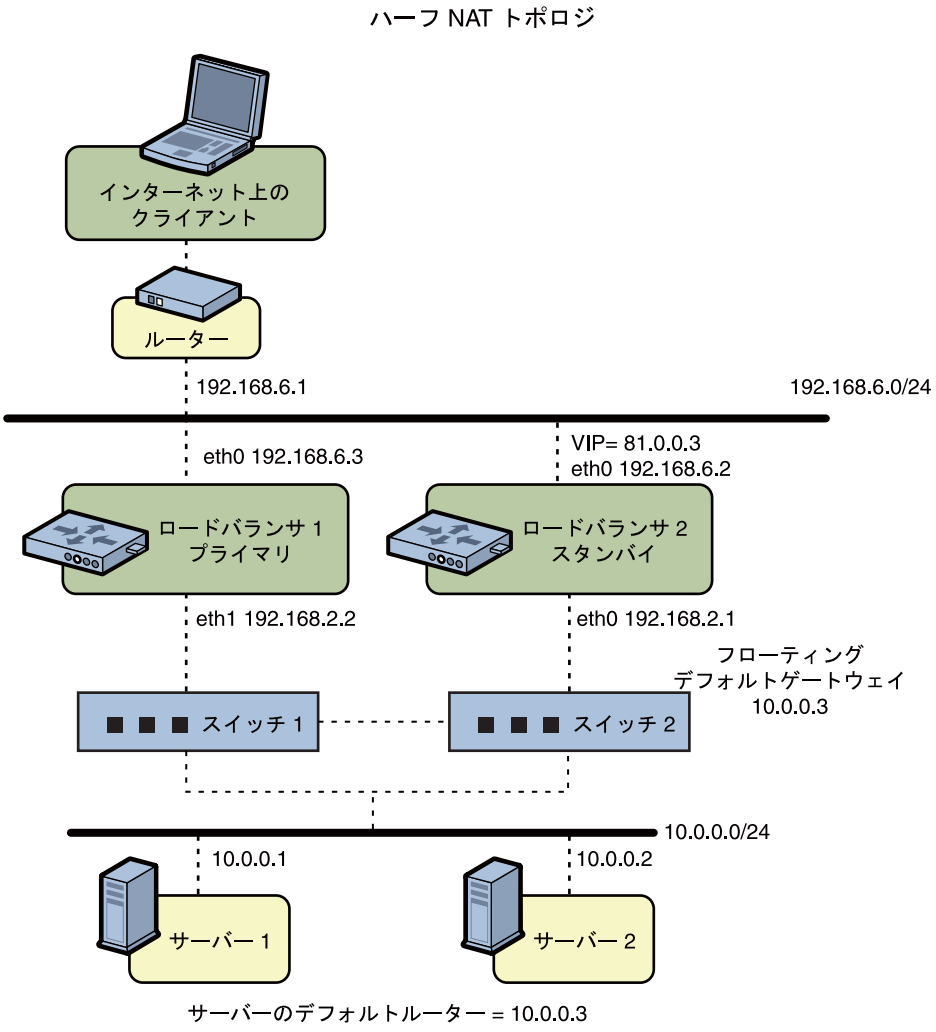
  ロードバランサ 1 の両方のインタフェースが正常になると、ロードバランサ 2 はスタンバイモードに戻ります。

ハーフ NAT トポロジを使用した ILB の高可用性構成

このセクションでは、ハーフ NAT トポロジを使用して高可用性 (HA) を実現するために ILB 接続を設定する方法について説明します。2 つのロードバランサを設定する必要があり、1 つはプライマリ、もう 1 つはスタンバイになります。プライマリロードバランサに障害が発生すると、スタンバイロードバランサがプライマリロードバランサの役割を引き受けます。

次の図は、ILB 接続を構成して HA を実現するためのハーフ NAT トポロジを示しています。

図 11–2　ハーフ NAT トポロジを使用した ILB の HA 構成



ロードバランサのすべての VIP はサブネット 192.168.6.0/24 に面したインタフェース上に構成されています。

▼ ハーフ NAT トポロジを使用して高可用性を実現するために ILB を構成する方法

1 **ILB Management** 権利プロファイルを含んでいる役割になるか、スーパーユーザーになります。

作成する役割に ILB Management 権利プロファイルを割り当てることができます。役割の作成およびユーザーへの割り当てについては、『Oracle Solaris 11.1 の管理: セキュリティーサービス』の「RBAC の初期構成 (タスクマップ)」を参照してください。

2 プライマリロードバランサとスタンバイロードバランサの両方を構成します。

```
# ilbadm create servergroup -s server=10.0.0.1,10.0.0.2 sg1
# ilbadm create-rule -ep -i vip=81.0.0.3,port=9001-9006,protocol=udp \
-m lbalg=roundrobin,type=HALF-NAT,pmask=24 \
-h hc-name=hc1,hc-port=9006 \
-t conn-drain=70,nat-timeout=70,persist-timeout=70 -o servergroup=sg1 rule1
```

3 ロードバランサ 1 がプライマリロードバランサとして機能するように構成します。

```
LB1# dladm create-vnic -m vrrp -V 1 -A inet -l eth0 vnic1
LB1# ipadm create-addr -d -a 81.0.0.3/24 vnic1
LB1# vrrpadm create-router -V 1 -A inet -l eth0 -p 255 vrrp1
LB1# dladm create-vnic -m vrrp -V 2 -A inet -l eth1 vnic2
LB1# ipadm create-addr -d -a 10.0.0.3/24 vnic2
LB1# vrrpadm create-router -V 2 -A inet -l eth1 -p 255 vrrp2
```

4 ロードバランサ 2 がスタンバイロードバランサとして動作するように構成します。

```
LB2# dladm create-vnic -m vrrp -V 1 -A inet -l eth0 vnic1
LB2# ipadm create-addr -d -a 81.0.0.3/24 vnic1
LB2# vrrpadm create-router -V 1 -A inet -l eth0 -p 100 vrrp1
LB2# dladm create-vnic -m vrrp -V 2 -A inet -l eth1 vnic2
LB2# ipadm create-addr -d -a 10.0.0.3/24 vnic2
LB2# vrrpadm create-router -V 2 -A inet -l eth1 -p 100 vrrp2
```

5 両方のサーバーにフローティングデフォルトゲートウェイの IP アドレスを追加します。

```
# route add net 192.168.6.0/24 10.0.0.3
```

前述の構成は、次の障害シナリオに対する保護を提供します。

- ロードバランサ 1 に障害が発生すると、ロードバランサ 2 がプライマリロードバランサになります。ロードバランサ 2 は VIP `81.0.0.3` のアドレス解決を引き継ぎ、着信先 IP アドレス `81.0.0.3` を持つクライアントからのすべてのパケットを処理します。ロードバランサ 2 は、フローティングゲートウェイアドレス `10.0.0.3` に送信されるすべてのパケットも処理します。

