Oracle® Solaris 11.3 ソフトウェアの追加と更新

Part No: E62521
2017 年 3 月

Part No: E62521

ドキュメントのアクセシビリティについて

オラクルのアクセシビリティについての詳細情報は、Oracle Accessibility ProgramのWeb サイト(http://www.oracle.com/pls/topic/lookup?ctx=acc&id=docacc)を参照してください。

Oracle Supportへのアクセス

サポートをご契約のお客様には、My Oracle Supportを通して電子支援サービスを提供しています。詳細情報は(http://www.oracle.com/pls/topic/lookup?ctx=acc&id=info) か、聴覚に障害のあるお客様は (http://www.oracle.com/pls/topic/lookup?ctx=acc&id=trs)を参照してください。

目次

例目次

このドキュメントの使用法

- **概要** – Oracle Solaris 11.3 Image Packaging System (IPS) 機能のソフトウェアインストール機能について説明します。IPS コマンドによって、ソフトウェアパッケージの一覧表示および検索、ソフトウェアのインストールおよび削除、ならびに新しい Oracle Solaris オペレーティングシステムリリースへのアップグレードが可能になります。
- **対象読者** – ソフトウェアのインストールと管理、およびシステムイメージの管理を行うシステム管理者。
- **前提知識** – Oracle Solaris システムの管理経験。

製品ドキュメントライブラリ

この製品および関連製品のドキュメントとリソースは `http://www.oracle.com/pls/topic/lookup?ctx=E62101-01` で入手可能です。

フィードバック

このドキュメントに関するフィードバックを `http://www.oracle.com/goto/docfeedback` からお聞かせください。

♦ ♦ ♦ 第 1 章

1 Image Packaging System の概要

Oracle Solaris Image Packaging System (IPS) は、次のタスクの実行を可能にするフレームワークです。

- ソフトウェアパッケージの一覧表示と検索
- ソフトウエアパッケージのインストール、更新、および削除
- Oracle Solaris オペレーティングシステムの新しいリリースへのアップグレード

IPS インタフェースを使用して、インストールできるパッケージ、インストールできるパッケージのバージョン、およびインストール済みソフトウェアに署名が必要になる状況を制限できます。

この章では、IPS の概念を説明し、ソフトウェアのインストールや更新などのタスクの実行に必要な特権を得る方法を説明します。

Image Packaging System

Oracle Solaris 11 ソフトウェアは、IPS パッケージで配布されます。IPS パッケージは、IPS パブリッシャーが提供する IPS パッケージリポジトリに格納されます。IPS パッケージは、Oracle Solaris 11 のイメージにインストールされます。IPS のコマンド行インタフェースから実行できる機能の一部は、パッケージマネージャーのグラフィカルユーザーインタフェースを使用して実行できます。

IPS ツールは次に示す機能を提供します。パブリッシャーやリポジトリなどの用語の定義については、14 ページの「IPS の概念」を参照してください。

- ソフトウェアパッケージを一覧表示、検索、インストール、インストールの制限、更新、および削除します。
- パッケージパブリッシャーを一覧表示、追加、および削除します。検索の優先順やスティッキネスなどの、パブリッシャーの属性を変更します。署名ポリシーなどのパブリッシャーのプロパティーを設定します。
- 新しいオペレーティングシステムリリースにイメージをアップグレードします。
- 既存の IPS パッケージリポジトリのコピーを作成します。新しいパッケージリポジトリを作成します。『Copying and Creating Package Repositories in Oracle Solaris 11.3』を参照してください。

- パッケージを作成および公開します。『Packaging and Delivering Software With the Image Packaging System in Oracle Solaris 11.3』を参照してください。
- ブート環境およびその他のイメージを作成します。

IPS を実行するには、Oracle Solaris 11 OS を実行している必要があります。Oracle Solaris 11 OS をインストールするには、『Installing Oracle Solaris 11.3 Systems』を参照してください。

IPS の概念

このセクションでは、IPS に関連する用語および概念を定義します。

IPS パッケージ

IPS パッケージはマニフェストというテキストファイルで定義します。パッケージマニフェストには、鍵/値のペアとおそらくデータペイロードの定義された形式でパッケージアクションが記述されます。パッケージアクションには、ファイル、ディレクトリ、リンク、ドライバ、依存関係、グループ、ユーザー、ライセンス情報が含まれます。パッケージアクションは、パッケージのインストール可能なオブジェクトを表します。`set` アクションと呼ばれるアクションは、分類、サマリー、説明などのパッケージメタデータを定義します。

パッケージアクションおよびアクションキーを指定して、パッケージを検索できます。パッケージアクションの説明については、「Package Content: Actions」 in 『Packaging and Delivering Software With the Image Packaging System in Oracle Solaris 11.3』または `pkg(5)` マニュアルページを参照してください。

制約パッケージおよびグループパッケージは、ファイルなどの内容を提供しません。制約パッケージおよびグループパッケージは、関連するパッケージのセットのインストールに役立つ依存関係を指定します。

制約パッケージ

制約パッケージは、互換性を持つパッケージバージョンスペース内にサーフェスを定義するために `incorporate` 依存関係を指定します。制約パッケージは、同時に作成されるソフトウェアパッケージのセットを定義するために使用され、個別にバージョン管理されません。`incorporate` 依存関係は、ソフトウェアの互換性のあるバージョンがまとめてインストールされるようにするために Oracle Solaris でよく使用されます。

incorporate 依存関係は、このイメージ内にインストールできる、そのパッケージのバージョンを制約します。パッケージは、incorporate 依存関係として指定するだけではインストールされません。パッケージをほかの何らかの方法でインストールする場合は (たとえば、require 依存関係でもある場合や、パッケージを明示的にインストールする場合)、incorporate 依存関係によって規定されたバージョンのみをインストールできます。たとえば、インストールされた制約パッケージ内で incorporate 依存関係として指定されたパッケージがバージョン値 1.4.3 を持つ場合、1.4.3 未満か 1.4.4 以上のバージョン値を持つそのパッケージのバージョンをインストールすることはできません。たとえば、バージョン値 1.4.3.7 を持つパッケージのバージョンはインストールできます。

制約パッケージ内で incorporate 依存関係として指定されているパッケージ自体が、制約パッケージの場合もあります。このように、制約パッケージのマニフェスト内でパッケージが指定されていない場合でも、多くのパッケージが制約パッケージによって影響を受ける可能性があります。そのインストールが制約パッケージによって影響を受けるパッケージは、その制約パッケージによって制約されます。制約パッケージ B-constraint に incorporate 依存関係を持つ制約パッケージ A-constraint を更新すると、B-constraint と、B-constraint によって制約されているほかのすべてのパッケージも更新されます。B-constraint を A-constraint とは別に更新しようとすると、エラーになることがあります。

制約パッケージは、使用中のサポート可能なイメージを容易に維持できるようにするために、制約されたパッケージを同期的に強制アップグレードします。一般的に、制約パッケージの incorporate 依存関係であるパッケージをインストールしたり更新したりしてはいけません。代わりに、制約パッケージを更新する必要があります。制約されているパッケージはアンインストールできますが、制約されているパッケージをインストールまたは更新する場合、バージョンが制約されます。関連情報については、114 ページの「制約パッケージによって指定されたバージョン制約の緩和」を参照してください。

pkg:/entire パッケージは、ほかの多くの制約パッケージ上の incorporate 依存関係を指定して、イメージにインストールされているほとんどのシステムソフトウェアのバージョンを制約する特殊な制約パッケージです。

注意 - pkg:/ パッケージを削除しないでください。pkg:/ パッケージは、システムパッケージのバージョンを制約して、結果として得られるパッケージセットがサポート可能なイメージになるようにします。適切なシステム更新と正しいパッケージ選択は、この制約パッケージに依存します。pkg:/ パッケージを削除すると、サポートされないシステムになります。

グループパッケージ

グループパッケージは、機能やツールを構成する一連のパッケージを指定します。グループパッケージをインストールすると、そのグループパッケージ内のすべての

group 依存関係パッケージがインストールされます。グループパッケージ内で group 依存関係としてに指定されたパッケージは、パッケージのバージョンを指定しません。グループパッケージは、コンテンツ管理ツールであり、バージョン管理ツールではありません。

グループパッケージは、それらの group 依存関係に指定されているパッケージを提供しますが、それらのパッケージがすでにインストールされているか、回避リストに含まれている場合は除きます。回避リストについては、119 ページの「グループパッケージに含まれる一部のパッケージのインストールの回避」を参照してください。

たとえば、group/feature/storage-server パッケージは、ストレージに関連するドライバ、サービス、ファイルシステム、I/O コンポーネント、ライブラリ、およびユーティリティーを提供します (これらがまだインストールされていない場合)。group/system/solaris-minimal-server パッケージは、サポートされる Oracle Solaris 環境に最低限必要なパッケージのセットを提供します。グループパッケージで提供されているすべてのパッケージを一覧表示する方法の例については、32 ページの「グループパッケージ内のすべてのインストール可能なパッケージの一覧表示」を参照してください。

グループパッケージをアンインストールしても、group 依存関係内に指定されているすべてのパッケージが必ずしもアンインストールされるわけではありません。グループパッケージをアンインストールするとき、ほかのソフトウェアによって必要とされているパッケージで依然としてインストールされているものは、アンインストールされません。

障害管理リソース識別子

それぞれのパッケージは障害管理リソース識別子 (FMRI) によって表されます。パッケージの完全な FMRI は、次の形式のスキーム、パブリッシャー、パッケージ名、およびバージョン文字列で構成されます。

scheme://*publisher*/*name*@*version*

スキーム、パブリッシャー、およびバージョン文字列はオプションです。 IPS コマンドオペランドでは、パッケージを一意的に識別するパッケージ名の最小部分を使用することができ、? および * の文字を、1 つ以上のパッケージと一致させるための glob(3C) スタイルのワイルドカードとして使用できます。

すべての IPS パッケージ FMRI のスキームは pkg です。suri ストレージライブラリの次のサンプルパッケージ FMRI では、solaris がパブリッシャー、system/library/storage/suri がパッケージ名、0.5.11,5.11-0.175.3.0.0.19.0:20150329T164922Z がバージョンです。

```
pkg://solaris/system/library/storage/suri@0.5.11,5.11-0.175.3.0.0.19.0:20150329T164922Z
```

スキーム

`pkg`

パブリッシャー

`solaris`

パブリッシャーを指定する場合、パブリッシャー名の前に `pkg://` または `//` を付ける必要があります。

パッケージ名

`system/library/storage/suri`

パッケージ名は、スラッシュ (*/*) 文字で区切られた任意の数のコンポーネントから成る階層形式です。IPS コマンドでは、コマンドで使用するパッケージ名によってそのパッケージが一意に識別される場合、パッケージ名の前のコンポーネントは省略できます。完全なパッケージ名を指定するがパブリッシャーは省略する場合、完全なパッケージ名の前に、`pkg://` または `//` ではなく、`pkg:/` または `/` を付けることができます。短縮したパッケージ名を指定する場合は、パッケージ名の左側にほかの文字を使用しないでください。

バージョン

バージョン文字列は次の 4 つの部分で構成され、タイムスタンプの形式は *date*`T`*time*`Z` です。

component_version`,`*release*`-`*branch_version*`:`*time_stamp*

コンポーネントバージョン: `0.5.11`

オペレーティングシステムに緊密に結合されたコンポーネントの場合、コンポーネントバージョンは通常、そのバージョンのオペレーティングシステムでの `uname -r` の値を含みます。独自の開発ライフサイクルを持つコンポーネントの場合、コンポーネントバージョンはドットで区切られたリリース番号 (2.2.29 など) です。

リリース: `5.11`

リリースはコンマ (,) のあとに続ける必要があります。リリースは、パッケージの内容が構築されたオペレーティングシステムのバージョンを指定します。

ブランチバージョン: `0.175.3.0.0.19.0`

ブランチバージョンはハイフン (-) の後に続ける必要があります。ブランチバージョンはベンダー固有の情報を提供します。

Oracle Solaris パッケージでは、パッケージ FMRI のバージョン文字列のブランチバージョン部分に次の情報が示されます。

メジャーリリース番号: `0.175`

メジャーまたはマーケティング開発リリースのビルド番号。この例では、`0.175` は Oracle Solaris 11 を示します。

更新リリース番号: `3`

この Oracle Solaris リリースの更新リリース番号。更新値は、Oracle Solaris リリースの FCS (First Customer Shipment) では 0、そのリリースの最初の更新では 1、そのリリースの 2 回目の更新では 2 というようになります。この例では、`3` は Oracle Solaris 11.3 を示します。

SRU 番号: `0`

この更新リリースのサポート・リポジトリの更新 (SRU) 番号。SRU はほぼ 1 か月に 1 回、更新され、バグを修正したり、セキュリティーの問題を修正したり、新しいハードウェアに対するサポートを提供します。SRU に新機能は含まれません。Oracle サポート・リポジトリはサポート契約下のシステムでのみ使用できます。

予約済み: `0`

このフィールドは、Oracle Solaris パッケージで現在使用されていません。

リリースまたは SRU ビルド番号: `19`

SRU のビルド番号、またはメジャーリリースの更新番号。

ナイトリービルド番号: `0`

個々のナイトリービルドのビルド番号。

パッケージが IDR (Interim Diagnostic or Relief) 更新である場合、パッケージ FMRI のブランチバージョンには次の 2 つの追加フィールドが含まれます。IDR は、正式なパッケージ更新が発行されるまで、顧客の問題の診断を支援したり、問題の一時的な解決策を提供したりするためのパッケージ更新です。IDR の詳細については、89 ページの「IDR カスタムソフトウェア更新のインストール」を参照してください。次の例は `idr824` (FMRI `pkg://solaris/idr824@4,5.11:20131114T034951Z` を含む) についてのもので、`pkg:/system/library@0.5.11-0.175.1.6.0.4.2.824.4` などのパッケージを含んでいます。

IDR: `824`

IDR の名前。

IDR ID: `4`

IDR のバージョン。

タイムスタンプ: `20150329T164922Z`

タイムスタンプはコロン (:) の後に続ける必要があります。タイムスタンプは、ISO-8601 基本形式 (*YYYYMMDD*T*HHMMSS*Z) のパッケージが発行された日時です。

パブリッシャー、リポジトリ、およびパッケージアーカイブ

パブリッシャーは、1 つ以上のパッケージを提供する人または組織を示します。パブリッシャーは、パッケージリポジトリまたはパッケージアーカイブを使用してパッケージを配布できます。パブリッシャーは、好きな検索順序で構成できます。パッケージインストールコマンドを指定し、パッケージ仕様にパブリッシャーの名前が含まれない場合、そのパッケージに対して、検索順序の先頭のパブリッシャーが検索されます。指定されたパッケージ FMRI パターンに一致するものが見つからない場合は、検索順序の 2 番目のパブリッシャーが検索されるというように、パッケージが見つかるか、すべてのパブリッシャーが検索されるまで繰り返されます。

リポジトリは、パッケージが公開される場所であり、またそれらのパッケージが取得される場所です。場所は URI (Universal Resource Identifier) によって指定されます。カタログは、リポジトリ内のすべてのパッケージのリストです。

パッケージアーカイブは、パブリッシャーの情報と、そのパブリッシャーによって提供された 1 つ以上のパッケージを含むファイルです。

リポジトリの起点とミラー

起点は、パッケージのメタデータ (カタログ、マニフェスト、検索インデックスなど) とパッケージの内容 (ファイル) の両方を含むパッケージリポジトリです。イメージ内の特定のパブリッシャーに対して複数の起点が構成されている場合、IPS クライアントは、パッケージデータの取得元として最適な起点を選択しようとします。

ミラーは、パッケージの内容のみを含むパッケージリポジトリです。ミラーリポジトリからパッケージをインストールおよび更新する IPS クライアントは、起点リポジトリからメタデータをダウンロードする必要があります。IPS クライアントがパッケージのコンテンツをミラーからダウンロードする場合でも、クライアントはパブリッシャーのカタログを取得するために起点にアクセスします。パブリッシャーに対してミラーが構成されている場合、IPS クライアントは、パッケージの内容の取得にミラーを優先します。イメージ内の特定のパブリッシャーに対して複数のミラーが構成されている場合、IPS クライアントは、パッケージの内容の取得元として最適なミラーを選択しようとします。すべてのミラーにアクセスできない、必要な内容がない、または遅くなる場合、IPS クライアントは起点から内容を取得します。詳細は、pkg(5) マニュアルページのパブリッシャーおよびリポジトリに関するセクションを参照してください。

注記 - ミラーリポジトリとして指定されたリポジトリの内容とメタデータの両方が完全であっても、同じパッケージの同じバージョンが、同じパブリッシャーの起点リポジトリ内にも存在しないかぎり、そのミラーリポジトリの内容にアクセスすることはできません。