  ロードバランサ 1 が回復すると、ロードバランサ 2 はスタンバイモードに戻ります。

- ロードバランサ 1 の 1 つまたは両方のインタフェースに障害が発生すると、ロードバランサ 2 はプライマリロードバランサとして引き継ぎます。ロードバランサ 2 は VIP `81.0.0.3` のアドレス解決を引き継ぎ、着信先 IP アドレス `81.0.0.3` を持つクライアントからのすべてのパケットを処理します。ロードバランサ 2 は、フローティングゲートウェイアドレス `10.0.0.3` に送信されるすべてのパケットも処理します。

  ロードバランサ 1 の両方のインタフェースが正常になると、ロードバランサ 2 はスタンバイモードに戻ります。

注 – ILB の現在の実装では、プライマリロードバランサとスタンバイロードバランサは同期されません。プライマリロードバランサに障害が発生してスタンバイロードバランサが引き継ぐと、既存の接続は失敗します。ただし、同期しない HA でも、プライマリロードバランサに障害が発生した状況では価値があります。

◆ ◆ ◆ 第 12 章

12 統合ロードバランサの管理

この章では、サーバーグループの作成や削除といったILBサーバーグループの管理、サーバーグループのサーバーの追加、削除、再有効化、無効化といったバックエンドサーバーの管理、規則の作成と削除、および統計の表示の手順について説明します。

次の内容について説明します。

ILB サーバーグループの管理

このセクションでは、`ilbadm` コマンドを使用してILBサーバーグループを作成、削除、および一覧表示する方法について説明します。

▼ ILB サーバーグループを作成する方法

1. 作成するサーバーグループの名前を選択します。
2. サーバーグループに含めるサーバーを選択します。
   サーバーは、ホスト名またはIPアドレスと、オプションのポートによって指定できます。
3. サーバーグループを作成します。

```
# ilbadm create-servergroup -s servers= \
server1,server2,server3  servergroup
```

例 12–1 ILB サーバーグループの作成

次の例では、3 つのサーバーで構成される webgroup というサーバーグループを作成します。

```
# ilbadm create-servergroup -s servers=webserv1,webserv2,webserv3 webgroup
```

▼ ILB サーバーグループを削除する方法

1 端末ウィンドウで **show-servergroup** サブコマンドを入力して、特定のサーバーグループまたはすべてのサーバーグループに関する情報を取得します。

```
# ilbadm show-servergroup -o all
```

次のコマンド例では、すべてのサーバーグループに関する詳細な情報が一覧表示されます。

sgname	serverID	minport	maxport	IP_address
specgroup	_specgroup.0	7001	7001	199.199.68.18
specgroup	_specgroup.1	7001	7001	199.199.68.19
test123	_test123.0	7002	7002	199.199.67.18
test123	_test123.1	7002	7002	199.199.67.19

上の表には、2 つのサーバーグループ specgroup および test123 が示されています。specgroup には 2 つのサーバー 199.199.68.18 および 199.199.68.19 があり、サーバーはポート 7001 を使用しています。同様に、test123 にも 2 つのサーバー 199.199.67.18 および 199.199.67.19 があります。このサーバーはポート 7002 を使用しています。

2 削除するサーバーグループを選択します。

サーバーグループは、アクティブな規則によって使用されていてはいけません。それ以外の場合、削除に失敗します。

3 端末ウィンドウで、次のコマンドを使用してサーバーグループを削除します。

```
# ilbadm delete-servergroup servergroup
```

例 12–2 ILB サーバーグループの削除

次の例では、webgroup というサーバーグループを削除します。

```
# ilbadm delete-servergroup webgroup
```

ILB でのバックエンドサーバーの管理

このセクションでは、ilbadm コマンドを使用して、サーバーグループ内の 1 つ以上のバックエンドサーバーを追加、削除、有効化、および無効化する方法について説明します。

▼ ILB サーバーグループにバックエンドサーバーを追加する方法

- サーバーグループにバックエンドサーバーを追加します。

  サーバー指定にはホスト名または IP アドレスを含める必要があり、オプションのポートまたはポート範囲を含めることもできます。同一の IP アドレスを持つサーバーエントリは、1 つのサーバーグループ内で許可されません。

  ```
  # ilbadm add-server -s server=192.168.89.1,192.168.89.2 ftpgroup
  # ilbadm add-server -s server=[2001:7::feed:6]:8080 sgrp
  ```

  -e オプションは、サーバーグループに追加されたサーバーを有効にします。

注 – IPv6 アドレスは、角括弧で囲む必要があります。

例 12–3 ILB サーバーグループへのバックエンドサーバーの追加

次の例では、サーバーグループ ftpgroup および sgrp にバックエンドサーバーを追加し、サーバーを有効にします。

```
# ilbadm add-server -e -s \
server=192.168.89.1,192.168.89.2 ftpgroup
# ilbadm add-server -e -s server=[2001:7::feed:6]:8080 sgrp
```

▼ ILB サーバーグループからバックエンドサーバーを削除する方法

1. サーバーグループからバックエンドサーバーを削除するには、次の手順に従います。

   a. サーバーグループから削除するサーバーのサーバー ID を特定します。

      サーバー ID は、サーバーがサーバーグループに追加されたときにシステムに割り当てられる IP アドレスに対応する一意の名前です。これは show-servergroup -o all サブコマンドの出力から取得できます。

   b. サーバーを削除します。

      ```
      # ilbadm remove-server -s server=serverID servergroup
      ```

2 すべてのサーバーグループからバックエンドサーバーを削除するには、次の手順に従います。

a. 削除するサーバーの **IP** アドレスとホスト名を特定します。

b. **ilbadm show-servergroup-o all** コマンドの出力を使用して、そのサーバーを含むサーバーグループを特定します。

c. 各サーバーグループで、前述のサブコマンドを実行して、サーバーグループからサーバーを削除します。

例 12–4 ILB サーバーグループからのバックエンドサーバーの削除

次の例では、サーバー ID `_sg1.2` を持つサーバーをサーバーグループ `sg1` から削除します。

```
# ilbadm remove-server -s server=_sg1.2 sg1
```

次の事項に注意してください:

- サーバーが NAT またはハーフ NAT 規則によって使用されている場合は、削除の前に `disable-server` サブコマンドを使用してサーバーを無効にしてください。詳細は、188 ページの「ILB サーバーグループのバックエンドサーバーを再有効化または無効化する方法」を参照してください。サーバーが無効になると、サーバーは接続排出状態に入ります。すべての接続が排出されたら、`remove-server` サブコマンドを使用してサーバーを削除できます。`disable-server` コマンドを発行したあと、NAT テーブルを定期的に確認して (`show-nat` コマンドを使用)、問題のサーバーにまだ接続が存在するかどうかを調べてください。すべての接続が排出されたら (サーバーが `show-nat` コマンド出力に表示されない)、`remove-server` コマンドを使用してサーバーを削除できます。
- `conn-drain` タイムアウト値が設定されている場合、接続排出状態はタイムアウト期間が終了した時点で完了します。`conn-drain` タイムアウトのデフォルト値は `0` で、つまり接続が正常にシャットダウンされるまで待機し続けることを意味します。

▼ ILB サーバーグループのバックエンドサーバーを再有効化または無効化する方法

1 再有効化または無効化するバックエンドサーバーの **IP** アドレス、ホスト名、またはサーバー **ID** を特定します。

IP アドレスまたはホスト名を指定すると、それに関連付けられたすべての規則についてサーバーが再有効化または無効化されます。サーバー ID を指定すると、サーバー ID に関連付けられた特定の規則についてサーバーが再有効化または無効化されます。

注 – サーバーは複数のサーバーグループに属する場合、複数のサーバー ID を持つことができます。

2 バックエンドサーバーを再有効化または無効化します。

```
# ilbadm enable-server webservergroup.1
# ilbadm disable-server webservergroup.1
```

例 12–5 ILB サーバーグループのバックエンドサーバーの再有効化または無効化

次の例では、サーバー ID websg.1 を持つサーバーを有効化したあと、無効化します。

```
# ilbadm enable-server websg.1
# ilbadm disable-server websg.1
```

ILB での健全性検査の管理

ILB では、ユーザーが選択できる次の種類のオプションのサーバー健全性検査が提供されています。

- 組み込み ping プローブ
- 組み込み TCP プローブ
- 組み込み UDP プローブ
- 健全性検査として実行できるユーザー指定のカスタムテスト

デフォルトでは、ILB は健全性検査を実行しません。負荷分散規則を作成するとき、サーバーグループごとに健全性検査を指定できます。1 つの負荷分散規則につき 1 つの健全性検査のみを構成できます。仮想サービスが有効であるかぎり、有効化されている仮想サービスに関連付けられたサーバーグループの健全性検査は自動的に開始され、定期的に繰り返されます。仮想サービスが無効化されると健全性検査はすぐに停止します。仮想サービスがふたたび有効化されたとき、以前の健全性検査状態は保持されていません。

健全性検査を実行するために TCP プローブ、UDP プローブ、またはカスタムテストプローブを指定した場合、ILB はデフォルトで、指定された TCP プローブ、UDP プローブ、またはカスタムテストプローブをサーバーに送信する前に、サーバーが到達可能かどうかを判別するために ping プローブを送信します。ping プローブはサーバーの健全性をモニターするための方法です。ping プローブに失敗すると、対応するサーバーは、健全性検査ステータスが unreachable になり無効化されます。ping プローブに成功しても、TCP プローブ、UDP プローブ、またはカスタムテストプローブに失敗した場合は、サーバーの健全性検査ステータスが dead になり無効化されます。

注 –

- デフォルトの ping プローブを無効にすることができます。
- UDP プローブの場合はデフォルトの ping プローブを無効にすることはできません。したがって、UDP 健全性検査では、ping プローブは常にデフォルトのプローブです。

健全性検査の作成

次の例では、2 つの健全性検査オブジェクト `hc1` および `hc-myscript` が作成されます。最初の健全性検査は組み込み TCP プローブを使用します。2 番目の健全性検査はカスタムテスト `/var/tmp/my-script` を使用します。

```
# ilbadm create-healthcheck \
-h hc-timeout=3,hc-count=2,hc-interval=8,hc-test=tcp hc1
# ilbadm create-healthcheck -h hc-timeout=3, \
hc-count=2,hc-interval=8,hc-test=/var/tmp/my-script hc-myscript
```

各引数の説明は次のとおりです。

`hc-timeout` 健全性検査が完了しない場合に失敗したと見なされるまでのタイムアウトを指定します。

`hc-count` `hc-test` 健全性検査の実行を試行する回数を指定します。

`hc-interval` 連続する健全性検査の間隔を指定します。同期を回避するために、実際の間隔は `0.5 * hc-interval` から `1.5 * hc-interval` の間でランダム化されます。

`hc-test` 健全性検査の種類を指定します。

注 – `hc-test` のポート指定は、`create-rule` サブコマンドの `hc-port` キーワードで指定します。詳細は、`ilbadm(1M)` のマニュアルページを参照してください。

ユーザー指定のテストの詳細

ユーザー指定のカスタムテストは次の条件を満たす必要があります。

- テストはバイナリまたはスクリプトにすることができます。
- テストはシステムのいずれの場所にも存在でき、`create-healthcheck` サブコマンドを使用するときは絶対パスを指定する必要があります。

  `create-rule` サブコマンドの健全性検査指定の一部としてテスト (`/var/tmp/my-script` など) を指定すると、`ilbd` デーモンがプロセスをフォークし、次のようにテストを実行します。

  ```
  /var/tmp/my-script $1 $2 $3 $4 $5
  ```

  各引数の説明は次のとおりです。

  `$1` VIP (リテラルの IPv4 または IPv6 アドレス)

  `$2` サーバー IP (リテラルの IPv4 または IPv6 アドレス)

  `$3` プロトコル (文字列としての UDP、TCP)

  `$4` 数値ポート範囲 (`hc-port` に対するユーザー指定の値)

  `$5` 失敗を返す前にテストが待機する最大時間 (秒)。指定された時間を超えてテストが実行されると、停止される可能性があり、テストは失敗したと見なされます。この値はユーザーによって定義され、`hc-timeout` に指定されます。

ユーザー指定のテスト`my-script` は、すべての引数を使用しても使用しなくてもかまいませんが、次のいずれかを返す必要があります:

- マイクロ秒単位の往復時間 (RTT)
- テストが RTT を計算しない場合は 0
- 失敗した場合は -1

デフォルトでは、健全性検査テストは次の特権で実行されます:
`PRIV_PROC_FORK`、`RIV_PROC_EXEC`、`RIV_NET_ICMPACCESS`。

さらに広い特権セットが必要な場合、テストで `setuid` を実装する必要があります。特権の詳細については、`privileges(5)` のマニュアルページを参照してください。

健全性検査の表示

次の `ilbadm list-healthcheck` サブコマンドを使用して、構成済みの健全性検査に関する詳細情報を取得できます。

```
# ilbadm list-healthcheck
```

次の出力例では、構成済みの 2 つの健全性検査が示されています。

NAME	TIMEOUT	COUNT	INTERVAL	DEF_PING	TEST
hc1	3	2	8	Y	tcp
hc2	3	2	8	N	/var/usr-script

健全性検査結果の表示

`ilbadm list-hc-result` サブコマンドを使用して、健全性検査の結果を取得できます。規則または健全性検査を指定しない場合、サブコマンドはすべての健全性検査を一覧表示します。

次の例は、`rule1` という規則に関連付けられた健全性検査の結果を示しています。

```
# ilbadm show-hc-result rule1
```