イメージとブート環境

イメージは、IPS パッケージをインストールでき、その他の IPS 操作を実行できる場所です。

ブート環境 (BE) は、イメージのブート可能なインスタンスです。物理または仮想システム上に複数の BE を維持することができ、異なるオペレーティングシステムバージョンなどの異なるソフトウェアバージョンを各 BE にインストールできます。システムをブートするとき、システム上の任意の BE にブートすることを選択できます。パッケージ操作の結果として、新しい BE が自動的に作成されることがあります。新しい BE が自動的に作成されるかどうかは、122 ページの「ブート環境ポリシー イメージのプロパティー」で説明するように、イメージポリシーに依存します。また、44 ページの「ブート環境オプション」に記載されているオプションを指定することによって、新しい BE を明示的に作成することもできます。新規 BE を作成するための `beadm` コマンドの使用方法については、『Creating and Administering Oracle Solaris 11.3 Boot Environments』を参照してください。

パッケージは、BE の一部であるファイルシステムにのみインストールできます。たとえば、デフォルトの Oracle Solaris 11 インストールでは、`rpool/ROOT/`*`BEname`*`/` の下のデータセットのみがパッケージ操作用にサポートされます。

もう 1 つのイメージの例として、Oracle Solaris ゾーンがあります。非大域ゾーンは、大域ゾーンと呼ばれる Oracle Solaris オペレーティングシステムのインスタンス内で作成される、仮想化されたオペレーティングシステム環境です。大域ゾーンは親イメージで、その大域ゾーン内の非大域ゾーンはその大域ゾーンの子イメージです。IPS コマンド出力では、非大域ゾーンは親の大域ゾーンイメージにリンクされていることから、しばしばリンクされたイメージと呼ばれます。

64 ページの「非大域ゾーンの操作」に記載されているように、大域ゾーンで実行される IPS コマンドは、非大域ゾーンに影響することがあります。大域ゾーンで実行される IPS コマンドは、カーネルゾーン (`solaris-kz` ブランドゾーン) または Oracle Solaris 10 ゾーン (`solaris10` branded zones) に影響しません。このガイドで、「非大域ゾーン」とは `solaris` ブランドの Oracle Solaris 11 非大域ゾーンを意味します。ゾーンについては、『Introduction to Oracle Solaris Zones』を参照してください。

パッケージのファセットとバリアント

ソフトウェアには、オプションのコンポーネントや、相互に排他的なコンポーネントが含まれることがあります。オプションのコンポーネントの例には、ロケールやドキュメントがあります。相互に排他的なコンポーネントの例には、SPARC バイナリと x86 バイナリや、デバッグバイナリと非デバッグバイナリなどがあります。IPS では、オプションのコンポーネントをファセットと呼び、相互に排他的なコンポーネントをバリアントと呼びます。

ファセットとバリアントは、イメージに設定される特殊プロパティーです。ファセットとバリアントは、パッケージマニフェスト内のアクションに設定されるタグでもあります。アクションのファセットおよびバリアントタグの値とイメージに設定されたファセットおよびバリアントの値の比較により、そのパッケージアクションがインストール可能かどうかを判別します。たとえば、イメージ内で特定のロケールファセットを `false` に設定した場合、そのファセットを指定しているファイルアクションはどれもインストールされず、そのファセットを指定している現在インストール済みのファイルアクションはアンインストールされます。

イメージに設定されているファセットとバリアントの値を表示または変更する方法など、ファセットとバリアントの詳細については、107 ページの「オプションのコンポーネントのインストールの制御」を参照してください。

インストール権限

第2章「ソフトウェアパッケージに関する情報の取得」で説明しているコマンドには、特別な権限を使用する必要はありません。IPS パッケージのインストールや、オペレーティングシステムのアップグレード、パブリッシャーおよびイメージプロパティーの構成などのタスクには、さらに強い権限が必要になります。必要な権限を取得するには、次のいずれかの方法を使用します。役割およびプロファイルの詳細 (必要な役割またはプロファイルを判別する方法を含む) は、『Securing Users and Processes in Oracle Solaris 11.3』を参照してください。

役割

`roles` コマンドを使用して、自分に割り当てられている役割を一覧表示します。その役割を引き受けるには、役割の名前を指定して `su` コマンドを使用します。この役割として、その役割に割り当てられている権利プロファイルで許可されているすべてのコマンドを実行できます。たとえば、役割にソフトウェアインストールに関連する権利プロファイルが割り当てられている場合、`pkg` および `beadm` コマンドを実行して、パッケージのインストールと更新、およびブート環境の管理を行うことができます。

権利プロファイル

`profiles` コマンドを使用して、自分に割り当てられている権利プロファイルを一覧表示します。権利プロファイルで実行が許可されているコマンドを実行するには、次のいずれかの方法を使用します。

- `pfbash` または `pfksh` などのプロファイルシェルを使用します。
- 実行するコマンドの前に `pfexec` コマンドを使用します。一般に、実行する各権限付きコマンドで `pfexec` コマンドを指定する必要があります。

sudo コマンド

サイトのセキュリティーポリシーに応じて、自分のユーザーパスワードで `sudo` コマンドを使用し、特権コマンドを実行できる場合があります。

♦ ♦ ♦ 第 2 章

2 ソフトウェアパッケージに関する情報の取得

この章では、パッケージに関する次の種類の情報を得るためのコマンドについて説明します。

- パッケージがインストールされているか、またはインストールや更新ができるかどうか
- パッケージの説明、サイズ、およびバージョン
- グループパッケージに含まれるパッケージ
- 指定したファイルを提供するパッケージ
- 指定した SMF サービスを提供するパッケージ
- 指定したパッケージに依存するパッケージ
- 特定のカテゴリに含まれるパッケージ

構成済みのパブリッシャーに対してパッケージリポジトリの内容が変更された可能性がある場合、セッションの開始時に使用可能なパッケージの一覧を更新して、最新情報を受け取っていることを確認します。パッケージの一覧を更新するには、`pkg refresh` コマンドを実行します。たとえば、リポジトリが新規パッケージによって更新された場合、`pkg refresh` を実行するまでは、これらの新規パッケージが使用可能なパッケージの一覧に表示されない場合があります。

この章で説明されているコマンドのすべてのオプションの完全なリストについては、pkg(1) のマニュアルページを参照してください。

パッケージのインストール状態情報の表示

`pkg list` コマンドは、現在のイメージにパッケージがインストールされているかどうか、および更新が使用可能かどうかを示します。オプションやオペランドを指定しない場合、`pkg list` コマンドによって、現在のイメージにインストールされているすべてのパッケージが一覧表示されます。結果を絞り込むには、パッケージ名を 1 つ以上指定します。パッケージ名にはワイルドカードを使用できます。引数が直接 `pkg` に渡され、シェルで展開されないようにするために、ワイルドカードは引用符で囲みます。

インストール済みのパッケージ

次の例に示すように、pkg list コマンドは、一致するパッケージごとに 1 行の情報を表示します。I 列内の「i」は、これらのパッケージがこのイメージにインストールされていることを示します。

```
$ pkg list '*toolkit'
NAME (PUBLISHER)                  VERSION                  IFO
isvtoolkit (isvpub)               1.0                      i--
system/dtrace/dtrace-toolkit      0.99-0.175.2.0.0.34.0    i--
```

丸括弧で囲まれたパブリッシャー名は、isvpub のパブリッシャーが、このイメージのパブリッシャーの検索順序で先頭のパブリッシャーではないことを示します。このイメージにインストールされた dtrace-toolkit パッケージは、検索順序の先頭のパブリッシャーによって発行されています。

インストール可能なパッケージ

このイメージにインストールされているパッケージと、インストールされていないが、インストール可能なパッケージの最新バージョンを一覧表示するには、-a オプションを使用します。

```
$ pkg list -a '*toolkit'
NAME (PUBLISHER)                  VERSION                  IFO
image/nvidia/cg-toolkit           3.0.15-0.175.2.0.0.17.0  ---
isvtoolkit (isvpub)               1.0                      i--
system/dtrace/dtrace-toolkit      0.99-0.175.2.0.0.34.0    i--
```

この出力は、image/nvidia/cg-toolkit パッケージが使用可能であり、このイメージにインストールできることを示しています。

最新のパッケージ

このイメージにインストールできないパッケージも含めて、一致するすべてのパッケージの最新バージョンを一覧表示するには、-n オプションを使用します。

```
$ pkg list -n '*toolkit'
NAME (PUBLISHER)                  VERSION                  IFO
developer/dtrace/toolkit          0.99-0.173.0.0.0.1.0     --r
image/nvidia/cg-toolkit           3.0.15-0.175.2.0.0.17.0  ---
isvtoolkit (isvpub)               1.0                      i--
system/dtrace/dtrace-toolkit      0.99-0.175.2.0.0.35.0    ---
```

developer/dtrace/toolkit パッケージと system/dtrace/dtrace-toolkit パッケージは、このイメージにインストールできません。このことは、これ

らのパッケージが -a オプションでは一覧表示されなかったことからわかります。developer/dtrace/toolkit パッケージは名前が変更されています。詳細は、26 ページの「パッケージの名前変更と廃止」を参照してください。

このイメージに現在インストールされているバージョンよりも新しいバージョンの dtrace-toolkit パッケージが、構成済みのパブリッシャーに存在します。次の例の「Reason」行に示されているように、dtrace-toolkit パッケージが依存関係になっているほかのパッケージも更新した場合、新しいバージョンがインストール可能になることもあります。簡潔に示すために、次のコマンド出力ではタイムスタンプが省略されています。pkg update および pkg install コマンドについては、第3章「ソフトウェアパッケージのインストールおよび更新」で説明します。

```
$ pkg update -nv dtrace-toolkit
No updates are available for this image.
$ pkg install -nv dtrace-toolkit@0.99-0.175.2.0.0.35
pkg install: No matching version of system/dtrace/dtrace-toolkit can be installed:
  Reject:  pkg://solaris/system/dtrace/dtrace-toolkit@0.99,5.11-0.175.2.0.0.35.0
  Reason:  This version is excluded by installed incorporation
pkg://solaris/consolidation/osnet/osnet-incorporation@0.5.11,5.11-0.175.2.0.0.34.0
```

更新が利用可能なパッケージ

-u オプションは、新しいバージョンが使用可能な、インストール済みの一致するすべてのパッケージを一覧表示します。24 ページの「最新のパッケージ」に例示されているように、より新しいバージョンがあるパッケージの数が、このイメージ内で更新可能なパッケージの数より多い場合があります。パッケージは、イメージの制約で許可されたバージョンにのみ更新できます。この制約は、インストール済みパッケージの依存関係およびパブリッシャーの構成によってイメージに課せられるものです。

```
$ pkg list -u '*toolkit'
NAME (PUBLISHER)                  VERSION                    IFO
system/dtrace/dtrace-toolkit      0.99-0.175.2.0.0.34.0      i--
```

使用可能なすべてのパッケージ

このイメージにインストールできないパッケージも含めて、一致するすべてのパッケージの使用可能なすべてのバージョンを一覧表示するには、-af オプションを使用します。-f オプションは、-a オプションを使用しないと使用できません。これらの結果を絞り込むために、バージョン文字列の一部を指定する場合もあります。特殊なバージョン文字列 @latest を指定すると、-n オプションで表示されるのと同じ結果が表示されます。

```
$ pkg list -af '*toolkit@0.99-0.175.2'
$ pkg list -af '*toolkit@latest'
```

パッケージの名前変更と廃止

次の例で、O 列の「r」はパッケージの名前が変更されていることを示し、O 列の「o」はパッケージが廃止されたことを示します。

```
$ pkg list -n developer/dtrace/toolkit database/mysql-50 web/amp
NAME (PUBLISHER)                  VERSION                  IFO
database/mysql-50                 5.0.91-0.171             --o
developer/dtrace/toolkit          0.99-0.173.0.0.0.1.0     --r
web/amp                           0.5.11-0.174.0.0.0.0.0   --r
```

これらのパッケージはどれもインストール可能ではありません。廃止されたパッケージをインストールしようとすると、インストールは失敗し、このイメージに必要な更新が存在しないというメッセージが表示されます。名前が変更されたパッケージをインストールしようとすると、システムは、名前が変更されたパッケージの名前が変更された場所に、パッケージのインストールを試行します。

`pkg info` コマンドを使用して、名前が変更されたパッケージの新しい名前を判別します。パッケージはインストールされていないため、`-r` オプションを使用して、構成済みのパッケージリポジトリを照会します。次の例に示されている「Renamed to」行を参照してください。

```
$ pkg info -r web/amp
          Name: web/amp
       Summary:
         State: Not installed (Renamed)
    Renamed to: group/feature/amp@0.5.11-0.174.0.0.0.0.0
                consolidation/ips/ips-incorporation
     Publisher: solaris
       Version: 0.5.11
 Build Release: 5.11
        Branch: 0.174.0.0.0.0.0
Packaging Date: September 21, 2011 07:15:02 PM
          Size: 5.45 kB
          FMRI: pkg://solaris/web/amp@0.5.11,5.11-0.174.0.0.0.0.0:20110921T191502Z
```

`web/amp` パッケージをインストールしようとすると、`group/feature/amp` パッケージが代わりにインストールされます (まだインストールされておらず、このイメージにインストール可能な場合)。

次の例で、「Renamed to」に示すパッケージはすでにインストールされているため、パッケージシステムから、必要な更新が存在しないという報告が出されます。

```
$ pkg info -r developer/dtrace/toolkit
          Name: developer/dtrace/toolkit
       Summary:
         State: Not installed (Renamed)
    Renamed to: pkg:/system/dtrace/dtrace-toolkit@0.99,5.11-0.173.0.0.0.0.0
                consolidation/osnet/osnet-incorporation
     Publisher: solaris
       Version: 0.99
 Build Release: 5.11
        Branch: 0.173.0.0.0.1.0
Packaging Date: August 26, 2011 02:55:51 PM
          Size: 5.45 kB
```

```
          FMRI: pkg://solaris/developer/dtrace/toolkit@0.99,5.11-0.173.0.0.0.1.0:
20110826T145551Z
$ pkg list dtrace-toolkit
NAME (PUBLISHER)                  VERSION                    IFO
system/dtrace/dtrace-toolkit      0.99-0.175.2.0.0.34.0      i--
$ pkg install developer/dtrace/toolkit
No updates necessary for this image.
```

特定バージョンに凍結されているパッケージ

F 列の「f」は、そのパッケージが凍結されていることを示します。パッケージが凍結されている場合は、凍結されたバージョンと一致するパッケージのみをインストールまたは更新できます。パッケージの凍結については、113 ページの「指定したバージョンへのパッケージのロック」を参照してください。

```
$ pkg list openssl
NAME (PUBLISHER)                  VERSION                    IFO
library/security/openssl          1.0.1.6-0.175.2.0.0.34.0 if-
```

パッケージの説明またはライセンスの表示

`pkg info` コマンドは、名前、説明、インストール状態、バージョン、パッケージ化の日付、パッケージのサイズ、完全な FMRI などのパッケージに関する情報を表示します。オプションやオペランドを指定しない場合、`pkg info` コマンドによって、現在のイメージにインストールされているすべてのパッケージに関する情報が表示されます。結果を絞り込むには、パッケージ名を 1 つ以上指定します。パッケージ名にはワイルドカードを使用できます。引数が直接 `pkg` に渡され、シェルで展開されないようにするために、ワイルドカードは引用符で囲みます。

`info` および `list` サブコマンドはいずれも、パッケージ名、パブリッシャー、およびバージョン情報を表示します。`pkg list` コマンドは、パッケージの更新が存在するかどうか、このイメージに更新をインストールできるかどうか、およびパッケージが廃止されたり、名前が変更されたり、凍結されたりしていないかどうかを表示します。`pkg list` コマンドは、パッケージのサマリーと完全な FMRI も表示できます。`pkg info` コマンドは、パッケージのサマリー、説明、カテゴリ、サイズを表示し、個別にライセンス情報を表示できます。

パッケージの説明、サイズ、完全な FMRI の表示

`pkg list -s` コマンドを使用して、パッケージのサマリーを表示できます。

```
$ pkg list -s entire
NAME (PUBLISHER)    SUMMARY
entire              Incorporation to lock all system packages to the same build
```

`pkg list -v` コマンドは、完全なパッケージ FMRI を一覧表示します。

```
$ pkg list -v entire
FMRI                                                                    IFO
pkg://solaris/entire@0.5.11,5.11-0.175.2.0.0.34.0:20140303T182643Z  ---
```

`pkg info` コマンドは、詳細な情報を表示します。

```
$ pkg info entire
               Name: entire
            Summary: entire incorporation including Support Repository Update (Oracle Solaris
 11.3.13.4.0).
        Description: This package constrains system package versions to the same
                     build.  WARNING: Proper system update and correct package
                     selection depend on the presence of this incorporation.
                     Removing this package will result in an unsupported system.
                     For more information see:
                     https://support.oracle.com/rs?type=doc&id=2045311.1
           Category: Meta Packages/Incorporations
              State: Installed
          Publisher: solaris
            Version: 0.5.11 (Oracle Solaris 11.3.13.4.0)
    Build Release: 5.11
             Branch: 0.175.3.13.0.4.0
     Packaging Date: September 29, 2016 05:55:02 PM
               Size: 5.46 kB
               FMRI: pkg://solaris/entire@0.5.11,5.11-0.175.3.13.0.4.0:20160929T175502Z
```