RULENAME	HCNAME	SERVERID	STATUS	FAIL	LAST	NEXT	RTT
rule1	hc1	_sg1:0	dead	10	11:01:19	11:01:27	941
rule1	hc1	_sg1:1	alive	0	11:01:20	11:01:34	1111

表の LAST 列は、サーバーで健全性検査が実行された時間を示します。NEXT 列は、サーバーで次の健全性検査が実行される時間を示します。

健全性検査の削除

次の例では、*hc1* という健全性検査を削除します。

```
# ilbadm delete-healthcheck hc1
```

ILB 規則の管理

ILB では、仮想サービスは負荷分散規則で表され、次のパラメータで定義されます。

- 仮想 IP アドレス
- トランスポートプロトコル: TCP または UDP
- ポート番号 (またはポート範囲)

- 負荷分散アルゴリズム
- 負荷分散モードの種類 (DSR、フル NAT、またはハーフ NAT)
- 一連のバックエンドサーバーで構成されるサーバーグループ
- サーバーグループ内の各サーバーに対して実行できる、オプションのサーバー健全性検査
- 健全性検査に使用するオプションのポート

> 注 – 健全性検査は、特定のポートか、`ilbd` デーモンがサーバーのポート範囲からランダムに選択する任意のポートを指定できます。

- 仮想サービスを表す規則名

このセクションでは、`ilbadm` コマンドを使用して負荷分散規則を作成、削除、および一覧表示する方法について説明します。

ILB 規則の一覧表示

規則の構成の詳細を一覧表示するには、`ilbadm show-rule` サブコマンドを使用します。規則名を指定しない場合、すべての規則の情報が提供されます。

```
# ilbadm show-rule
```

コマンドの出力例を次に示します。

RULENAME	STATUS	LBALG	TYPE	PROTOCOL	VIP	PORT
rule-http	E	hash-ip-port	HALF-NAT	TCP	10.0.0.1	80
rule-dns	D	hash-ip	DSR	UDP	10.0.0.1	53
rule-abc	D	roundrobin	NAT	TCP	2003::1	1024
rule-xyz	E	hash-ip-vip	NAT	TCP	2003::1	2048-2050

▼ ILB 規則を作成する方法

1 該当するバックエンドサーバーを含むサーバーグループを作成します。

```
# ilbadm create-servergroup -s server=server1:port-range1,server2:port-range2  sg1
```

2 サーバー健全性検査を規則に関連付けるには、健全性検査を作成します。

```
# ilbadm create-healthcheck -h hc-test=protocol, \
hc-timeout=value1,hc-count=value2 \
,hc-interval=value3 hc1
```

3 規則に関連付ける VIP、ポート、およびオプションのプロトコルを特定します。
これらは -i オプションを使用して指定します。

4 使用する処理 (DSR、ハーフ NAT、またはフル NAT) を選択します。
NAT を選択する場合、proxy-src アドレスとして使用する IP アドレス範囲を指定する必要があります。フル NAT トポロジの場合、この範囲は 10 個の IP アドレスに制限されます。

5 使用する負荷分散アルゴリズムを選択します。
手順 4 および手順 5 のパラメータは、-m オプションで指定できます。詳細は、169 ページの「ILB のアルゴリズム」を参照してください。

6 その他のオプションの機能を選択します。
詳細は、ilbadm(1M) のマニュアルページを参照してください。

7 規則名を選択します。

8 規則を作成して有効にします。
各オプションの詳細については、ilbadm(1M) のマニュアルページを参照してください。

```
# ilbadm create-rule -e -i vip=ipaddr,port=port,protocol=protocol \
-m lbalg=lb-algorithm,type=topology-type,proxy-src=ipaddr1-ipaddr2, \
pmask=value4 -h hc-name=hc1 \
-o servergroup=sg1   rule1
```

次の例は、健全性検査を備えたフル NAT 規則を作成する手順を示しています。

例 12–6 健全性検査セッション持続性を持つフル NAT 規則の作成

この例では、hc1 という健全性検査と、sg1 というサーバーグループを作成します。サーバーグループは、それぞれポート範囲を持つ 2 つのサーバーで構成されます。最後のコマンドは、rule1 という規則を作成して有効にし、この規則をサーバーグループおよび健全性検査に関連付けます。この規則はフル NAT 動作モードを実装します。サーバーグループおよび健全性検査の作成は、規則の作成よりも前に行う必要があります。

```
# ilbadm create-healthcheck -h hc-test=tcp,hc-timeout=2, \
hc-count=3,hc-interval=10  hc1
# ilbadm create-servergroup -s server=60.0.0.10:6000-6009,60.0.0.11:7000-7009 sg1
# ilbadm create-rule -e -i vip=81.0.0.10,port=5000-5009, \
protocol=tcp  -m lbalg=rr,type=NAT, \
```

```
proxy-src=60.0.0.101-60.0.0.104,persist=24 \
-h hc-name=hc1 -o servergroup=sg1 rule1
```

ハーフ NAT またはフル NAT 規則を作成する場合は、connection-drain タイムアウト値を指定します。conn-drain タイムアウトのデフォルト値は 0 で、つまり接続が正常にシャットダウンされるまで待機し続けることを意味します。

ILB 規則の削除

規則を削除するには、ilbadm delete-rule サブコマンドを使用します。すべての規則を削除するには、-a オプションを使用します。次の例では、rule1 という規則を削除します。

```
# ilbadm delete-rule rule1
```

ILB 統計の表示

このセクションでは、ilbadm コマンドを使用して情報を取得する方法 (サーバーの統計や規則の統計を出力するなど) について説明します。NAT テーブルの情報およびセッション持続性マッピングテーブルを表示することもできます。

統計情報の取得

負荷分散の詳細を表示するには、ilbadm show-statistics サブコマンドを使用します。次の例は、show-statistics サブコマンドの使用法を示しています。

```
# ilbadm show-statistics
 PKT_P   BYTES_P   PKT_U   BYTES_U   PKT_D   BYTES_D
   9       636       0         0       0         0
```

PKT_P　処理済みパケット

BYTES_P　処理済みバイト

PKT_U　未処理パケット

BYTES_U　未処理バイト

PKT_D　破棄されたパケット

BYTES_D　破棄されたバイト

NAT 接続テーブルの表示

NAT 接続テーブルを表示するには、`ilbadm show-nat` サブコマンドを使用します。このコマンドを連続して実行する際、要素の相対的な位置について想定しないようにしてください。たとえば、`{{ ilbadm show-nat 10}}` を 2 回実行しても、活発に使用されているシステムでは特に、同じ 10 項目が 2 回表示されることは保証されません。カウント値を指定しない場合、NAT 接続テーブル全体が表示されます。

例 12–7 NAT 接続テーブルのエントリ

次の例では、NAT 接続テーブルの 5 個のエントリが示されています。

```
# ilbadm show-nat 5
UDP: 124.106.235.150.53688 > 85.0.0.1.1024 >>> 82.0.0.39.4127 > 82.0.0.56.1024
UDP: 71.159.95.31.61528 > 85.0.0.1.1024 >>> 82.0.0.39.4146 > 82.0.0.55.1024
UDP: 9.213.106.54.19787 > 85.0.0.1.1024 >>> 82.0.0.40.4114 > 82.0.0.55.1024
UDP: 118.148.25.17.26676 > 85.0.0.1.1024 >>> 82.0.0.40.4112 > 82.0.0.56.1024
UDP: 69.219.132.153.56132 > 85.0.0.1.1024 >>> 82.0.0.39.4134 > 82.0.0.55.1024
```

エントリの形式は次のとおりです。

```
T: IP1 > IP2 >>> IP3 > IP4
```

`T` このエントリで使用されるトランスポートプロトコル

`IP1` クライアントの IP アドレスとポート

`IP2` VIP とポート

`IP3` ハーフ NAT モードの場合、クライアントの IP アドレスとポート。

フル NAT モードの場合、クライアントの IP アドレスとポート。

`IP4` バックエンドサーバーの IP アドレスとポート。

セッション持続性マッピングテーブルの表示

セッション持続性マッピングテーブルを表示するには、`ilbadm show-persist` サブコマンドを使用します。

例 12–8 セッション持続性マッピングテーブルのエントリ

次の例では、セッション持続性マッピングテーブルの 5 個のエントリが示されています。

```
# ilbadm show-persist 5
rule2: 124.106.235.150 --> 82.0.0.56
rule3: 71.159.95.31 --> 82.0.0.55
rule3: 9.213.106.54 --> 82.0.0.55
```

例 12–8 セッション持続性マッピングテーブルのエントリ (続き)

```
rule1: 118.148.25.17 --> 82.0.0.56
rule2: 69.219.132.153 --> 82.0.0.55
```

エントリの形式は次のとおりです。

```
R: IP1 --> IP2
```

R この持続性エントリが関連付けられている規則。

IP1 クライアントの IP アドレス。

IP2 バックエンドサーバーの IP アドレス。

◆ ◆ ◆ 第 13 章

13 仮想ルーター冗長プロトコル (概要)

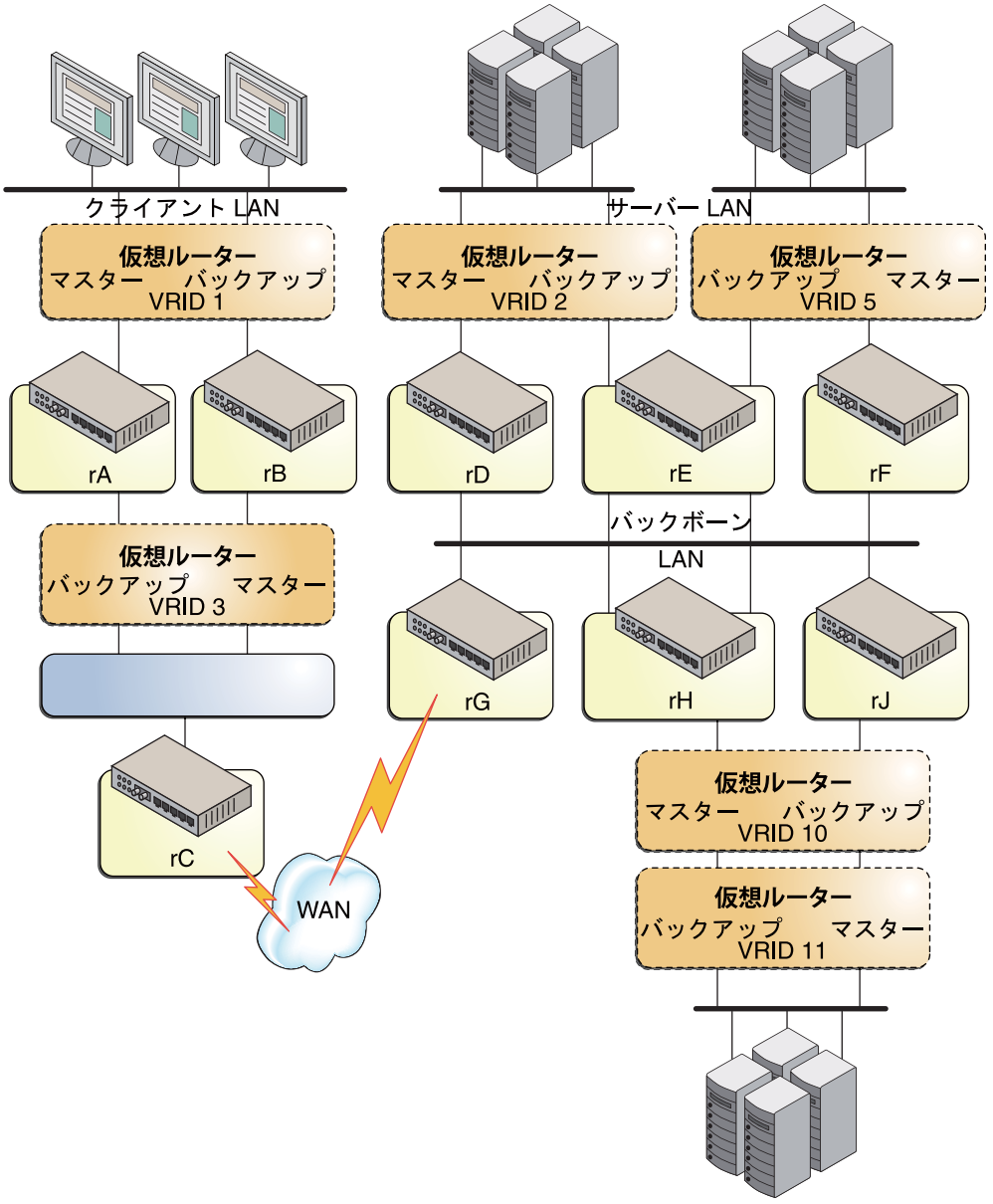
仮想ルーター冗長プロトコル (VRRP) は、Virtual Router Redundancy Protocol Version 3 for IPv4 and IPv6 で規定されたインターネットの標準プロトコルです。VRRP は高い可用性を提供するために Oracle Solaris でサポートされています。Oracle Solaris は、VRRP サービスの構成や管理を行うための管理ツールを提供します。