26 ページの「パッケージの名前変更と廃止」に示すように、`pkg info` コマンドを使用して、名前が変更されたパッケージの新しい名前を見つけることができます。

パッケージライセンスの表示

一致するパッケージのライセンステキストを表示するには、`--license` オプションを使用します。インストールする予定のパッケージのライセンステキストを表示するには、`-r` オプションを使用します。

```
$ pkg info -r --license osnet-incorporation
You acknowledge that your use of this Oracle Solaris software product
is subject to, and may not exceed the use for which you are authorized,
(i) the license or cloud services terms that you accepted when you
obtained the right to use Oracle Solaris software; or (ii) the license
terms that you agreed to when you placed your Oracle Solaris software
order with Oracle; or (iii) the Oracle Solaris software license terms
included with the hardware that you acquired from Oracle; or, if (i),
(ii) or (iii) are not applicable, then, (iv) the OTN License Agreement
for Oracle Solaris (which you acknowledge you have read and agree to)
available at
http://www.oracle.com/technetwork/licenses/solaris-cluster-express-license-167852.html.
Note: Software downloaded for trial use or downloaded as replacement
media may not be used to update any unsupported software.
```

ライセンス情報はきわめて長くなることがあります。`--license` オプションが指定されない場合に `pkg info` コマンドで表示されるその他の情報は表示されません。そのライセンスに同意する必要があるパッケージを一覧表示するには、33 ページの「ライセンス要件の表示」を参照してください。

パッケージマニフェストからの情報の表示

pkg contents コマンドは、パッケージのファイルシステムの内容を表示します。オプションやオペランドを指定せずにこのコマンドを使用すると、現在のイメージにインストールされているすべてのパッケージのパス情報が表示されます。表示するパッケージの内容を指定するには、コマンドオプションを使用します。結果を絞り込むには、パッケージ名を 1 つ以上指定します。パッケージ名にはワイルドカードを使用できます。引数が直接 pkg に渡され、シェルで展開されないようにするために、ワイルドカードは引用符で囲みます。

contents サブコマンドと search サブコマンドはどちらも、パッケージの内容をクエリーします。pkg contents コマンドは、パッケージのアクションと属性を表示します。pkg search コマンドは、クエリーに一致するパッケージを一覧表示します。34 ページの「pkg search コマンドと pkg contents コマンドの比較」も参照してください。

パッケージによってインストールされるファイルの一覧表示

次の例では、pkg contents コマンドのデフォルト動作、つまり、このイメージにインストール可能なファイルシステムオブジェクトごとの path 属性値の表示が示されています。

```
$ pkg contents amp
pkg: This package delivers no filesystem content, but may contain metadata. Use
the -o option to specify fields other than 'path', or use the -m option to show
the raw package manifests.
$ pkg contents zip
PATH
usr/bin/zip
usr/bin/zipcloak
usr/bin/zipnote
usr/bin/zipsplit
usr/share/man/man1/zip.1
usr/share/man/man1/zipcloak.1
usr/share/man/man1/zipnote.1
usr/share/man/man1/zipsplit.1
```

pkg contents コマンドは、このイメージにインストール可能な内容のみ表示します。パッケージマニフェストを表示した場合 (-m オプションを使用)、zip パッケージに 12 個のファイルアクションがあることがわかります。この出力に表示されていない 4 つのファイルは、このイメージにインストールできないファイルです。このイメージは x86 アーキテクチャーです。表示されていないファイルは、SPARC アーキテクチャーの 4 つの /usr/bin ファイルです。バリアントとファセットについては、107 ページの「オプションのコンポーネントのインストールの制御」を参照してください。

パス以外の情報の表示

パス以外の情報を表示するには、またはパス情報のサブセットを表示するには、`pkg contents` コマンドの `-t`、`-a`、および `-o` オプションを使用します。

`-t` オプションは、`file`、`link`、`depend` などの選択するアクションタイプを指定します。パッケージアクションのリストについては、pkg(5) のマニュアルページを参照してください。コンマ区切りリストで複数のアクションタイプを指定するか、`-t` オプションを複数回指定できます。

`-a` オプションは、選択するアクションの属性値を指定します。各アクションタイプの属性のリストについては、pkg(5) のマニュアルページを参照してください。`-a` オプションは何度でも指定できます。

`-o` オプションは、表示する出力を指定します。`-a` オプションの説明に示されている属性の名前、または pkg(1) のマニュアルページにリストされている疑似属性のいずれかを指定できます。コンマ区切りリストで複数の属性引数を指定するか、`-o` オプションを複数回指定できます。それぞれの `-o` 引数は出力の 1 つの列です。

例 1 パッケージによってインストールされるファイルの属性の表示

この例では、29 ページの「パッケージによってインストールされるファイルの一覧表示」の例にリストされているパスの追加の属性が表示されています。

```
$ pkg contents -o owner,group,mode,path zip
OWNER GROUP MODE PATH
root  bin   0555 usr/bin/zip
root  bin   0555 usr/bin/zipcloak
root  bin   0555 usr/bin/zipnote
root  bin   0555 usr/bin/zipsplit
root  bin   0444 usr/share/man/man1/zip.1
root  bin   0444 usr/share/man/man1/zipcloak.1
root  bin   0444 usr/share/man/man1/zipnote.1
root  bin   0444 usr/share/man/man1/zipsplit.1
```

`-o` オプションが指定されていない場合に表示されるデフォルトの出力は、`path` 属性の値です。次の出力は、`oracle-rdbms-server-12-1-preinstall` パッケージがファイル、リンク、またはディレクトリを直接提供しないことを示しています。

```
$ pkg contents -r oracle-rdbms-server-12-1-preinstall
pkg: This package delivers no filesystem content, but may contain metadata. Use
the -o option to specify fields other than 'path', or use the -m option to show
the raw package manifests.
```

次のコマンドはすべて、`oracle-rdbms-server-12-1-preinstall` パッケージに `set`、`signature`、および `depend` アクションが含まれることを示しています。

```
$ pkg contents -rm oracle-rdbms-server-12-1-preinstall
$ pkg contents -ro action.name oracle-rdbms-server-12-1-preinstall
$ pkg contents -ro action.raw oracle-rdbms-server-12-1-preinstall
```

例 2 アクションタイプの指定

次のコマンドは、oracle-rdbms-server-12-1-preinstall パッケージ内の各依存関係の、依存関係のタイプとパッケージ名を表示します。

```
$ pkg contents -rt depend -o type,fmri oracle-rdbms-server-12-1-preinstall
TYPE    FMRI
group   x11/diagnostic/x11-info-clients
group   x11/library/libxi
group   x11/library/libxtst
group   x11/session/xauth
require compress/unzip
require developer/assembler
require developer/build/make
```

例 3 アクション属性の指定

一部の依存関係にのみ関心がある場合は、-a オプションを使用して選択範囲を絞り込みます。

```
$ pkg contents -rt depend -a fmri='*lib*' -o type,fmri oracle-rdbms-server-12-1-preinstall
TYPE  FMRI
group x11/library/libxi
group x11/library/libxtst
```

デフォルトで、出力は、パスまたは -o オプションで指定された最初の属性によってソートされます。-s オプションを使用して、別の属性をソート鍵として指定できます。-s オプションは複数回指定できます。

次の例では、指定された属性は set アクションに固有であり、set アクションタイプを指定する必要はありません。

```
$ pkg contents -ra name=pkg.summary -a name=pkg.description -o name,value -s value oracle-rdbms-
server-12-1-preinstall
NAME            VALUE
pkg.summary     Prerequisite package for Oracle Database 12.1
pkg.description Provides the set of Oracle Solaris packages required for installation and
 operation of Oracle Database 12.
```

例 4 メディエーションに含まれているすべてのリンクの表示

次のコマンドは、python-34 パッケージによって提供される python メディエーションに参加しているリンクの path と target を示しています。複数バージョンの調整については、116 ページの「デフォルトのアプリケーション実装の指定」を参照してください。

```
$ pkg contents -a mediator=python -o path,target python-34
PATH                              TARGET
usr/bin/2to3                      2to3-3.4
usr/bin/idle                      idle3.4
usr/bin/pydoc                     pydoc3.4
usr/bin/python                    python3.4
usr/bin/python-config             python3.4-config
usr/bin/pyvenv                    pyvenv-3.4
```

```
usr/lib/amd64/pkgconfig/python3.pc python-3.4.pc
usr/lib/pkgconfig/python3.pc       python-3.4.pc
usr/share/man/man1/python3.1       python3.4.1
```

例 5 他のファイルシステムオブジェクトおよび属性の表示

次の例は、指定されたパッケージによってインストールされたリンクのパスおよびターゲットを示しています。pkg(5) のマニュアルページに示されている属性のほかに、いくつかの疑似属性を使用できます。疑似属性の一覧については、pkg(1) のマニュアルページを参照してください。

次の例で、pkg.name 疑似属性は、指定されたアクションを提供するパッケージの名前を示しています。この例では、Python 2.6.8 と Python 2.7.3 の両方がインストールされており、/usr/bin/python リンクに依存しない場合に特定バージョンにアクセスするために使用するパスがコマンドによって表示されます。複数バージョンの調整については、116 ページの「デフォルトのアプリケーション実装の指定」を参照してください。

```
$ pkg contents -t link -a path=usr/bin/python -o path,target,pkg.name
PATH            TARGET    PKG.NAME
usr/bin/python  python2.6 runtime/python-26
usr/bin/python  python2.7 runtime/python-27
```

グループパッケージ内のすべてのインストール可能なパッケージの一覧表示

Oracle Solaris には、いくつかのシステムインストールグループパッケージが用意されています。Oracle Solaris 11 GUI インストーラは solaris-desktop グループパッケージをインストールします。テキストインストーラおよび自動インストーラインストールのデフォルトの AI マニフェストは、solaris-large-server グループパッケージをインストールします。非大域ゾーン用のデフォルトのインストールマニフェストは、solaris-small-server グループパッケージをインストールします。solaris-minimal-server グループパッケージは、Oracle Solaris を実行するために必要な、サポートされる最小限のパッケージのセットをインストールします。

次のコマンドを使用して、指定されたグループパッケージに含まれる一連のパッケージを表示できます。

注記 - このパッケージリストには、グループパッケージのインストール時にインストールされるすべてのパッケージが含まれているわけではありません。このリスト内のパッケージの依存関係、およびそれらの依存関係の依存関係もインストールされます。

```
$ pkg contents -ro type,fmri -t depend solaris-minimal-server
TYPE    FMRI
```

```
group   network/ping
group   service/network/ssh-common
group   shell/tcsh
group   shell/zsh
group   system/network
require developer/debug/mdb
require editor/vim/vim-core
require group/system/solaris-core-platform
require package/pkg
require release/name
require release/notices
require shell/bash
require shell/ksh93
require system/core-os
require system/library/platform
```

`-t` オプションは、パッケージの `depend` アクションに一致します。`-o` オプションは、`depend` アクションの `type` および `fmri` 属性の値を表示します。各パッケージの名前とバージョンを表示するには、`pkg.shortfmri` を使用します。グループパッケージではファイルシステムの内容は指定されないことを思い出してください。グループパッケージは、グループに含まれるほかのパッケージを指定します。グループパッケージの詳細については、15 ページの「グループパッケージ」を参照してください。

各パッケージのサマリーも表示するには、`pkg list -s` コマンドを使用します。`pkg contents` コマンドの `-o fmri` オプションは、前の例の 2 番目の列に示されている出力を制限します。

```
$ pkg list -Has `pkg contents -Hro fmri -t depend solaris-minimal-server`
developer/debug/mdb                     Modular Debugger (MDB)
editor/vim/vim-core                     Vi IMproved (core executables)
group/system/solaris-core-platform      Oracle Solaris Core Platform
network/ping                            Ping command
package/pkg                             Image Packaging System
release/name                            Solaris Naming Enabler
release/notices                         Oracle Solaris notices
service/network/ssh-common              Secure Shell (SSH) service and configuration files
shell/bash                              GNU Bourne-Again shell (bash)
shell/ksh93                             Ksh93 - The AT&T Korn Shell
shell/tcsh                              Tenex C-shell (tcsh)
shell/zsh                               Z Shell (zsh)
system/core-os                          Core Solaris
system/library/platform                 Core Architecture, (Kvm)
system/network                          Core Network Infrastructure
```

ライセンス要件の表示

次の例は、パッケージライセンスに同意する必要があるすべてのパッケージを表示します。

```
$ pkg contents -rt license -a must-accept=true -o license,pkg.name '*'
LICENSE PKG.NAME
BCL     developer/java/jdk-7
BCL     runtime/java/jre-7
LICENSE developer/java/jdk-6
LICENSE library/java/java-demo-6
```

```
LICENSE runtime/java/jre-6
lic_OTN consolidation/osnet/osnet-incorporation
lic_OTN install-image/solaris-auto-install
```

これらのパッケージをインストールまたは更新するには、`--accept` オプションを指定しなければならない場合があります。

ライセンステキストを表示するには、28 ページの「パッケージライセンスの表示」 に記載されているように、次の例のようなコマンドを使用します。複数の FMRI を一覧表示できます。

```
$ pkg info -r --license runtime/java/jre-7 osnet-incorporation
```

パッケージの検索

指定したパターンにデータが一致するパッケージを検索するには、`pkg search` コマンドを使用します。

pkg search コマンドと pkg contents コマンドの比較

`pkg search` コマンドは、`pkg contents` コマンドと同じようにパッケージの内容を調べます。`pkg contents` コマンドは内容を返しますが、`pkg search` コマンドは検索クエリーに一致するパッケージの名前を返します。次の表に、これら 2 つのコマンドについてのいくつかの類似点と相違点を示します。

`pkg contents`	■ インストール済みパッケージを調べます。このイメージに対して構成されているすべてのパブリッシャーに関連付けられたリポジトリ内のパッケージを調べる場合は、`-r` オプションを使用します。 ■ 調べるリポジトリの URI を指定する場合は、`-g` オプションを使用します。 ■ アクションを指定する場合は、`-t` オプションを使用します。 ■ 属性および属性値を指定する場合は、`-a` オプションを使用します。 ■ 結果の列を指定する場合は、`-o` オプションを使用します。 ■ 結果をソートする場合は、`-s` オプションを使用します。
`pkg search`	■ このイメージに対して構成されているすべてのパブリッシャーに関連付けられたリポジトリ内でパッケージを検索します。インストール済みパッケージのみ検索する場合は、`-l` オプションを使用します。 ■ 検索するリポジトリの URI を指定する場合は、`-s` オプションを使用します。 ■ アクションを指定する場合は、検索クエリーを使用します。 ■ 属性および属性値を指定する場合は、検索クエリーを使用します。 ■ 結果の列を指定する場合は、`-o` オプションを使用します。

ヒント - 指定したパッケージの内容を表示するには、`pkg contents` コマンドを使用し、クエリーに一致するパッケージを表示するには、`pkg search` コマンドを使用します。関心のある内容を提供するパッケージがわかっている場合は、`pkg contents` コマンドを使用します。

検索クエリーの指定

デフォルトでは、検索クエリーは、大文字と小文字を除いて完全一致する一連の用語です。大文字と小文字を区別する検索を指定する場合は、`-I` オプションを使用します。

クエリー用語には、`?` と `*` のワイルドカードを使用できます。フレーズを検索するには、単一引用符または二重引用符を使用します。ワイルドカードや引用符を使用するときは、使用しているシェルを必ず考慮してください。

複数のクエリー用語を指定できます。デフォルトで、複数の用語は AND で結合されます。OR で 2 つの用語を明示的に結合できます。

検索クエリーは次の構造化された形式で表現できます。

package : *action* : *index* : *token*

package	検索するパッケージの名前または複数のパッケージに一致するパターン。
action	`pkg(5)` のマニュアルページのアクションに関するセクションに一覧表示されているアクションの名前。
index	`pkg`(5) のマニュアルページのアクションに関するセクションに一覧表示されている *action* の属性の名前。
token	*index* の値または *index* の値に一致するパターン。

欠けているフィールドは、暗黙的にワイルドカード化されます。*token* 文字列は明示的にワイルドカード化できます。アクションおよびインデックスの名前を明示的にワイルドカード化することはできません。