次の内容について説明します。

VRRP の動作

次の図は、VRRP の動作を示しています。図で使用されている VRRP コンポーネントについては、図に続くテキストで説明します。

図 13–1　VRRP の動作



前の図で、VRRPは次のコンポーネントを使用しています。

- ルーター rA は、仮想ルーター VRID 1 のマスタールーターであり、VRID 3 のバックアップルーターです。ルーター rA は、VRID 1 の VIP にアドレス指定されたパケットのルーティングを処理し、VRID 3 のルーティングの役割を担う準備が整っています。
- ルーター rB は、仮想ルーター VRID 3 のマスタールーターであり、VRID 1 のバックアップルーターです。ルーター rB は、VRID 3 の VIP にアドレス指定されたパケットのルーティングを処理し、VRID 1 のルーティングの役割を担う準備が整っています。
- ルーター rC は、VRRP 機能を持っていませんが、VRID 3 の VIP を使用してクライアント LAN のサブネットに到達します。
- ルーター rD は VRID 2 のマスタールーターです。ルーター rF は VRID 5 のマスタールーターです。ルーター rE は、これらの VRID の両方に対するバックアップルーターです。rD または rF で障害が発生すると、rE がその VRID のマスタールーターになります。rD と rF の両方で同時に障害が発生する可能性があります。VRRP ルーターが、ある VRID のマスタールーターであっても、別の VRID のマスタールーターになることができます。
- ルーター rG は、バックボーン LAN の広域ネットワーク (WAN) ゲートウェイです。バックボーンに接続されているルーターはすべて、OSPF (Open Shortest Path First) などの動的ルーティングプロトコルを使用して WAN 上のルーターとルーティング情報を共有しています。VRRP はこの点に関与しませんが、ルーター rC は、クライアント LAN のサブネットへのパスが VRID 3 の VIP 経由であることを通知します。
- ルーター rH は、VRID 10 のマスタールーターであり、VRID 11 のバックアップルーターです。同様に、ルーター rJ は VRID 11 のマスタールーターであり、VRID 10 のバックアップルーターです。この VRRP 負荷共有構成は、単一のルーターインタフェース上に複数の VRID が存在できることを示しています。

VRRP は、ネットワーク上のすべてのシステムに対してほぼ完全なルーティング冗長性を提供するネットワーク設計の一部として使用できます。

ローカルエリアネットワークでの VRRP の使用

ローカルエリアネットワーク (LAN) などのネットワークを設定するときに、可用性の高いサービスを提供することは非常に重要です。ネットワークの信頼性を高める方法の 1 つは、ネットワーク内の重要なコンポーネントのバックアップを提供することです。ルーター、スイッチ、リンクなどのコンポーネントをネットワークに追加すると、障害時のサービスの継続性が保証されます。ネットワークのエンドポイントで冗長性を提供することは非常に重要なタスクですが、VRRP を使用すれば容易にそれを実現できます。VRRP を使用して LAN に仮想ルーターを導入すれば、ルーターの障害回復を実現できます。

VRRP は、仮想ルーターの役割を LAN 内の VRRP ルーターのいずれかに動的に割り当てるための選出プロトコルです。VRRP は、LAN 上で静的に構成された 1 つのルーターに対し、1 つ以上のバックアップルーターを提供します。

マスタールーターと呼ばれる VRRP ルーターが、仮想ルーターに関連付けられた 1 つまたは複数の IPv4 または IPv6 アドレスを制御します。仮想ルーターは、マスタールーターの IP アドレスに送信されたパケットを転送します。

選出プロセスは、これらの IP アドレスに送信されたパケットの転送中に動的なフェイルオーバーを提供します。VRRP は、静的なデフォルトルーティング環境に固有の単一障害点を取り除きます。

Oracle Solaris の VRRP 機能を使用することで、動的ルーティングまたはルーター発見プロトコルをすべてのエンドホストで構成する必要なく、ルーティング処理のデフォルトパスの可用性を高めることができます。

VRRP ルーター

VRRP は各 VRRP ルーター上で実行され、そのルーターの状態を管理します。1 つのホストで複数の構成された VRRP ルーターを持つことができ、各 VRRP ルーターは異なる仮想ルーターに属することができます。

VRRP ルーターは次の属性を持ちます。

- ルーター名 – システム全体で一意の識別子
- 仮想ルーター **ID (VRID)** – 特定のネットワークセグメント上の仮想ルーターを識別するために使用される一意の番号。VRID は LAN 内の仮想ルーターを識別します
- プライマリ **IP** アドレス – VRRP 通知の発信元 IP アドレス
- 仮想 **IP** アドレス **(VRIP)** – VRID に関連付けられる IP アドレス。ほかのホストはそこからネットワークサービスを取得できます。VRIP は、VRID に属する VRRP インスタンスによって管理されます。
- **VRRP** パラメータ – 優先順位、通知間隔、横取りモード、および受け入れモードを含みます
- **VRRP** の状態情報と統計

VRRP のサブコマンドの管理

次のセクションでは、vrrpadm の概要を示します。詳細は、vrrpadm(1M) のマニュアルページを参照してください。vrrpadm show-router サブコマンドを除き、すべてのサブコマンドの結果に持続性があります。たとえば、vrrpadm create-router で作成された VRRP ルーターはリブート後も持続します。

VRRP VNIC の作成

ネットワークインタフェースカード (NIC) とも呼ばれる、システムの物理ネットワークアダプタ上で構成される擬似的なネットワークインタフェース。1 つの物理インタフェースが複数の VNIC を持つことができます。VNIC は、ネットワーク仮想化の不可欠なコンポーネントです。詳細は、『Oracle Solaris 11.1 での仮想ネットワークの使用』を参照してください。

既存の dladm create-vnic サブコマンドが拡張され、VRRP VNIC を作成できるようになりました。構文は次のとおりです。

```
# dladm create-vnic [-t] [-R root-dir] [-l link] [-m vrrp -V VRID -A \
{inet | inet6}] [-v vlan-id] [-p prop=value[,...]] vnic-link
```

新しい VNIC アドレスタイプ vrrp が導入されました。この新しい VNIC アドレスタイプでは、VRID とアドレスファミリを指定する必要があります。

結果として、既知の仮想ルーターの MAC アドレスを持つ VNIC が作成されます。

ルーターの作成

vrrpadm create-router サブコマンドは、指定された VRID とアドレスファミリ、およびその他の指定されたパラメータを持つ VRRP ルーターを作成します。VRRP ルーターごとに特殊な VRRP VNIC を作成する必要があり、VNIC は dladm create-vnic コマンドを使用して作成できます。詳細は、vrrpadm(1M) のマニュアルページを参照してください。構文は次のとおりです。

```
# vrrpadm create-router -V vrid -l link -A {inet | inet6} \
[-p priority] [-i adv-interval] [-o flags] router-name
```

-o オプションは、VRRP ルーターの横取りモードと受け入れモードを構成するために使用されます。値は preempt、 un_preempt、accept、no_accept のいずれかになります。デフォルトでは、どちらのモードも true に設定されます。

router-name はこの VRRP ルーターの一意の識別子として使用され、ほかの vrrpadm サブコマンドで使用されます。ルーター名で使用できる文字は、英数字 (a-z、A-Z、0-9) と下線 (_) です。ルーター名の最大長は 31 文字です。

ルーターの有効化

無効化された VRRP ルーターは、`enable-router` サブコマンドを使用することでふたたび有効化できます。VRRP ルーターを有効化するときに、そのルーターの作成先となったベースとなるデータリンク (`vrrpadm create-router` でルーターを作成するときに `-l` オプションに指定) とルーターの VRRP VNIC が存在している必要があります。それ以外の場合、有効化の操作は失敗します。構文は次のとおりです。

```
# vrrpadm enable-router router-name
```

ルーターの変更

`vrrpadm modify-router` サブコマンドは、指定された VRRP ルーターの構成を変更します。構文は次のとおりです。

```
# vrrpadm modify-router [-p priority] [-i adv-interval] \
[-o flags] router-name
```

ルーターの構成の表示

`vrrpadm show-router` サブコマンドは、指定された VRRP ルーターの構成やステータスを表示します。詳細は、vrrpadm(1M) のマニュアルページを参照してください。構文は次のとおりです。

```
# vrrpadm show-router [-P | -x] [-p] [-o field[,...]] [router-name]
```

次に、`vrrpadm show-router` の出力例を示します。

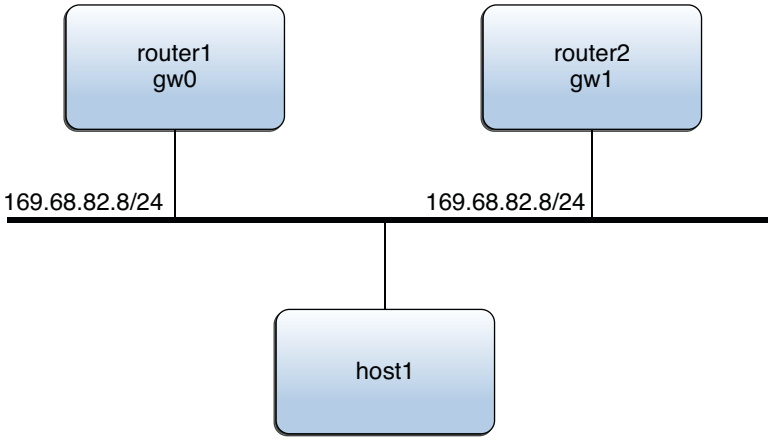
```
# vrrpadm show-router vrrp1
NAME VRID LINK AF PRIO ADV_INTV MODE STATE VNIC
vrrp1 1 bge1 IPv4 100 1000 e-pa- BACK vnic1

# vrrpadm show-router -x vrrp1
NAME STATE PRV_STAT STAT_LAST VNIC PRIMARY_IP VIRTUAL_IPS
vrrp1 BACK MAST 1m17s vnic1 10.0.0.100 10.0.0.1

# vrrpadm show-router -P vrrp1
NAME PEER P_PRIO P_INTV P_ADV_LAST M_DOWN_INTV
vrrp1 10.0.0.123 120 1000 0.313s 3609
```

例 13–1　VRRP の構成例

次の図は、VRRP の典型的な構成を示したものです。



例 13–1　VRRP の構成例　　*(続き)*

この例では、IP アドレス 169.68.82.8 が host1 のデフォルトゲートウェイとして構成されます。この IP アドレスは、2 つの VRRP ルーター router1 と router2 から成る仮想ルーターによって保護される仮想 IP アドレスです。ある瞬間に、2 つのルーターのうちの一方だけがマスタールーターとして機能し、仮想ルーターの役割になり、host1 から届いたパケットを転送します。

仮想ルーターの VRID が 12 であると想定します。次の例は、router1 と router2 で前述の VRRP 構成を構成するために使用されるコマンドを示しています。router1 が仮想 IP アドレス 169.68.82.8 の所有者であり、その優先順位はデフォルト値 (255) です。router2 はスタンバイルーターであり、その優先順位は 100 です。

```
router1:
# dladm create-vnic -m vrrp -V 12 -A inet -l gw0 vnic1
# vrrpadm create-router -V 12 -A inet -l gw0 vrrp1
# ipadm create-addr -d -a 169.68.82.8/24 vnic1/router1
# ipadm create-addr -d -a 169.68.82.100/24 gw0/router1
# vrrpadm show-router -x vrrp1
NAME STATE PRV_STAT STAT_LAST VNIC PRIMARY_IP VIRTUAL_IPS
vrrp1 MAST BACK 1m17s vnic1 169.68.82.100 169.68.82.8
router2:
# dladm create-vnic -m vrrp -V 12 -A inet -l gw1 vnic1
# vrrpadm create-router -V 12 -A inet -l gw1 -p 100 vrrp1
# ipadm create-addr -d -a 169.68.82.8/24 vnic1/router2
# ipadm create-addr -d -a 169.68.82.101/24 gw1/router2
# vrrpadm show-router -x vrrp1
NAME STATE PRV_STAT STAT_LAST VNIC PRIMARY_IP VIRTUAL_IPS
vrrp1 BACK INIT 2m32s vnic1 169.68.82.101 169.68.82.8
```

router1 の構成を例として使用すると、gw0 上で IP アドレスを少なくとも 1 つ構成する必要があります。次の例では、router1 のこの IP アドレスは、VRRP 通知パケットの送信に使用されるプライマリ IP アドレスです。

例 13–1　VRRP の構成例　　*(続き)*

```
# vrrpadm show-router -x vrrp1
NAME STATE PRV_STAT STAT_LAST VNIC PRIMARY_IP VIRTUAL_IPS
vrrp1 MAST BACK 1m17s vnic1 169.68.82.100 169.68.82.8
```

ルーターの無効化

VRRP ルーターは、有効化されるまで機能しません。VRRP ルーターはデフォルトで、最初の作成時に有効化されます。ただし、構成を変更し、その後ルーターをふたたび有効化できるように、VRRP ルーターを一時的に無効化すると便利なときもあります。構文は次のとおりです。

```
# vrrpadm disable-router router-name
```

ルーターの削除

`vrrpadm delete-router` サブコマンドは、指定された VRRP ルーターを削除します。構文は次のとおりです。

```
# vrrpadm delete-router router-name
```

VRRP におけるセキュリティー上の考慮事項

VRRP サービスを構成するには `solaris.network.vrrp` 承認が必要です。読み取り専用操作の `vrrpadm showrouter` ではこの承認は不要です。

`solaris.network.vrrp` 承認は Network Management プロファイルに含まれています。

VRRP の制限事項

このセクションでは、VRRP の制限事項について説明します。

排他的 IP ゾーンのサポート

各排他的 IP ゾーンでは、そのゾーン内で VRRP ルーターが 1 つでも作成されると、VRRP サービス `svc:/network/vrrp/default` が自動的に有効になります。その VRRP サービスが、その特定のゾーンの VRRP ルーターを管理します。