すべての属性が検索可能なわけではありません。たとえば、`mode` は `file` アクションの属性ですが、`mode` は *index* の有効な値ではありません。

index の一部の値は、ほかの属性から派生した値です。たとえば、*index* に `basename` を指定できますが、これは `file` または `dir` アクションの `path` 属性の最後のコンポーネントです。*index* の便利な値には、`file` および `dir` アクションの `basename` と

path、depend アクションの依存関係タイプ (require や group など)、driver アクションの driver_name と alias などがあります。

token の値は、*index* で指定された属性の値と比較されます。たとえば、次に示す部分的な driver アクションの場合、*index* には属性名 alias を指定し、*token* には pci108e* を指定できます。

```
driver alias=pci108e,1647 alias=pci108e,16a7
```

set アクションの構文は少し違います。set アクションの 2 つの属性は、name と value です。この場合、*index* の値は name 属性の値であり、*token* の値は一致する value 属性の値と比較されます。次の例は、ドライバパッケージに対する set アクションを示しています。

```
set name=pkg.summary value="Broadcom NetXtreme II 10GbE NIC Driver"
```

次の例は、*action* に set を、*index* に pkg.summary を、*token* に Broadcom を指定します。search.match および pkg.name 列指定子は疑似属性です。pkg(1) のマニュアルページを参照してください。

```
$ pkg search -o search.match,pkg.name pkg.summary:Broadcom
SEARCH.MATCH                          PKG.NAME
Broadcom NetXtreme II 10GbE NIC Driver driver/network/ethernet/bnxe
Broadcom 57xx 1GbE NIC Driver         driver/network/ethernet/bge
Broadcom NetXtreme II 1GbE NIC Driver  driver/network/ethernet/bnx
Broadcom BCM4401 NIC Driver           driver/network/ethernet/bfe
Broadcom HT1000 SATA driver           driver/storage/bcm_sata
```

set アクションの name 属性の明確に定義された値には、pkg.fmri、info.classification、pkg.description、pkg.summary などがあります。pkg(5) のマニュアルページの設定アクションに関する項目を参照してください。

デフォルトでは、現在インストールされているパッケージバージョン以上の一致のみが表示されます。一致したすべてのバージョンを表示するには、-f オプションを使用します。

デフォルトでは、一致するすべてのアクションについて結果が表示されるため、1 つのパッケージに複数行の結果が生成されることがあります。一致する各パッケージを 1 回だけ表示するには、-p オプションを使用します。

指定されたファイルを提供するパッケージの識別

次の例は、libpower ライブラリが system/kernel/power パッケージから取得されたことを示しています。

```
$ pkg search -Hlo pkg.name /lib/libpower.so.1
system/kernel/power
$ pkg search -lo path,pkg.name libpower.so.1
PATH              PKG.NAME
lib/libpower.so.1 system/kernel/power
```

```
$ pkg search -Hlo path,pkg.name basename:libpower.so.1
lib/libpower.so.1 system/kernel/power
$ pkg search -Hlo path,pkg.name 'path:*libpower.so.1'
lib/libpower.so.1 system/kernel/power
```

指定した SMF サービスを提供するパッケージの識別

特定の SMF サービスがどのパッケージで提供されているかを表示するには、サービスの名前を `org.opensolaris.smf.fmri` 属性の値として検索します。

```
$ pkg search -o pkg.name,search.match 'org.opensolaris.smf.fmri:*network/http*'
PKG.NAME                SEARCH.MATCH
web/java-servlet/tomcat svc:/network/http
web/proxy/squid         svc:/network/http
web/proxy/privoxy       svc:/network/http
web/server/lighttpd-14  svc:/network/http
web/server/apache-22    svc:/network/http
web/server/apache-22    svc:/network/http:apache22
web/server/lighttpd-14  svc:/network/http:lighttpd14
web/proxy/privoxy       svc:/network/http:privoxy
web/proxy/squid         svc:/network/http:squid
web/java-servlet/tomcat svc:/network/http:tomcat6
```

この場合、各属性は、サービスインスタンス名が指定されたサービス名とサービスインスタンス名が指定されていないサービス名の、2 つの値を持ちます。次の例は、この属性がパッケージマニフェストでどのように指定されるかを示しています。

```
set name=org.opensolaris.smf.fmri value=svc:/network/http value=svc:/network/http:apache22
```

次の例は、これと同じ情報を、各パッケージが 1 回だけ示されるようにして表示しています。`-p` オプションは、`search.match` などのアクションレベルの出力を要求する場合は使用できません。コロン文字をエスケープすることで、コロン文字は別の検索クエリーフィールドでなく *token* の一部として解釈されます。

```
$ pkg search -o pkg.name,search.match 'org.opensolaris.smf.fmri:*network/http\:*'
PKG.NAME                SEARCH.MATCH
web/server/apache-22    svc:/network/http:apache22
web/server/lighttpd-14  svc:/network/http:lighttpd14
web/proxy/privoxy       svc:/network/http:privoxy
web/proxy/squid         svc:/network/http:squid
web/java-servlet/tomcat svc:/network/http:tomcat6
```

指定されたユーザーを提供するパッケージの識別

IPS パッケージは、使用するデーモンまたはその他のソフトウェアのユーザーを提供します。

次のコマンドは、インストールされているパッケージによって提供されるユーザーを示しています。

```
$ pkg search -lo action.key user::
```

次のコマンドは、特定のユーザー定義を提供するパッケージを示しています。

```
$ pkg search -o action.key,pkg.name user::nova
ACTION.KEY PKG.NAME
nova       cloud/openstack/nova
```

次のコマンドは、OR キーワードの使用例を示しています。

```
$ pkg search -o action.key,pkg.name user::nova OR user::neutron
ACTION.KEY PKG.NAME
nova       cloud/openstack/nova
neutron    cloud/openstack/neutron
```

分類またはカテゴリ別のパッケージの一覧表示

次の例は、`info.classification` 属性の値に「Source Code Management」を持つ、すべてのインストール済みパッケージを識別します。

```
$ pkg search -Hlo pkg.shortfmri info.classification:'source code management'
pkg:/developer/versioning/sccs@0.5.11-0.175.2.0.0.8.0
pkg:/developer/versioning/git@1.7.9.2-0.175.2.0.0.34.0
pkg:/developer/versioning/mercurial-27@2.2.1-0.175.2.0.0.34.0
```

次の例は、この一致検索に使用されるパッケージメタデータを示しています。

```
set name=info.classification value="org.opensolaris.category.2008:Development/Source Code
 Management"
```

この情報は、`pkg info` コマンドからの出力の「Category」行に表示されます。

```
$ pkg info mercurial-27
...
      Category: Development/Source Code Management
...
```

検索可能なほかの分類については、「Classification Values」 in 『Packaging and Delivering Software With the Image Packaging System in Oracle Solaris 11.3』を参照してください。

また、次の例に示すように、パッケージ名のコンポーネントの 1 つを推測するように `pkg list` コマンドを使用することもできます。

```
$ pkg list '*storage*'
$ pkg list -a '*database*'
```

依存パッケージの表示

次の例は、指定したパッケージの依存関係であるパッケージを示しています。

次の例は、system/kernel/power パッケージに require 依存関係のあるパッケージを示しています。

```
$ pkg search -Hlo pkg.name require:system/kernel/power
system/kernel/dynamic-reconfiguration/i86pc
system/hal
```

次の pkg contents コマンドは、検索の結果を確認します。要求された出力 action.raw は、パッケージマニフェストに示されるものとまったく同じアクションを表示する疑似属性です。

```
$ pkg contents -rt depend -a fmri='*power*' -o pkg.name,action.raw i86pc system/hal
PKG.NAME                                        ACTION.RAW
system/hal                                      depend fmri=pkg:/system/kernel/power
@0.5.11-0.175.2.0.0.34.0 type=require variant.opensolaris.zone=global
system/kernel/dynamic-reconfiguration/i86pc depend fmri=pkg:/system/kernel/power
 type=require
```

次の例は、多くのパッケージで pkg:/x11/server/xorg@1.14.99 への exclude 依存関係があることを示しています。

```
$ pkg search -lo pkg.name,fmri 'depend:exclude:*xorg*'
PKG.NAME                                        FMRI
x11/server/xorg/driver/xorg-video-ati           pkg:/x11/server/xorg@1.14.99
x11/server/xvnc                                 pkg:/x11/server/xorg@1.14.99
x11/server/xserver-common                       pkg:/x11/server/xorg@1.14.99
x11/server/xorg/driver/xorg-input-vmmouse       pkg:/x11/server/xorg@1.14.99
x11/server/xephyr                               pkg:/x11/server/xorg@1.14.99
...
```

グループパッケージ内のすべてのパッケージの一覧表示

Oracle Solaris 11 GUI インストーラは solaris-desktop グループパッケージをインストールします。テキストインストーラおよび自動インストーラインストールのデフォルトの AI マニフェストは、solaris-large-server グループパッケージをインストールします。非大域ゾーン用のデフォルトのインストールマニフェストは、solaris-small-server グループパッケージをインストールします。solaris-minimal-server グループパッケージは、Oracle Solaris を実行するために必要な、サポートされる最小限のパッケージのセットをインストールします。

次の検索書式を使用して、指定されたグループパッケージに含まれている一連のパッケージを表示できます。

注記 - このパッケージリストには、グループパッケージのインストール時にインストールされるすべてのパッケージが含まれているわけではありません。このリスト内のパッケージの依存関係、およびそれらの依存関係の依存関係もインストールされます。

```
$ pkg search -o type,fmri '*/solaris-minimal-server:depend::'
TYPE    FMRI
require developer/debug/mdb
require editor/vim/vim-core
require group/system/solaris-core-platform
group   network/ping
require package/pkg
require release/name
require release/notices
group   service/network/ssh-common
require shell/bash
require shell/ksh93
group   shell/tcsh
group   shell/zsh
require system/core-os
require system/library/platform
group   system/network
```

depend クエリーフィールドは、パッケージの depend アクションに一致します。-o オプションは、depend アクションの type および fmri 属性の値を表示します。各パッケージの名前とバージョンを表示するには、pkg.shortfmri を使用します。グループパッケージではファイルシステムの内容は指定されないことを思い出してください。グループパッケージは、グループに含まれるほかのパッケージを指定します。グループパッケージの詳細については、15 ページの「グループパッケージ」を参照してください。

pkg search コマンドは、指定したパッケージのアクションの属性の値を返しています。この例では、その属性値はパッケージ名になります。このコマンドの結果の数は、32 ページの「グループパッケージ内のすべてのインストール可能なパッケージの一覧表示」で示す、類似の pkg contents コマンドの結果の数より多くなることがあり、この理由は、これらの検索結果に、インストール可能なパッケージだけでなく、指定したパッケージの group タイプの depend アクションで名前が付けられているすべてのパッケージの名前が含まれるためです。たとえば、このイメージにインストールできないパッケージバリアントおよびファセットが含まれることがあります。この相違を確認するには、solaris-large-server パッケージを使用して両方の例を試してみてください。

♦ ♦ ♦ 第 3 章

3 ソフトウェアパッケージのインストールおよび更新

パッケージのインストールと更新は、一部のパッケージを特定のバージョンに制限する、パブリッシャーの検索順序を構成する、パッケージの署名プロパティーを設定するなどのイメージの構成によって影響を受けます。イメージの構成については、第5章「インストールされるイメージの構成」で説明しています。

すでにインストールされているパッケージ、インストールに使用できるパッケージ、および使用可能な更新のあるパッケージを判断する方法については、第2章「ソフトウェアパッケージに関する情報の取得」で取り上げています。

この章では、次のタスクの実行方法を示します。

- 試験的なインストールを実行して、インストールが成功するかどうか、何がインストールされるかを確認する
- パッケージをインストール、更新、およびアンインストールする
- パッケージを検証する
- インストールされたパッケージの問題を修正する
- インストールされたファイルをその元の内容に復元する
- パッケージをアンインストールする

64 ページの「非大域ゾーンの操作」では、Oracle Solaris ゾーンに固有のパッケージ操作のさまざまな側面について説明します。

パッケージのインストール、更新、およびアンインストールには強力な権限が必要です。詳細は、21 ページの「インストール権限」を参照してください。

この章で説明されているコマンドのすべてのオプションの完全なリストについては、pkg(1) のマニュアルページを参照してください。

操作のプレビュー

この章と第5章「インストールされるイメージの構成」で示す多くのコマンドには -n オプションがあり、変更を加えることなくコマンドが何を実行するかを確認できます。

ヒント - 使用可能な場合は常に -n オプションを使用することをお勧めします。-n オプションと 1 つまたは複数の詳細オプション (-nv、-nvv) を使用して、コマンドの効果を確認してから、-n オプションを付けずにコマンドを実行します。

次の例は、実際には実行しないパッケージインストールに関する情報を表示します。

```
$ pkg install -nv group/feature/amp
           Packages to install:         6
           Mediators to change:         1
            Services to change:         2
     Estimated space available:  22.70 GB
Estimated space to be consumed: 751.08 MB
       Create boot environment:        No
Create backup boot environment:        No
          Rebuild boot archive:        No

Changed mediators:
  mediator mysql:
           version: None -> 5.1 (system default)

Changed packages:
solaris
  database/mysql-51
    None -> 5.1.37,5.11-0.175.2.0.0.34.0:20140303T160611Z
  database/mysql-common
    None -> 5.11,5.11-0.175.2.0.0.34.0:20140303T161628Z
  group/feature/amp
    None -> 0.5.11,5.11-0.175.2.0.0.33.0:20140217T134747Z
  web/server/apache-22/module/apache-dtrace
    None -> 0.3.1,5.11-0.175.2.0.0.34.0:20140303T175456Z
  web/server/apache-22/module/apache-fcgid
    None -> 2.3.9,5.11-0.175.2.0.0.34.0:20140303T175502Z
  web/server/apache-22/module/apache-php5
    None -> 5.2.17,5.11-0.175.1.0.0.18:20120611T210317Z
Services:
  restart_fmri:
    svc:/system/manifest-import:default
    svc:/system/rbac:default
```

この出力は、このインストール操作は新規 BE でなく現在の BE で実行されること、およびこの現在の BE のバックアップは作成されないことを示しています。新規 BE またはバックアップ BE を求めるオプションまたはイメージプロパティーを指定することもできます。「Changed packages」セクションには、amp グループパッケージがインストールされ、その 5 つのグループ依存関係がインストールされることが示されています。出力には、インストールされる各パッケージのバージョンが示されています。トークン None は、これらのパッケージが現在インストールされておらず、したがって更新されないことを示しています。

次のコマンドは、多数のパッケージが影響を受けるため、大量の出力が生成されます。このファセットを設定すると、すべてのパッケージについてのローカライズされたすべての内容がインストールされます。このプレビューコマンドを実行することで、この操作のスケジュールを決定する方法や、追加する新規ロケールを少なくするかどうかが変更される場合もあります。この出力は、新しい BE はデフォルトで作成されませんが、バックアップ BE が作成されることを示しています。

```
$ pkg change-facet -nv 'facet.locale.*=true'
            Packages to change:       130
     Variants/Facets to change:         1
     Estimated space available:  22.70 GB
Estimated space to be consumed:   3.45 GB
       Create boot environment:        No
Create backup boot environment:       Yes
          Rebuild boot archive:        No
Changed variants/facets:
    facet locale.* (local): False -> True
Changed packages:
solaris
  ...
```

パッケージのインストールおよび更新

次の表は、`pkg install` と `pkg update` コマンドの類似点と相違点を示します。

`pkg install`	■ オペランドとして 1 つ以上のパッケージ名が必要です。 ■ 現在インストールされていないパッケージをインストールします。 ■ すでにインストールされているパッケージを更新します。 ■ パッケージのダウングレードは行いません。指定したインストール済みパッケージのバージョンの方が低いと、システムはそのパッケージをインストールしません。
`pkg update`	■ すでにインストールされているパッケージの名前をオペランドとして 0 個以上指定します。 ■ インストール済みパッケージを更新します。 ■ パッケージ名を指定しないか `'*'` を指定すると、イメージにインストールされているすべてのパッケージが更新されます。 ■ インストール済みパッケージを、FMRI に指定されているバージョンにダウングレードします。 ■ まだインストールされていないパッケージはインストールしません。まだインストールされていないパッケージを指定すると、システムはそのパッケージをインストールしません。

インストールおよび更新時に、`preserve` 属性と `overlay` 属性を持つファイルがどのように処理されるかについては、pkg(5) のマニュアルページの `file` アクションのこれらの属性を参照してください。

パッケージのインストールまたは更新後に、インストールされているパッケージを検証します。55 ページの「パッケージの検証と検証エラーの修正」を参照してください。

一般的なインストールオプション

このセクションでは、複数のインストール関連のコマンドに共通するオプションについて説明します。メディエータの設定または設定解除、バリアントまたはファセットの変更、パッケージの修正、およびファイルを元に戻す操作は、パッケージのインストール、更新、またはアンインストールも伴うことがあります。

ブート環境オプション

パッケージをインストール、更新、またはアンインストールした場合に、新しい BE またはバックアップ BE が自動的に作成されることがあります。BE に関するイメージポリシーの制約内で、下に示すオプションを使用して、新しい BE とバックアップ BE の作成を制御できます。新しい BE とバックアップ BE について、および BE に関するイメージポリシーの設定方法については、122 ページの「ブート環境ポリシーイメージのプロパティー」を参照してください。

次の BE オプションを使用して、新しい BE またはバックアップ BE を作成するかどうかを強制したり、BE にカスタム名を指定したり、新しい BE をアクティブにしないように指定したりします。これらのオプションは、`install`、`exact-install`、`uninstall`、`update`、`revert`、`set-mediator`、`unset-mediator`、`change-variant`、および `change-facet` サブコマンドで使用できます。

`--no-be-activate`

BE が作成される場合に、それを次回ブート時にアクティブ BE として設定しません。

コマンド出力で、新しいブート環境が作成されたことを示すメッセージに注意してください。新しいブート環境を作成してアクティブにすると、`--no-be-activate` オプションを指定しない場合、その BE が次回のリブート時にデフォルトでブートされます。

アクティブ BE を表示および変更するには、`beadm(1M)` コマンドを使用します。

`--no-backup-be`

バックアップ BE を作成しません。

`--require-backup-be`

新しい BE が作成されない場合に、バックアップ BE を作成します。このオプションを指定しないと、イメージポリシーに基づいてバックアップ BE が作成されます。バックアップ BE が自動的に作成される状況の説明については、122 ページの「ブート環境ポリシーイメージのプロパティー」を参照してください。

`--backup-be-name` *name*

バックアップ BE が作成される場合に、デフォルトの名前ではなく、*name* という名前を付けます。`--backup-be-name` を使用すると `--require-backup-be` が暗黙的に指定されます。

`--deny-new-be`

新しい BE を作成しません。新しい BE が必要な場合、インストール、更新、アンインストール、または元に戻す操作は実行されません。

`--require-new-be`

新しい BE を作成します。このオプションを指定しないと、イメージポリシーに基づいて BE が作成されます。BE が自動的に作成される状況の説明については、122 ページの「ブート環境ポリシーイメージのプロパティー」を参照してください。このオプションを `--require-backup-be` と組み合わせることはできません。

`--be-name` *name*

BE が作成される場合に、デフォルトの名前ではなく、*name* という名前を付けます。`--be-name` を使用すると `--require-new-be` が暗黙的に指定されます。このオプションを使用することは、操作を行うためのもっとも安全な方法です

非大域ゾーンで動作するオプション

64 ページの「非大域ゾーンの操作」で説明されているように、大域ゾーンで実行されるパッケージのインストール、除去、および更新の一部のみが、非大域ゾーンに自動的に影響します。`-r` オプションは、大域ゾーンに入力した `pkg` 操作と同じ操作を非大域ゾーンで実行し、`-r` を使用しなかった場合に影響を及ぼすよりもさらに多くのパッケージに影響を及ぼすことになります。これらのオプションは、`install`、`uninstall`、`update`、`change-variant`、および `change-facet` サブコマンドで使用できます。

`-r`

この操作を大域ゾーンで実行し、インストール済みの `solaris` ブランドのすべての非大域ゾーンでも実行します。非大域ゾーンに対する影響は、各非大域ゾーンにログインしてコマンドを直接実行することと似ています。

このオプションを指定せずに pkg コマンドを大域ゾーンで実行すると、64 ページの「非大域ゾーンの操作」に記載されているように、非大域ゾーンは大域ゾーンとの互換性を維持するために必要な範囲で変更されます。このオプションを指定すると、pkg 操作は -z および -Z オプションによって制限される場合を除き、インストール済みのすべての非大域ゾーンに適用されます。-z オプションおよび -Z オプションによって除外されたゾーンは、大域ゾーンとの同期を維持するための更新が必要であれば、変更されることもあります。

`-z` *zone*

指定された非大域ゾーンでのみこの操作を実行します。-z オプションは複数回指定できます。-z オプションは、-r オプションと同時に指定した場合のみ使用できます。-z オプションは -Z オプションと一緒に使用できません。

`-Z` *zone*

指定されたゾーンを除くすべての非大域ゾーンでこの操作を実行します。-Z オプションは複数回指定できます。-Z オプションは、-r オプションと同時に指定した場合のみ使用できます。-Z オプションは -z オプションと一緒に使用できません。

次のオプションは、大域ゾーンと同時に更新する非大域ゾーンの数を指定します。このオプションは、install、exact-install、uninstall、update、change-variant、および change-facet サブコマンドで使用できます。

`-C` *n*

n 個までのインストール済みの solaris ブランドの非大域ゾーンを大域ゾーンと並行して更新します。*n* が 0 または負の数値の場合、すべての非大域ゾーンが大域ゾーンと同時に更新されます。

環境変数 PKG_CONCURRENCY も値 *n* に設定できます。-C オプションは PKG_CONCURRENCY 設定をオーバーライドします。-C オプションが指定された場合、PKG_CONCURRENCY は無視されます。

サービスアクションのオプション

パッケージがインストールまたは更新されるとき、指定されたサービスの再開またはリフレッシュなどの SMF サービスアクションがパッケージに指定されることがあります。多数のパッケージを操作している場合、すべてのサービスアクションが終了する前に pkg 操作が終了することがあります。このとき、関連付けられたサービスがまだ利用できないという理由で、新しくインストールされたソフトウェアを使用できない場合があります。

この問題を回避するには、次のいずれかのオプションを使用して、SMF アクチュエータを pkg コマンドと同期させて実行します。これらのオプション

は、install、uninstall、update、change-variant、および change-facet サブコマンドで使用できます。

--sync-actuators

このオプションを指定した場合、pkg コマンドは、pkg が呼び出されたゾーン (大域ゾーンまたは非大域ゾーン) 内ですべての SMF アクチュエータが完了するまで返されません。

--sync-actuators-timeout *timeout*

このオプションを指定した場合、pkg コマンドは、すべての SMF アクチュエータが完了するか、*timeout* 期間が終了するか、いずれか短い方が終了するまで返されません。指定された *timeout* (秒単位) までにアクチュエータが完了しない場合、pkg コマンドは動作を続行し、リターンコード 8 で終了します。

ライセンスオプション

パッケージをインストールまたは更新する前に、ライセンスを受諾することが必要な場合があります。必要なライセンスを表示して受諾するには、次のオプションを使用します。これらのオプションは、install、exact-install、update、fix、change-variant、および change-facet サブコマンドで使用できます。

--licenses

この操作の一環としてインストールまたは更新されるパッケージのすべてのライセンスを表示するには、--licenses オプションを使用します。この操作を続行するために受諾する必要があるライセンスだけでなく、すべてのパッケージのライセンスが表示されます。続行するために受諾する必要があるライセンスは、--licenses オプションを指定しなくても表示されます。ほかの操作を開始せずにパッケージのライセンスを表示するには、28 ページの「パッケージライセンスの表示」に記載されているように pkg info コマンドを使用します。同意する必要があるライセンスを一覧表示するには、33 ページの「ライセンス要件の表示」に記載されているように pkg contents コマンドを使用します。

--accept

更新またはインストールされるパッケージのライセンス条項に同意することを示す場合は、--accept オプションを使用します。このオプションを指定せず、パッケージのライセンスに同意が必要になった場合、必要なライセンスが表示されてインストール操作は失敗します。

その他のインストールオプション

`--no-index`

デフォルトでは、パッケージをインストール、更新、またはアンインストールしたときに検索インデックスが更新されます。これらの操作が正しく終了したあとに検索インデックスを更新しない場合は、`--no-index` オプションを使用します。このオプションを指定すると、多数のパッケージをインストールする場合に、いくらか時間を節約できます。すべてのインストール、更新、およびアンインストール操作を完了したら、`pkg refresh` を使用して、使用可能なパッケージのリストと、指定された各パブリッシャーのパブリッシャーメタデータを更新できます。パブリッシャーを指定しない場合、すべてのパブリッシャーを対象にリフレッシュが実行されます。このオプションは、`install`、`exact-install`、`uninstall`、および `update` サブコマンドで使用できます。

`--no-refresh`

`--no-refresh` オプションを指定した場合、使用可能なパッケージやその他のメタデータの最新のリストを取得するために、イメージのパブリッシャーのリポジトリに接続しません。このオプションは、`install`、`exact-install`、および `update` サブコマンドで使用できます。

新しいパッケージのインストール

デフォルトで、パッケージを提供したパブリッシャーの検索順序で先頭のパブリッシャーから、イメージの残りの部分と互換性のあるパッケージの最新バージョンがインストールされます。最新バージョンを明示的に要求するには、パッケージ FMRI のバージョン部分に `latest` を使用します。

パッケージがすでにインストールされている場合は、現在インストールされているバージョンを提供したパブリッシャーから、残りのイメージと互換性のある最新バージョンのパッケージをインストールすることによってパッケージが更新されます。

複数のパッケージが指定され、かつ指定されたパッケージのいずれかをこのイメージにインストールできない場合、指定されたパッケージは一切インストールされません。

パッケージが回避リストにある場合は、インストールすると回避リストから削除されます。回避リストについては、119 ページの「グループパッケージに含まれる一部のパッケージのインストールの回避」を参照してください。

インストール可能なパッケージの識別と指定

イメージで複数のパブリッシャーが有効になっている場合は、パブリッシャーのスティッキネスと検索順序を設定したり、またはパッケージ FMRI でパブリッシャーを指定したりすることによって、パッケージを提供するパブリッシャーを制御できます。また、インストールするバージョンをパッケージ FMRI で指定することもできます。パッケージ FMRI の説明については、16 ページの「障害管理リソース識別子」を参照してください。パブリッシャーのスティッキネスと検索順序の設定については、99 ページの「パブリッシャーの構成」を参照してください。

パッケージ名がパブリッシャーを指定しない場合、一致するパッケージを提供する最初のパブリッシャーがインストール元として使用されます。該当するパブリッシャーがこのイメージ内にインストール可能なパッケージのバージョンを提供しない場合、インストール操作は失敗します。このイメージにインストール可能なパッケージのバージョンを提供するパブリッシャーを確認するには、`pkg list -a` コマンドを使用します。

次のコマンドは、`atool` のインストール可能なバージョンが構成済みパブリッシャーから利用可能であるが、検索順序で最初のパブリッシャーには、このイメージにインストールできないバージョンがあることを示しています。`pkg list` コマンドのオプションについては、23 ページの「パッケージのインストール状態情報の表示」を参照してください。

```
$ pkg list -a atool
NAME (PUBLISHER)     VERSION    IFO
atool (isvpub)       2.0        ---
$ pkg list -af atool
NAME (PUBLISHER)     VERSION    IFO
atool                1.1        ---
atool (isvpub)       2.0        ---
```

この場合、次のインストールコマンドは失敗します。パッケージシステムでは、検索順序の最初にあるパブリッシャーからパッケージ名 `atool` に一致するものが検索されますが、そのパッケージはインストールできません。

```
$ pkg install atool
```

このパッケージをインストールするには、次の例に示すように、より具体的なパッケージ名を指定します。

```
$ pkg install //isvpub/atool
$ pkg install atool@2.0
```

実際にインストールを実行する前に、インストールされるものを確認するには、`-nv` オプションを使用します。エラーメッセージを受け取る場合、付録A パッケージのインストールおよび更新のトラブルシューティングを参照してください。

パッケージのソースの指定

指定したパッケージリポジトリまたはパッケージアーカイブを、パッケージデータの取得元になるイメージ内のソースのリストに一時的に追加するには、-g オプションを使用します。クライアント SSL 証明書が必要なリポジトリは、このオプションとともに使用できません。このオプションは、子イメージ (非大域ゾーン) を持つイメージでは使用できません。このイメージに非大域ゾーンがインストールされている場合は、pkg set-publisher コマンドを使用してこのパブリッシャーと起点を追加します。このオプションは複数回指定できます。

-g オプションを指定すると、パッケージの取得時に、イメージで有効になっているパブリッシャーが優先されます。

- 指定されたパッケージ名またはパッケージ名パターンに一致するパッケージが、イメージ内で有効になっているパブリッシャーから利用可能な場合に、その同じパブリッシャーが -g オプションで指定された場所に見つからないとき、パッケージシステムは、イメージ内で有効になっているパブリッシャーからパッケージをインストールしようとします。install または update のあとに、イメージ内で構成されていないパブリッシャーによって提供されるすべてのパッケージは、起点なしでイメージ構成に追加されます。
- 指定されたパッケージ名またはパッケージ名パターンに一致するパッケージが、イメージ内で有効になっているパブリッシャーから利用可能な場合に、その同じパブリッシャーが -g オプションで指定された場所にパッケージを発行しているとき、パッケージシステムは、-g オプションで指定された場所からパッケージをインストールしようとします。

次の例では、イメージ内に構成されている solaris パブリッシャーから btool パッケージが利用可能です。btool パッケージは、リポジトリの起点 http://pkg.example1.com/ の devtool パブリッシャーからも利用可能ですが、devtool パブリッシャーはイメージ内に構成されていません。イメージ内に構成されているパブリッシャーからパッケージが利用可能な場合は、構成済みのパブリッシャーが -g のソースより優先されるため、コマンドは solaris パブリッシャーからパッケージをインストールしようとします。

```
$ pkg install -g http://pkg.example1.com/ btool
```

devtool パブリッシャーからパッケージをインストールするには、パッケージ名にパブリッシャーの名前を指定します。

```
$ pkg install -g http://pkg.example1.com/ //devtool/btool
```

次の例で、isvpub は起点 /var/share/pkgrepos/isvrepo でイメージ内に構成されているパブリッシャーです。isvpub パブリッシャーは http://pkg.example2.com/ にあるリポジトリにもパッケージを発行していますが、イメージ内で構成されているこのパブリッシャーにその起点は指定されていません。同じパブリッシャーが両方の

場所にパッケージを提供しているため、次のコマンドは `http://pkg.example2.com/` の場所からパッケージをインストールしようとします。

```
$ pkg install -g http://pkg.example2.com/ atool
```

101 ページの「パッケージパブリッシャーの追加、変更、削除」にあるパブリッシャーのスティッキネスの説明も参照してください。

新しいブート環境へのパッケージのインストール

ヒント - 新しい BE を明示的に指定することは、インストールまたは更新のもっとも安全な方法です。BE が作成される状況については、122 ページの「ブート環境ポリシーイメージのプロパティー」を参照してください。

新しい BE は指定したインストール、アンインストール、または更新の変更が適用された現在の BE のクローンになります。現在の BE は変更されません。システムは自動的に再起動されません。システムの次回の再ブート時に、新しい BE がデフォルトのブート選択肢になります。現在の BE も引き続きブートに使用できます。

`--no-be-activate` オプションを指定した場合、次回のリブート時に、新しい BE がデフォルトのブート選択肢になりません。

新規 BE の作成を強制するか、新規 BE がデフォルトで作成される場合に新規 BE にわかりやすい名前を付けるには、`--be-name` オプションを使用します。

42 ページの「操作のプレビュー」の例では、`group/feature/amp` パッケージをインストールするとき、新規 BE がデフォルトで作成されないことを示しました。次の出力の一部では、`--be-name` オプションが指定されているため、新規 BE が作成されます。

```
$ pkg install -v --be-name s11amp group/feature/amp
           Packages to install:         6
           Mediators to change:         1
     Estimated space available:  22.70 GB
Estimated space to be consumed: 751.08 MB
       Create boot environment:       Yes
     Activate boot environment:       Yes
Create backup boot environment:        No
          Rebuild boot archive:        No
```

インストール操作の終わりに次のメッセージが表示されます。

```
A clone of s11 exists and has been updated and activated.
On the next boot the Boot Environment s11amp will be
mounted on '/'.  Reboot when ready to switch to this updated BE.
```

`pkg list` コマンドは、現在の BE に `group/feature/amp` パッケージがインストールされていないため、`group/feature/amp` パッケージがインストールされていないこと

を報告します。group/feature/amp パッケージは新しい s11amp BE にインストールされます。

```
$ pkg list group/feature/amp
pkg list: no packages matching 'group/feature/amp' installed
```

beadm list コマンドを使用して、システムに s11amp という新しいアクティブな BE があることを確認します。「N」で示された BE は現在ブートされています。「R」で示された BE はリブート時のデフォルトです。リブート時のデフォルトの BE を変更するには、beadm activate コマンドを使用します。

```
$ beadm list
BE            Active Mountpoint Space   Policy Created
--            ------ ---------- -----   ------ -------
s11           N      /          30.92M  static 2014-03-05 08:51
s11amp        R      -          25.75G  static 2014-03-26 10:45
```