ただし、次の理由のため、排他的 IP ゾーンのサポートは限定されます。

- VNIC は、非大域ゾーン内で作成できません。したがって、まず VRRP VNIC を大域ゾーン内に作成する必要があります。その後、VRRP ルーターが存在する非大域ゾーンに VNIC を割り当てる必要があります。次に `vrrpadm` コマンドを使用して、非大域ゾーンで VRRP ルーターを作成して起動できます。
- 単一の Oracle Solaris システム上で 2 つの VRRP ルーターを異なるゾーン内に作成し、同じ仮想ルーターに参加させることはできません。その理由は、Oracle Solaris では、同じ MAC アドレスを持つ 2 つの VNIC を作成できないためです。

ほかネットワーク機能との相互運用

VRRP サービスは、IP ネットワークマルチパス (IPMP) インタフェース上で動作できません。VRRP は特定の VRRP MAC アドレスを必要としますが、IPMP は完全に IP 層で動作します。

さらに、VRRP の仮想 IP アドレスについては、静的な構成のみが可能であり、既存の 2 つの IP アドレス自動構成ツール (IPv6 自動構成用の `in.ndpd` と DHCP 構成用の `dhcpagent`) を使用して自動構成することはできません。マスターおよびバックアップ VRRP ルーター (VNIC) は同じ MAC アドレスを共有するため、`in.ndpd` と `dhcpagent` が混同される場合があります。最終的に予期しない結果が発生する可能性があります。したがって、IPv6 自動構成と DHCP 構成は VRRP VNIC 上ではサポートされません。VRRP VNIC 上で IPv6 自動構成または DHCP のいずれかを構成する場合、自動構成された IP アドレスを有効にしようとすると失敗し、自動構成処理も失敗します。

◆ ◆ ◆ 付 録 A

リンクアグリゲーションの種類: 機能比較

トランクアグリゲーションとデータリンクマルチパス (DLMP) アグリゲーションはどちらも、ネットワークパフォーマンス向上のためのほぼ同じ機能をサポートしています。ただし、相違点はあります。次の表は、Oracle Solaris におけるこの 2 種類のリンクアグリゲーションの全般的な比較を示します。

機能	トランクアグリゲーション	DLMP アグリゲーション
リンクベースの障害検出	サポートされています	サポートされています
Link Aggregation Control Protocol (LACP)	サポートされています	サポートされていません
スタンバイインタフェースの使用	サポートされていません	サポートされています
複数スイッチへのまたがり	ベンダー独自のソリューションを使用している場合を除き、サポートされていません	サポートされています
スイッチ構成	必須	必須ではありません
負荷分散ポリシー	サポートされています	適用できません
アグリゲーションのすべてのポートにわたる負荷分散	サポートされています	制限されています[1]
ユーザー定義フローによるリソース管理	サポートされています	サポートされています
リンク保護	サポートされています	サポートされています
バックツーバック並列構成	サポートされています	サポートされていません[2]

[1]アグリゲーションはその VNIC をすべてのポートに分散します。ただし、個々の VNIC が複数のポートに負荷を分散することはできません。

[2]DLMP アグリゲーションは、ほかの宛先システムにパケットを送信する場合、常に中間のスイッチを使用する必要があります。ただし、DLMP アグリゲーションを使用している場合は、スイッチをリンクアグリゲーション用に構成しないでください。

◆ ◆ ◆ 付 録 B

リンクアグリゲーションと IPMP: 機能比較

IPMP とリンクアグリゲーションは、改善されたネットワークパフォーマンスを実現するとともにネットワーク可用性を維持する、異なるテクノロジです。

次の表は、リンクアグリゲーションと IPMP の全般的な比較を示します。

機能	IPMP	リンクアグリゲーション
ネットワークテクノロジのタイプ	レイヤー 3 (IP 層)	レイヤー 2 (リンク層)
構成ツール	`ipadm`	`dladm`
リンクベースの障害検出	サポートされています	サポートされています
プローブベースの障害検出	ICMP ベースであり、介在するレイヤー 2 スイッチの複数のレベルにわたって検査用アドレスと同じ IP サブネット内で定義された任意のシステムをターゲットとします	LACP ベースであり、直接のピアホストまたはスイッチをターゲットとします DLMP アグリゲーションではサポートされていません。
スタンバイインタフェースの使用	サポートされています	トランクアグリゲーションではサポートされていません DLMP アグリゲーションでサポートされています
複数スイッチへのまたがり	サポートされています	トランクアグリゲーションではサポートされていません。一部のベンダーは、複数のスイッチにまたがるための、相互運用不可能な独自の解決を提供しています。 DLMP アグリゲーションでサポートされています

機能	IPMP	リンクアグリゲーション
ハードウェアのサポート	必須ではありません	トランクアグリゲーションには必須です。たとえば、Oracle Solaris システムのトランクアグリゲーションでは、対応するスイッチポートも集約されている必要があります。 DLMP アグリゲーションでは必要ありません
リンク層の要件	ブロードキャスト可能	Ethernet 固有
ドライバフレームワークの要件	なし	GLDv3 フレームワークを使用する必要があります
負荷分散のサポート	サポートされており、カーネルによって制御されます。インバウンド負荷分散は、発信元アドレス選択によって間接的に影響されます。	トランクアグリゲーションだけでサポートされており、管理者が `dladm` コマンドを使用して制御します。インバウンド負荷分散はサポートされます。 個々のインタフェースによるアグリゲーション全体への負荷分散は、DLMP アグリゲーションではサポートされていません。
VNIC と統合する場合のサポートレベル	貧弱なサポート	卓越したサポート
ユーザー定義フローによるリソース管理	サポートされていません	サポートされています
リンク保護	サポートされていません	サポートされています
プロトコル要件	なし	なし

リンクアグリゲーションでは、受信トラフィックは、その標準モードのアグリゲーションを構成する複数のリンクにわたって分散されます。したがって、より多くの NIC を取り付けてアグリゲーションにリンクを追加すると、ネットワークのパフォーマンスが向上します。

DLMP アグリゲーションは複数のスイッチにまたがります。レイヤー 2 テクノロジとして、アグリゲーションはほかの Oracle Solaris 仮想化テクノロジと最適に統合されます。

IPMP のトラフィックは IPMP インタフェースのデータアドレスを使用しますが、これは、それらのデータアドレスが、使用可能なアクティブインタフェースにバインドされるからです。たとえば、すべてのデータトラフィックは 2 つの IP アドレスの間のみを流れるが、それらのトラフィックが同じ接続上を流れるとはかぎらない場

合には、より多くの NIC を追加しても IPMP のパフォーマンスは改善しません。なぜなら、使用可能な IP アドレスは依然として 2 つだけだからです。

トランクアグリゲーションと IPMP は互いに補い合うため、両者を一緒に配備して、ネットワークパフォーマンスと可用性の両方の利点を提供できます。

索引

L