新しい BE に group/feature/amp パッケージがインストールされていることを確認します。新規 BE をマウントし、-R オプションを使用して、マウントされた BE 上で操作を行います。I 列の「i」は、group/feature/amp パッケージがインストールされていることを示します。

```
$ beadm mount s11amp /mnt
$ beadm list
BE            Active Mountpoint Space   Policy Created
--            ------ ---------- -----   ------ -------
s11           N      /          30.92M  static 2014-03-05 08:51
s11amp        R      /mnt       25.75G  static 2014-03-26 10:45
$ pkg -R /mnt list group/feature/amp
NAME (PUBLISHER)       VERSION                    IFO
group/feature/amp      0.5.11-0.175.2.0.0.33.0    i--
```

s11amp BE を必ずアンマウントしてください。

```
$ beadm unmount s11amp
```

パッケージの拒否

指定したパッケージがインストールされないようにするには、pkg install コマンドの --reject オプションを使用します。一致するパッケージがすでにインストールされている場合、それらはこの操作の一環として削除されます。

グループ依存関係である拒否されたパッケージは、回避リストに配置されます。回避リストについては、119 ページの「グループパッケージに含まれる一部のパッケージのインストールの回避」を参照してください。

次のコマンド例は、developer-gnu パッケージと、その cvs 依存関係以外のすべての依存関係をインストールします。

```
$ pkg install --reject developer/versioning/cvs group/feature/developer-gnu
```

パッケージの更新

`install` または `update` サブコマンドを使用して、インストールされているパッケージを、現在インストールされているバージョンを提供したパブリッシャーから、残りのイメージと互換性のあるパッケージの最新バージョンに更新できます。まだインストールされていないパッケージが意図せずにインストールされることを避けるには、`pkg update` コマンドを使用して、パッケージを更新します。

イメージで複数のパブリッシャーが有効になっている場合は、パブリッシャーのスティッキネスと検索順序を設定したり、またはパッケージ FMRI でパブリッシャーを指定したりすることによって、パッケージを提供するパブリッシャーを制御できます。また、インストールするバージョンをパッケージ FMRI で指定することもできます。パッケージの最新バージョンを明示的に要求するには、パッケージ名のバージョン部分に `latest` を使用します。パッケージ FMRI の説明については、16 ページの「障害管理リソース識別子」を参照してください。パブリッシャーのスティッキネスと検索順序の設定については、99 ページの「パブリッシャーの構成」を参照してください。

更新されるパッケージの一部となる保持される構成ファイルは、ファイルの `preserve` 属性の値と、ファイルが変更されたかどうかに応じて、インストール、保存、または名前変更されます。パッケージの更新中にファイルが保持されるしくみについては、`pkg(5)` のマニュアルページのファイルアクションに関するセクションで `preserve` 属性を参照してください。

パブリッシャーのスティッキネスと検索順序および `-g` オプションの使用については、48 ページの「新しいパッケージのインストール」を参照してください。

現在インストールされていないパッケージの更新を試行すると、`pkg update` 操作はパッケージを更新せずに終了します。インストールされていないパッケージを無視することで、更新する一部のパッケージが現在インストールされていない場合に `pkg update` が失敗するのを回避するには、`--ignore-missing` オプションを使用します。

パッケージ FMRI またはパターンが指定されていないとき、または指定されたパターンがアスタリスク文字 (*) である場合の `pkg update` コマンドの特殊な動作については、第4章「Oracle Solaris イメージの更新またはアップグレード」を参照してください。

パッケージのダウングレード

`pkg update` コマンドを使用して、パッケージのアップグレードだけでなくダウングレードを行うこともできます。パッケージをダウングレードするには、現在インス

トールされているよりも古いバージョンのパッケージ FMRI を指定します。パッケージ FMRI の説明については、16 ページの「障害管理リソース識別子」を参照してください。インストールされているパッケージのバージョンと、構成されているパブリッシャーから使用可能なバージョンを表示するには、`pkg list` コマンドを使用します。

ダウングレードされるパッケージの一部となる保持される構成ファイルは、ファイルの `preserve` 属性の値と、ファイルが変更されたかどうかに応じてインストールまたは名前変更されます。パッケージのダウングレード中にファイルが保持されるしくみについては、pkg(5) のマニュアルページのファイルアクションに関するセクションで `preserve` 属性を参照してください。

`-g` オプションの使用については、48 ページの「新しいパッケージのインストール」を参照してください。

インストール済みパッケージの問題の修正

IPS では、インストール済みパッケージが正しくインストールされていることを検証したり、検証の問題を修正したり、インストール済みファイルをそれらがパッケージされた状態に復元したりする操作が提供されています。

注記 - セキュリティーベストプラクティスでは、定期的に `pkg verify -v` を実行して、パッケージ化されたファイルシステムオブジェクトが危険に変更されていないことを確認することをお勧めします。

`pkg fix` コマンドと `pkg revert` コマンドの比較

`pkg fix` コマンドと `pkg revert` コマンドは両方とも、インストール済みパッケージのコンポーネントを再インストールします。次の表に、これら 2 つのコマンドについてのいくつかの類似点と相違点を示します。

`pkg fix`	■ パッケージに対して操作が行われます。パッケージ名に一致する 1 つ以上のパッケージ名またはパターンをオペランドとして指定します。 ■ `pkg verify` で失敗したパッケージについてのみ操作が行われます。 ■ `pkg verify` で報告されたエラーのみ修正します。ほかの内容またはメタデータをパッケージからふたたび提供しません。
`pkg revert`	■ ファイルに対して操作が行われます。1 つ以上のパッケージ名またはタグ名をオペランドとして指定します。

	■ オペランドで識別されたファイルをふたたび提供します。ほかの内容またはメタデータをパッケージからふたたび提供しません。

パッケージの検証と検証エラーの修正

イメージ内のパッケージのインストールを検証するには、`pkg verify` コマンドを使用します。関連するパブリッシャーの現在の署名ポリシーが `ignore` でない場合、各パッケージの署名がポリシーに基づいて検証されます。署名ポリシーが適用されるしくみについては、124 ページの「署名付きパッケージのイメージプロパティー」を参照してください。インストール済みパッケージの内容は、独自の内容解析に基づいて検証されるため、ほかのプログラムの場合とは異なる結果が返されることがあります。

パッケージ名を指定しない場合、すべてのインストール済みパッケージが検査されます。`-v` オプションは、インストール済みパッケージごとに少なくとも 1 行の情報メッセージを提供します。次の例には、出力の少量のサンプルのみが表示されています。`pkg/depot` パッケージのインストールにエラーがあります。

```
$ pkg verify -v
PACKAGE                                                                 STATUS
pkg://solaris/archiver/gnu-tar                                             OK
pkg://solaris/audio/audio-utilities                                        OK
pkg://solaris/benchmark/x11perf                                            OK
...
pkg://solaris/package/pkg/depot                                         ERROR
        dir: var/cache/pkg/depot
                Group: 'pkg5srv (97)' should be 'bin (2)'
        file: var/log/pkg/depot/access_log
                editable file has been changed
        file: var/log/pkg/depot/error_log
                editable file has been changed
...
pkg://solaris/security/sudo                                                OK
        file: etc/sudoers
                editable file has been changed
...
pkg://solaris/x11/xlock                                                    OK
pkg://solaris/x11/xmag                                                     OK
pkg://solaris/x11/xvidtune                                                 OK
```

`pkg verify` コマンドで報告されたパッケージエラーを修正するには、`pkg fix` コマンドを使用します。修正がライブイメージで変更できないファイルに影響を与える場合、修正は新しい BE で実行されます。`-nv` オプションを指定すると、どのような変更が行われるかを確認できます。44 ページの「ブート環境オプション」の説明に従って BE オプションを指定することもできます。

`pkg verify` の出力には、インストールされた `sudo` パッケージのコンポーネントはパッケージされたコンポーネントと異なるということが表示されますが、これらの相違点は検証エラーとして報告されません。`pkg fix` は変更を行いません。`/etc/sudoers` ファイルは置換されません。

```
$ pkg fix pkg://solaris/security/sudo
```

```
No repairs for this image.
```

/etc/sudoers ファイルを削除すると、検証は失敗し、pkg fix によってファイルが置換されます。

```
$ pkg fix pkg://solaris/security/sudo
Verifying: pkg://solaris/security/sudo                          ERROR
        file: etc/sudoers
                Missing: regular file does not exist
Created ZFS snapshot: 2014-03-13-22:05:42
Repairing: pkg://solaris/security/sudo
Creating Plan (Evaluating mediators):

DOWNLOAD                                PKGS         FILES    XFER (MB)   SPEED
Completed                                1/1           1/1      0.0/0.0  990B/s

PHASE                                          ITEMS
Updating modified actions                       1/1
Updating package state database                Done
Updating package cache                          0/0
Updating image state                           Done
Creating fast lookup database                  Done
```

1 つのファイルをダウンロードし、1 つのアクション (file アクション) を変更したという記述が示すように、欠落したファイルのみが置換されます。その他の sudo パッケージ内容は影響を受けていません。修復を実行する前に、操作によって現在のインストールのスナップショットが保存されています。pkg fix 出力の「Created ZFS snapshot」の行を参照してください。修復は現在のイメージ内で実行されました。

```
$ zfs list -r rpool/ROOT/s11
NAME                             USED  AVAIL  REFER  MOUNTPOINT
rpool/ROOT/s11                  16.3G  22.5G  26.1G  /
rpool/ROOT/s11@2014-03-13-23:52:19   249M      -  26.1G  -
```

pkg verify の出力は、インストールされている pkg/depot パッケージ内のディレクトリの所有権にエラーがあることを示しています。pkg fix の出力は、「Verifying」セクションにのみエラーを表示しています。パッケージ化されたコンポーネントとのほかの相違点は表示されません。

```
$ ls -ld /var/cache/pkg/depot
drwxr-xr-x   3 pkg5srv  pkg5srv        3 Dec  2 19:47 /var/cache/pkg/depot/
$ pkg fix pkg://solaris/package/pkg/depot
Verifying: pkg://solaris/package/pkg/depot                      ERROR
        dir: var/cache/pkg/depot
                Group: 'pkg5srv (97)' should be 'bin (2)'
Created ZFS snapshot: 2014-03-13-22:18:52
Repairing: pkg://solaris/package/pkg/depot
Creating Plan (Evaluating mediators):

PHASE                                          ITEMS
Updating modified actions                       1/1
Updating package state database                Done
Updating package cache                          0/0
Updating image state                           Done
Creating fast lookup database                  Done
```

次の出力は、エラーのみが修正されたことを示します。インストールされたコンポーネントとパッケージ化されたコンポーネントのその他の相違点は、そのまま残っています。

```
$ ls -ld /var/cache/pkg/depot
drwxr-xr-x   3 pkg5srv  bin            3 Dec  2 19:47 /var/cache/pkg/depot/
$ pkg verify -v pkg://solaris/package/pkg/depot
PACKAGE                                                                 STATUS
pkg://solaris/package/pkg/depot                                            OK
        file: var/log/pkg/depot/access_log
                editable file has been changed
        file: var/log/pkg/depot/error_log
                editable file has been changed
```

ファイルの復元

ファイルをパッケージ化されたときの状況に復元する場合は、`pkg revert` コマンドを使用します。ファイルの所有権および保護も復元されます。

注意 - 一部の編集可能ファイルを元に戻すと、システムがブート不可になったり、その他の異常動作の原因になったりする可能性があります。

重要な編集可能ファイルを元に戻すには、`--require-backup-be` オプションを使用します。

名前付きファイルを元に戻す

次の例は、パッケージ化されたバージョンとは異なる `pkg/depot` パッケージからの 2 つのインストール済みファイルの 1 つを指定します。

```
$ pkg revert -v /var/log/pkg/depot/access_log
              Packages to fix:         1
     Estimated space available:  21.08 GB
Estimated space to be consumed: 460.87 MB
       Create boot environment:        No
Create backup boot environment:        No
          Rebuild boot archive:        No

Changed packages:
solaris
  package/pkg/depot
    0.5.11,5.11-0.175.2.0.0.33.0:20140217T134751Z
DOWNLOAD                                  PKGS         FILES    XFER (MB)   SPEED
Completed                                  1/1           1/1      0.0/0.0   50B/s

PHASE                                          ITEMS
Updating modified actions                        1/1
Updating package state database                 Done
Updating package cache                           0/0
Updating image state                            Done
Creating fast lookup database                   Done
```

指定されたファイルはパッケージ化されたバージョンで置き換えられました。`pkg.depot` パッケージのほかのコンポーネントは変更されていません。

タグ付けされたファイルおよびディレクトリを元に戻す

次の操作を実行するには、`--tagged` オプションを使用します。

- 指定されたタグ名でタグ付けされたすべてのファイルを元に戻します。
- 指定したタグ名を持つディレクトリの下にあり、指定されたパターンに一致する、パッケージ化されていないファイルまたはディレクトリを削除します。

詳細については、「File Actions」 in 『Packaging and Delivering Software With the Image Packaging System in Oracle Solaris 11.3』および「Directory Actions」 in 『Packaging and Delivering Software With the Image Packaging System in Oracle Solaris 11.3』の `revert-tag` 属性の説明を参照してください。

次のコマンドは、タグ名 `system:dev-init` でタグ付けされたファイルを表示します。これらのファイルはそのシステムに固有の構成を含んでおり、回復用アーカイブに含まれるべきではないため、回復用アーカイブの作成時にこれらのファイルはパッケージ化された状態に戻されます。詳細は、`archiveadm`(1M) のマニュアルページを参照してください。

```
$ pkg contents -H -a revert-tag='system:dev-init*' '*'
```

クローンアーカイブの作成時に、次のファイルはパッケージ化された状態に戻されます。前の例で説明されているインスタンス固有の情報に加えて、ログファイルの内容やいくつかの構成ファイルなどの情報も、クローンアーカイブでは元に戻されます。

```
$ pkg contents -H -a revert-tag='system:dev-init*' -a revert-tag='system:clone*' '*'
```

次のコマンドは、タグ名 `system:dev-init` を持つすべてのファイルを元に戻す操作のプレビューを示します。元に戻されるファイルは `-v` オプションで一覧表示されますが、この例では示されていません。ブートアーカイブが再構築される点に注意してください。`--be-name` オプションを使用して、わかりやすい名前を持つ新規ブート環境を作成することをお勧めします。

```
$ pkg revert -nv --tagged system:dev-init
              Packages to fix:         5
     Estimated space available: 867.33 GB
Estimated space to be consumed: 240.24 MB
       Create boot environment:        No
Create backup boot environment:       Yes
          Rebuild boot archive:       Yes
```

パッケージのアンインストール

インストールされているパッケージを削除するには、`pkg uninstall` コマンドを使用します。

パッケージをアンインストールするときのコマンド出力はパッケージをインストールするときの出力に非常によく似ていて、たとえば、「Packages to install」の代わりに「Packages to remove」と表示されます。

52 ページの「パッケージの拒否」で説明されているように、`pkg install` コマンドの `--reject` オプションはインストール済みのパッケージを削除することもできます。

例 6 パッケージインストールの取り消し

パッケージをアンインストールしても、そのパッケージの依存関係はアンインストールされません。依存関係を必要とするパッケージのみをアンインストールする場合でも、ユーザーは依存関係パッケージによって提供されるコンテンツを引き続き使用することができます。

パッケージのインストール操作を取り消して、インストール操作でインストールしたすべてのパッケージをアンインストールする場合は、`pkg history` コマンドを使用して、インストールされている正確な内容を確認します。詳細は、130 ページの「操作履歴の表示」を参照してください。

次のコマンドでは、18 個の新しいパッケージがインストールされています。

```
$ pkg install -v amp
           Packages to install:        18
...

Changed packages:
solaris
  database/mysql-55
    None -> 5.5.43,5.11-0.175.3.0.0.30.0:20150821T162149Z
  database/mysql-common
    None -> 5.11,5.11-0.175.3.0.0.30.0:20150821T163446Z
  group/feature/amp
    None -> 0.5.11,5.11-0.175.3.0.0.30.0:20150821T165202Z
  system/library/security/libmcrypt
    None -> 2.5.8,5.11-0.175.3.0.0.30.0:20150821T172551Z
  text/tidy
    None -> 1.0.0,5.11-0.175.3.0.0.30.0:20150821T172736Z
  web/php-53
    None -> 5.3.29,5.11-0.175.3.1.0.2.0:20150921T191759Z
  web/php-53/extension/php-apc
    None -> 3.1.9,5.11-0.175.3.1.0.2.0:20150921T191747Z
  web/php-53/extension/php-idn
    None -> 0.2.0,5.11-0.175.3.1.0.2.0:20150921T191749Z
  web/php-53/extension/php-memcache
    None -> 3.0.6,5.11-0.175.3.1.0.2.0:20150921T191750Z
  web/php-53/extension/php-mysql
    None -> 5.3.29,5.11-0.175.3.1.0.2.0:20150921T191751Z
  web/php-53/extension/php-pear
    None -> 5.3.29,5.11-0.175.3.1.0.2.0:20150921T191752Z
  web/php-53/extension/php-suhosin
    None -> 0.9.33,5.11-0.175.3.1.0.2.0:20150921T191754Z
  web/php-53/extension/php-tcpwrap
    None -> 1.1.3,5.11-0.175.3.1.0.2.0:20150921T191755Z
  web/php-53/extension/php-xdebug
    None -> 2.2.0,5.11-0.175.3.1.0.2.0:20150921T191756Z
```

```
  web/php-common
    None -> 11.1,5.11-0.175.3.0.0.30.0:20150821T173003Z
  web/server/apache-22/module/apache-dtrace
    None -> 0.3.1,5.11-0.175.3.0.0.30.0:20150821T173046Z
  web/server/apache-22/module/apache-fcgid
    None -> 2.3.9,5.11-0.175.3.0.0.30.0:20150821T173050Z
  web/server/apache-22/module/apache-php53
    None -> 5.3.29,5.11-0.175.3.1.0.2.0:20150921T191828Z
...
```

次のコマンドは、amp パッケージのすべての依存関係のリストが、インストールされたパッケージのリストと一致しないことを示しています。amp の一部の依存はすでにインストールされていますが、インストールされている一部のパッケージは、amp の依存関係に依存します。

```
$ pkg contents -o type,fmri -t depend amp
TYPE  FMRI
group web/server/apache-22
group web/server/apache-22/module/apache-dtrace
group web/server/apache-22/module/apache-fcgid
group web/php-53/extension/php-mysql
group web/server/apache-22/module/apache-php53
group web/php-53/extension/php-apc
group web/php-53
group database/mysql-55
```

元の pkg install 出力がない場合にこのインストールを取り消すには、履歴でインストールされているパッケージのリストを見つけます。短いリストを使用して目的のインストールコマンドを見つけてから、130 ページの「操作履歴の表示」に示されているように、-l オプションを使用して、その時点でインストールされていた正確な内容を一覧表示します (-t)。リストは見出し「End State」の下の出力の末尾にあります。

```
$ pkg history -n 1
START                   OPERATION               CLIENT             OUTCOME
2016-10-14T07:05:53     install                 pkg                Succeeded
$ pkg history -lt 2016-10-14T07:05:53
         Operation: install
...
           Command: /usr/bin/pkg install -v amp
...
         End State:
None -> pkg://solaris/web/php-53@5.3.29,5.11-0.175.3.1.0.2.0:20150921T191759Z
None -> pkg://solaris/group/feature/amp@0.5.11,5.11-0.175.3.0.0.30.0:20150821T165202Z
None -> pkg://solaris/web/php-53/extension/php-
tcpwrap@1.1.3,5.11-0.175.3.1.0.2.0:20150921T191755Z
None -> pkg://solaris/web/php-53/extension/php-
mysql@5.3.29,5.11-0.175.3.1.0.2.0:20150921T191751Z
None -> pkg://solaris/web/server/apache-22/module/apache-
php53@5.3.29,5.11-0.175.3.1.0.2.0:20150921T191828Z
None -> pkg://solaris/web/server/apache-22/module/apache-
dtrace@0.3.1,5.11-0.175.3.0.0.30.0:20150821T173046Z
None -> pkg://solaris/web/php-53/extension/php-
memcache@3.0.6,5.11-0.175.3.1.0.2.0:20150921T191750Z
None -> pkg://solaris/web/php-common@11.1,5.11-0.175.3.0.0.30.0:20150821T173003Z
None -> pkg://solaris/web/php-53/extension/php-
xdebug@2.2.0,5.11-0.175.3.1.0.2.0:20150921T191756Z
None -> pkg://solaris/database/mysql-55@5.5.43,5.11-0.175.3.0.0.30.0:20150821T162149Z
None -> pkg://solaris/web/php-53/extension/php-
suhosin@0.9.33,5.11-0.175.3.1.0.2.0:20150921T191754Z
```

```
None -> pkg://solaris/system/library/security/
libmcrypt@2.5.8,5.11-0.175.3.0.0.30.0:20150821T172551Z
None -> pkg://solaris/database/mysql-common@5.11,5.11-0.175.3.0.0.30.0:20150821T163446Z
None -> pkg://solaris/web/php-53/extension/php-
apc@3.1.9,5.11-0.175.3.1.0.2.0:20150921T191747Z
None -> pkg://solaris/web/php-53/extension/php-
idn@0.2.0,5.11-0.175.3.1.0.2.0:20150921T191749Z
None -> pkg://solaris/web/server/apache-22/module/apache-
fcgid@2.3.9,5.11-0.175.3.0.0.30.0:20150821T173050Z
None -> pkg://solaris/web/php-53/extension/php-
pear@5.3.29,5.11-0.175.3.1.0.2.0:20150921T191752Z
None -> pkg://solaris/text/tidy@1.0.0,5.11-0.175.3.0.0.30.0:20150821T172736Z
```

例 7 アンインストールする複数のパッケージの指定

`pkg uninstall` コマンドに複数のパッケージを指定する場合、または `-r` などのオプションを指定して非大域ゾーンでこの操作を実行する場合は、そのイメージにインストールされていないパッケージをアンインストールしようとしている可能性があります。

現在インストールされていないパッケージをアンインストールしようとすると、`pkg uninstall` 操作はパッケージをアンインストールせずに終了します。

アンインストールするパッケージがそのイメージ内に存在していない場合は、そのパッケージ名に一致するパッケージがないというメッセージが表示されます。アンインストールする追加のパッケージがイメージ内に存在する場合、それらのパッケージがアンインストールされていないことを示すメッセージは表示されません。

指定されたパッケージのうちインストールされているものはすべてアンインストールし、指定されたパッケージのうちインストールされていないものに対してはこの操作をスキップするには、`--ignore-missing` オプションを使用します。`--ignore-missing` オプションは、アンインストールする一部のパッケージが現在インストールされていない場合に、インストールされていないパッケージを無視し、`pkg uninstall` が失敗するのを回避します。

次のコマンドでは、一部のゾーンに `amp` パッケージがインストールされていても、1つのゾーンに `amp` がインストールされていない場合、パッケージはアンインストールされません。

```
$ pkg uninstall -r amp
```

次のコマンドでは、`amp` パッケージは、現在インストールされているすべてのゾーンからアンインストールされます。

```
$ pkg uninstall -r --ignore-missing amp
```

例 8 別のパッケージが必要としているパッケージのアンインストール

インストールされている別のパッケージが必要としているパッケージをアンインストールしようとすると、失敗します。

たとえば、mysql-common パッケージをアンインストールしようとすると、次のエラーメッセージを受け取ります。

```
$ pkg uninstall mysql-common
pkg uninstall: Unable to remove 'database/mysql-common@5.11-0.175.3.0.0.30.0' due to the
 following packages that depend on it:
  database/mysql-55@5.5.43-0.175.3.0.0.30.0
```

次の出力によって、mysql-common が mysql-55 に必要であることが確認されます。

```
$ pkg contents -o type,pkg.name -t depend -a fmri='*mysql-common*' '*mysql-55'
TYPE    PKG.NAME
require database/mysql-55
```

mysql-common をアンインストールするには、mysql-common と mysql-55 を両方ともアンインストールする必要があります。単に mysql-55 をアンインストールしても、依存パッケージ mysql-common はアンインストールされないことに注意してください。

```
$ pkg uninstall mysql-common mysql-55
```

例 9 group 依存関係であるパッケージのアンインストール

group 依存関係であるパッケージをアンインストールすると、そのパッケージは回避リストに登録されます。回避リストについては、119 ページの「グループパッケージに含まれる一部のパッケージのインストールの回避」を参照してください。

```
$ pkg avoid
$ pkg uninstall mysql-55
...
$ pkg list mysql-55
pkg list: No packages matching 'mysql-55' installed
$ pkg avoid
    database/mysql-55 (group dependency of 'group/feature/amp')
```

イメージの再インストール

最終的に必要な結果が正確にわかっていて、その結果に到達するために、大量のパッケージをアンインストールするなどの大量のパッケージ変更が必要な場合、pkg exact-install コマンドを使用した方がよい場合もあります。pkg exact-install コマンドの結果は、指定されたパッケージとそれらの依存関係のみがインストールされたイメージとなります。現在インストール済みのパッケージのうち、pkg exact-install コマンド行で指定されず、指定されたパッケージの依存関係でもないものは、削除されます。

pkg:/entire 制約パッケージのバージョンが、exact-install のオペランドに含まれている必要があります。pkg:/entire がインストールされていないイメージはサポー

トされません。pkg:/entire パッケージのバージョンは、現在インストールされている pkg:/entire パッケージのバージョン以上である必要があります。

pkg exact-install コマンドは、回避リストにあるパッケージをインストールしないという制約を無視します。パッケージが回避リストにある場合は、インストールすると回避リストから削除されます。回避リストについては、119 ページの「グループパッケージに含まれる一部のパッケージのインストールの回避」を参照してください。pkg exact-install コマンドは、凍結リストにあるパッケージを更新しないという制約を無視します。凍結されたパッケージについては、113 ページの「指定したバージョンへのパッケージのロック」を参照してください。

それ以外については、exact-install サブコマンドは install サブコマンドと同じように動作します。イメージバリアントとファセットの設定、イメージプロパティー設定、およびパブリッシャー設定は保持されます。いずれかのパッケージをこのイメージにインストールできない場合、指定されたパッケージは一切インストールされません。64 ページの「非大域ゾーンの操作」に記載されているように、非大域ゾーンはパッケージの更新または削除の結果による影響を受けます。exact-install には -r オプションを使用できません。

パブリッシャーのスティッキネスと検索順序および -g オプションの使用については、48 ページの「新しいパッケージのインストール」を参照してください。

pkg exact-install コマンドを使用する際は、次のプラクティスを推奨します。

- 24 ページの「インストール可能なパッケージ」に記載されているように、pkg list -a を使用して、構成済みのパブリッシャーから使用可能なパッケージのバージョンを確認します。exact-install を使用して現在のバージョンを再インストールする場合、新しいバージョンがインストール可能であれば、インストールするパッケージの一覧にパッケージ FMRI のバージョン部分を指定する必要があります。
- インストールするパッケージの一覧に、pkg:/entire 制約パッケージを含めます。
- インストールするパッケージの一覧に、solaris-minimal-server パッケージなどのシステムグループパッケージの 1 つを含めます。
- -nv または -nvv オプションを付けてコマンドを最初に実行し、インストールされるものと削除されるものを正確に調べます。
- わかりやすい名前を持つ新規 BE にインストールするために、--be-name オプションを使用します。

次の例では、現在のイメージと同じバージョンの最小インストールによる新規イメージが作成されます。

```
$ pkg list -Hv entire
pkg://solaris/entire@0.5.11,5.11-0.175.2.0.0.34.0:20140303T182643Z
$ pkg exact-install --be-name s11.2 entire@0.5.11,5.11-0.175.2.0.0.34 solaris-minimal-server
```

非大域ゾーンの操作

ほとんどの IPS コマンドは、コマンドを大域ゾーンで使用する場合と同じ方法で非大域ゾーンで使用できます。ゾーンの概要情報は、20 ページの「イメージとブート環境」を参照してください。

パッケージインストールについては、64 ページの「大域ゾーンと非大域ゾーンの関係」および67 ページの「複数の非大域ゾーンの同時更新」に記載されているように、大域ゾーンと非大域ゾーンは親子関係を持ちます。

65 ページの「システムリポジトリおよびプロキシサービス」に記載されているように、大域ゾーンと非大域ゾーンの重要な違いは、パッケージパブリッシャーの使用方法です。

大域ゾーンと非大域ゾーンの関係

インストールされた `solaris` ブランドの非大域ゾーンは、大域ゾーンでのパッケージのインストール、更新、およびアンインストールの影響を受けることがあります。

ファセットとバリアントを変更すると、パッケージのインストールおよび削除が行われ、非大域ゾーンに影響することがあります。

大域ゾーンから非大域ゾーンを更新する場合、非大域ゾーンをブートする必要はありません。非大域ゾーンをただインストールするだけで、大域ゾーンのパッケージ変更による影響を受けることになります。

インストールおよび更新コマンドを大域ゾーンで実行すると、デフォルトでは、大域ゾーンとインストール済みの各非大域ゾーンは順に更新され、非大域ゾーンの変更は、非大域ゾーンが大域ゾーンとの互換性を維持するために必要な範囲で実行されます。

- 非大域ゾーンで必要最小限の更新を実行する代わりに、大域ゾーンで実行するのと同じ操作を非大域ゾーン内で実行するには、45 ページの「非大域ゾーンで動作するオプション」に記載されている `-r` オプションを使用します。
- 複数の非大域ゾーンを大域ゾーンと同時に更新するには、45 ページの「非大域ゾーンで動作するオプション」に記載され、67 ページの「複数の非大域ゾーンの同時更新」に示されているように、`-C` オプションを使用します。

ヒント - 大域ゾーンとともに非大域ゾーンに加えられる変更を確認するには、`-nv` オプションを使用します。

非大域ゾーンにログインしているときにパッケージコマンドを実行すると、その非大域ゾーンだけが影響を受けます。次に例を示すように、非大域ゾーンはその親となる大域ゾーンと異なっていてもかまいません。

- 異なるパッケージをインストールできます。
- 結果として大域ゾーンとの互換性が保たれる場合は、同じパッケージの異なるバージョンをインストールできます。
- 異なるパッケージを回避リストに載せることができます。
- 異なるパッケージを凍結したり、異なるバージョンで凍結したりできます。
- 異なるデフォルト実装を選択するようにメディエータを設定できます。
- 異なるファセットを設定できます。

大域ゾーンにインストールされるバージョンにより、非大域ゾーンにインストールされるパッケージのバージョンが制限されることがあります。非大域ゾーンの一部のパッケージは、非大域ゾーンと大域ゾーンで同じバージョンでなければならないため、更新やダウングレードができません。たとえば、`entire` という名前のパッケージは、各非大域ゾーンと大域ゾーンで同じバージョンでなければなりません。

非大域ゾーンにインストールされるパッケージが `parent` 依存関係を持つ場合、大域ゾーン内でそのパッケージを更新すると、非大域ゾーンでそのパッケージが更新されます。`parent` 依存関係を持つパッケージに依存しているパッケージも影響を受けます。

`parent` 依存関係による影響を受けないパッケージは、大域ゾーンにインストールされているバージョンと異なるバージョンで非大域ゾーンにインストールできます。非大域ゾーンで異なるバージョンをインストールするには、`pkg install` コマンドでバージョンを指定するか、必要なバージョンでバージョンを凍結します。

非大域ゾーンにパッケージをインストールする際に役立つ関連情報については、156 ページの「同期リンクされたパッケージをインストールできない」および158 ページの「非大域ゾーンをインストールできない」を参照してください。

システムリポジトリおよびプロキシサービス

非大域ゾーンでは、「システムリポジトリ」は大域ゾーン内に構成されているパッケージリポジトリへのアクセスを提供します。大域ゾーンに加えられたパブリッシャー構成の変更は、システムリポジトリを介してただちにすべての非大域ゾーンで認識されます。

非大域ゾーンに構成されたパブリッシャーの起点またはミラーは、その場所が大域ゾーンパブリッシャーリストに構成されていなくても、大域ゾーンからアクセス可能でなければなりません。たとえば、`localsw` パブリッシャーが非大域ゾーンに構成さ

れているが、大域ゾーンには構成されていない場合、localsw パブリッシャーのすべての起点およびミラーは大域ゾーンからアクセス可能でなければなりません。

システムリポジトリは http、https、ファイル、および .p5p アーカイブリポジトリをプロキシ設定できます。ファイルシステムリポジトリバージョン 4 のみがサポートされており、これは pkgrepo create コマンドのデフォルトの形式です。

ゾーンプロキシは、ゾーン内で実行されている pkg コマンドが、大域ゾーン内で実行されているシステムリポジトリと通信できるようにするサービスです。ゾーンプロキシには 2 つの部分があります。次のサービスは大域ゾーンで実行されます。

```
svc:/application/pkg/zones-proxyd:default
```

次のサービスは非大域ゾーンで実行されます。

```
svc:/application/pkg/zones-proxy-client:default
```

システムリポジトリとゾーンプロキシサービスの詳細については、pkg.sysrepo(1M) のマニュアルページを参照してください。

次の例は、大域ゾーンのパブリッシャーを示しています。

```
global:~$ pkg publisher
PUBLISHER             TYPE     STATUS P LOCATION
solaris               origin   online F http://pkg.oracle.com/solaris/release/
solaris               origin   online F file:///var/share/pkgrepos/solaris/
devtool  (disabled)   origin   online F http://pkg.example1.com/
isvpub                origin   online F http://pkg.example2.com/
```

次の例は、非大域ゾーンにログインしているときに、これらの同じパブリッシャーがどのように見えるかを示しています。

```
z1:~$ pkg publisher
PUBLISHER             TYPE     STATUS P LOCATION
solaris  (syspub)     origin   online T <system-repository>
solaris  (syspub)     origin   online F <system-repository>
isvpub   (syspub)     origin   online T <system-repository>
```

非大域ゾーンでは、無効になっているリポジトリは使用できません。

system-repository の場所の URI とプロキシ値を表示するには、-F オプションを使用します。

```
z1:~$ pkg publisher -F tsv
PUBLISHER  STICKY  SYSPUB  ENABLED  TYPE    STATUS  URI
 PROXY
solaris    true    true    true     origin  online  http://pkg.oracle.com/solaris/release/
 http://localhost:1008
solaris    true    true    true     origin  online  http://localhost:1008/
solaris/35024e7d1859bedee9af156d22a591c433adc0ee/ -
isvpub     true    true    true     origin  online  http://pkg.example2.com/
 http://localhost:1008
```

大域ゾーン内の file:// リポジトリには、非大域ゾーン内の http:// の場所が割り当てられています。

非大域ゾーンでは、システムリポジトリは常にプロキシとして表示されます。これは、非大域ゾーンが大域ゾーンのシステムリポジトリと通信するために使用するプロキシです。

非大域ゾーン内からシステムリポジトリを再構成することはできません。たとえば、パブリッシャーの場所が `<system-repository>` の場合、パブリッシャーの起点やプロパティー、およびパブリッシャーの検索順序を変更することはできません。パブリッシャーが大域ゾーンに追加または再構成された場合、これらの変更は非大域ゾーンによってただちに認識されます。パブリッシャーが大域ゾーン内で設定解除される場合、そのパブリッシャーからパッケージが非大域ゾーンにインストールされないかぎり、パブリッシャーは非大域ゾーン内で設定解除されます。

ヒント - 大域ゾーン内のパブリッシャーを設定解除する前に、非大域ゾーン内でそのパブリッシャーからのパッケージをアンインストールしてください。

パブリッシャーにアクセスできない場合、105 ページの「プロキシの指定」に記載されているように、大域ゾーンにプロキシを設定することができます。`http_proxy` および `https_proxy` 環境変数を使用する場合と使用方法についての説明を含む、非大域ゾーンを持つ場合のプロキシ設定の詳細については、「Proxy Configuration on a System That Has Installed Zones」 in 『Creating and Using Oracle Solaris Zones』を参照してください。

大域ゾーンですでに構成されているパブリッシャーについて、`pkg list` コマンドは、大域ゾーンと非大域ゾーンの両方で同じ結果を生成します。

```
z1:~$ pkg list -a isvtool
NAME (PUBLISHER)    VERSION    IFO
isvtool (isvpub)    2.0        ---
isvtool (isvpub)    1.0        ---
```

リポジトリが大域ゾーンで構成されていなくても、リポジトリは非大域ゾーンにアクセスできるネットワークまたはファイルシステムとすることができます。非大域ゾーンのパブリッシャー構成は、大域ゾーンのパブリッシャー構成と一致するか、大域ゾーンのパブリッシャー構成のスーパーセットでなければなりません。たとえば、`localsw` パブリッシャーは起点 `file:///var/share/pkgrepos/localrepo` を使用して非大域ゾーン内で構成できますが、これは、`localsw` パブリッシャーが大域ゾーン内で構成されていなくても、この場所が大域ゾーン内でアクセス可能であるためです。

複数の非大域ゾーンの同時更新

デフォルトでは、大域ゾーンで `pkg update` コマンドを使用すると、パッケージシステムは大域ゾーンと各非大域ゾーンを順に更新します。複数の非大域ゾーンを同時に

更新するには、-C オプションを使用するか、大域ゾーンで PKG_CONCURRENCY 環境変数を設定します。詳細は、45 ページの「非大域ゾーンで動作するオプション」を参照してください。

次の例では、両方の非大域ゾーンが大域ゾーンと同時に更新されます。非大域ゾーンは親の大域ゾーンイメージにリンクされているため、この出力では非大域ゾーンをリンクされたイメージとして示しています。

```
global:~$ pkg update -C 0 --be-name s11.2
 Startup: Linked image publisher check ... Done
 Startup: Refreshing catalog 'solaris' ... Done
 Startup: Refreshing catalog 'isvpub' ... Done
 Startup: Checking that pkg(5) is up to date ... Done
Planning: Solver setup ... Done
Planning: Running solver ... Done
Planning: Finding local manifests ... Done
Planning: Package planning ... Done
Planning: Merging actions ... Done
Planning: Checking for conflicting actions ... Done
Planning: Consolidating action changes ... Done
Planning: Evaluating mediators ... Done
Planning: Planning completed in 39.00 seconds
            Packages to remove:   2
           Packages to install:   1
            Packages to update: 640
       Create boot environment: Yes
Create backup boot environment:  No

Planning: Linked images: 0/2 done; 2 working: zone:z1 zone:z2
Planning: Linked image 'zone:z1' output:
| Packages to install:   1
|  Packages to update: 161
|  Services to change:   2
`
Planning: Linked images: 1/2 done; 1 working: zone:z2
Planning: Linked image 'zone:z2' output:
| Packages to install:   1
|  Packages to update: 161
|  Services to change:   2
`
Planning: Finished processing linked images.
Download:     0/12068 items    0.0/350.9MB  0% complete
...
Download: 11664/12068 items  336.1/350.9MB  95% complete
Download: Completed 350.91 MB in 187.08 seconds (0B/s)
Download: Linked images: 0/2 done; 2 working: zone:z1 zone:z2
Download: Linked images: 1/2 done; 1 working: zone:z1
Download: Finished processing linked images.
 Actions:     1/23382 actions (Removing old actions)
 Actions:  3867/23382 actions (Installing new actions)
 Actions:  8192/23382 actions (Updating modified actions)
...
 Actions: 23266/23382 actions (Updating modified actions)
 Actions: Completed 23382 actions in 96.16 seconds.
Finalize: Updating package state database ...  Done
Finalize: Updating package cache ...  Done
Finalize: Updating image state ...  Done
Finalize: Creating fast lookup database ...  Done
Finalize: Reading search index ...  Done
Finalize: Building new search index ...  Done
Finalize: Linked images: 0/2 done; 2 working: zone:z1 zone:z2
Finalize: Linked images: 1/2 done; 1 working: zone:z2
```

```
Finalize: Finished processing linked images.

A clone of s11 exists and has been updated and activated.
On the next boot the Boot Environment s11u1 will be
mounted on '/'.  Reboot when ready to switch to this updated BE.
```

♦ ♦ ♦ 第 4 章

Oracle Solaris イメージの更新またはアップグレード

第3章「ソフトウェアパッケージのインストールおよび更新」では、コマンド行で指定された 1 つまたは複数のパッケージのインストール、更新、修正、およびアンインストールについて説明しました。この章では、Oracle Solaris イメージを、次にサポートされる更新または次のリリースにアップグレードする方法について説明します。この章では、次の内容について説明します。

- ベストプラクティス
- 更新の上限の制御
  - バージョン番号の指定
  - 指定されたバージョン番号でのイメージの凍結
  - カスタムの制約パッケージの使用
- サポート更新の適用
  - Support Repository Updates (SRU)
  - クリティカルパッチアップデート (CPU)
- プラットフォームファームウェア更新の適用
- IDR (Interim Diagnostic or Relief) 更新の適用

イメージの更新の概要

パッケージ FMRI またはパターンを指定せずに、あるいはパターンとしてアスタリスク文字 (*) を使用して `pkg update` コマンドを使用すると、使用可能な更新があるすべてのインストール済みパッケージが、インストール済みパッケージの依存関係およびパブリッシャーの構成によってシステムに課せられた制約で許可される最新バージョンに更新されます。

許可される最新バージョンに更新したくない場合は、76 ページの「許可される最新バージョンより古いバージョンへの更新」で説明されているオプションを参照してください。

- 更新されたインストール済みパッケージによって必要となる新しい依存関係でないかぎり、新しいパッケージはインストールされません。
- インストール済みパッケージが更新されるのは、構成済みパブリッシャーによって提供される更新バージョンが、更新されるイメージの制約に適合する場合に限られます。制約は、パッケージ依存関係のほかに、第5章「インストールされるイメージの構成」に記載されている、ユーザーが制御可能な次のような構成によって課されます。
  - 特定のバージョンにロックされたパッケージ
  - イメージに設定されたファセットとバリアント
  - 構成されたパッケージの署名プロパティー
  - パブリッシャーの検索順序およびスティッキネス
- 必要ないずれかのパッケージをインストールできない場合、パッケージは一切更新またはインストールされません。付録A パッケージのインストールおよび更新のトラブルシューティングを参照してください。

システムをアップグレードすることは、ブート可能なイメージを更新することを意味します。20 ページの「イメージとブート環境」に記載されているように、システムは複数のブート可能なイメージを持つことができます。

現在のイメージに非大域ゾーンがインストールされている場合、これらのゾーンも更新されます。64 ページの「非大域ゾーンの操作」を参照してください。

更新には強力な権限が必要です。詳細は、21 ページの「インストール権限」を参照してください。

`pkg update` コマンドのすべてのオプションの完全な一覧については、`pkg`(1) のマニュアルページを参照してください。

イメージ更新のベストプラクティス

- 更新する前に、次の手順を実行してください。
  - ライセンスを確認します。33 ページの「ライセンス要件の表示」は、そのライセンスに同意する必要があるパッケージを一覧表示する方法について説明します。これらのパッケージをインストールまたは更新するには、`--accept` オプションを指定しなければならない場合があります。28 ページの「パッケージライセンスの表示」は、ライセンステキストを表示する方法について説明しています。
  - リリースノートを読みます。Oracle Solaris リリースのリリースノートは、`docs.oracle.com` にあります。SRU のリリースノートは、`support.oracle.com` にあります。
  - 構成済みのパブリッシャーの起点から使用可能なパッケージバージョンを確認します。73 ページの「使用可能なバージョンのチェック」を参照してくだ

さい。パブリッシャーに対して `pkg refresh` コマンドを実行するか、起点の場所で `pkgrepo refresh` コマンドを実行することが必要な場合もあります。独自のリポジトリを作成する場合は、部分的なリポジトリを作成しないでください。『Copying and Creating Package Repositories in Oracle Solaris 11.3』の説明に従って、完全なリポジトリを作成してください。

  - `-nv` オプションを指定して `pkg update` コマンドを使用することで、実際には更新を実行せずに、更新されるパッケージのリストを表示します。75 ページの「更新操作のプレビュー」を参照してください。
- 更新の際は、76 ページの「新しいブート環境の指定」に記載されているように、`--be-name` または `--require-new-be` オプションを使用して、現在のブート環境ではなく新規ブート環境で変更を行います。
- 更新が失敗した場合は、付録A パッケージのインストールおよび更新のトラブルシューティングを確認してください。
- 更新後に、インストールされているパッケージを確認します。

```
$ beadm mount name-of-new-BE /mnt
$ pkg -R /mnt verify -v
```

エラーが報告されている場合は、`pkg fix` コマンドを使用して、エラーが修正されていることを確認します。

```
$ pkg -R /mnt fix -v
$ pkg -R /mnt verify -v
```

BE を必ずアンマウントしてください。

```
$ beadm unmount name-of-new-BE
```

55 ページの「パッケージの検証と検証エラーの修正」も参照してください。

- 更新済みまたはインストール済みのパッケージでリリースノートが報告されているかどうかを確認するには、`pkg history` コマンドで `-N` オプションを使用します。
- 85 ページの「サポート更新の適用」の説明に従って、すべてのサポート更新を適用してシステムを常に最新にします。「Oracle Solaris バイナリおよびソース保証プログラム」(ドキュメント ID 1391762.1) は、リリースの境界を越える更新のリスクが低いことを保証します。

使用可能なバージョンのチェック

オペレーティングシステムのリリースを更新する場合は、`pkg:/entire` 制約パッケージの入手可能なバージョンを確認します。次のコマンドは、Oracle Solaris 11 11/11 SRU 10 がインストールされており、Oracle Solaris 11 11/11 SRU 11、12、および 13 が入手可能であり、さらに現在構成されている `solaris` パブリッシャーから Oracle Solaris 11.1 が入手可能であることを示しています。FMRI 内のフィールドについては、16 ページの「障害管理リソース識別子」を参照してください。

```
$ pkg list -af entire
```

```
NAME (PUBLISHER)  VERSION                       IFO
entire            0.5.11,5.11-0.175.1.0.0.24.2  ---
entire            0.5.11,5.11-0.175.0.13.0.4.0  ---
entire            0.5.11,5.11-0.175.0.12.0.4.0  ---
entire            0.5.11,5.11-0.175.0.11.0.4.1  ---
entire            0.5.11,5.11-0.175.0.10.0.5.0  i--
```

これらの中に必要なバージョンがない場合は、`solaris` パブリッシャー起点を別のパッケージリポジトリの場所に設定する必要があります。

デフォルトでは、各パッケージは、現在インストールされているバージョンを提供したパブリッシャーから更新されます。パブリッシャーのスティッキネスと検索順を指定することにより、パッケージを提供するパブリッシャーを制御できます。101 ページの「パッケージパブリッシャーの追加、変更、削除」を参照してください。

システムは、システムに現在インストールされているパッケージを提供するパッケージリポジトリにアクセスできる必要があります。たとえば、Oracle Solaris 11.2 から Oracle Solaris 11.3 に更新する場合は、`solaris` パブリッシャーに、インストールされている Oracle Solaris 11.2 パッケージと目的の Oracle Solaris 11.3 パッケージの両方へのアクセスが構成されている必要があります。パッケージが `ha-cluster` や `solarisstudio` などの別のパブリッシャーからインストールされている場合は、それらのパブリッシャーにも現在インストールされているパッケージに加えて目的の新しいパッケージへのアクセスが構成されている必要があります。

ほとんどの更新エラーの原因は不完全なパッケージリポジトリです。「Best Practices for Creating and Using Local IPS Package Repositories」 in 『Copying and Creating Package Repositories in Oracle Solaris 11.3』を参照してください。

オープンソースソフトウェアの使用可能なバージョン

Oracle Solaris パッケージリポジトリに、そのリポジトリ内の `pkg:/entire` の最新のバージョンよりも新しいオープンソースソフトウェアのバージョンが含まれている場合があります。たとえば、次のパッケージはすべて同じ Oracle Solaris パッケージリポジトリから使用できる場合があります。

```
entire            0.5.11-0.175.3.1.0.5.0
runtime/python-26  2.6.8-0.175.3.0.0.30.0
runtime/python-27  2.7.11-5.12.0.0.0.95.0
runtime/python-27  2.7.11-5.12.0.0.0.90.0
runtime/python-27  2.7.9-0.175.3.0.0.30.0
runtime/python-34  3.4.4-5.12.0.0.0.95.0
runtime/python-34  3.4.3-5.12.0.0.0.90.0
runtime/python-34  3.4.3-0.175.3.0.0.30.0
runtime/python-35  3.5.1-5.12.0.0.0.95.0
runtime/python-35  3.5.1-5.12.0.0.0.90.0
```

`runtime/python-26`、`runtime/python-27`、および `runtime/python-34` の `0.175.3.0.0.30.0` バージョンが、 (`userland-incorporation@0.5.11-0.175.3.1.0.3.0` によって) オペレーティングシステムバージョン `0.175.3.1.0.5.0`

に組み込まれています。これらのパッケージと runtime/python-35 の新しいバージョンはリポジトリから使用できます。version-lock ファセットをロック解除すると、これらの新しいバージョンをインストールできます。詳細な手順については、使用可能な FOSS コンポーネントを判断する方法のドキュメントを参照してください。いくつかの使用可能な FOSS コンポーネントのリスト、および FOSS コンポーネント関連のハウツー記事については、Oracle Solaris での FOSS に関する記事を参照してください。

更新操作のプレビュー

次のコマンドは、更新によって実際にどのパッケージがインストールされるか (存在する場合) を示しています。-v オプションが指定されているため、このコマンドは、完全な FMRI (複数のバージョンを含む) の 627 のすべてのパッケージが更新され、3 つのパッケージが削除され、1 つの新しいパッケージがインストールされることを示しています。この例では、その出力のほとんどが省略され、entire パッケージのみが示されています。-n オプションが指定されているため、実際に更新は実行されません。-n オプションなしで更新を実行する前に、この出力を確認してください。

```
$ pkg update -nv
            Packages to remove:        3
           Packages to install:        1
            Packages to update:      627
     Estimated space available: 48.43 GB
Estimated space to be consumed:  3.14 GB
       Create boot environment:      Yes
     Activate boot environment:      Yes
Create backup boot environment:       No
          Rebuild boot archive:      Yes

Changed packages:
solaris
...
  entire
    0.5.11,5.11-0.175.0.10.0.5.0:20120803T182627Z -> 0.5.11,5.11-0.175.1.0.0.24.2:
20120919T190135Z
...
```

前の例は、Oracle Solaris 11.1 の pkg:/entire 制約パッケージがインストールされることを示しています。

- それに応じて、entire 制約パッケージによって制約されているすべてのインストール済みパッケージが更新されます。
- パッケージ FMRI が指定されなかったため、entire 制約パッケージによって制約されていないインストール済みパッケージもすべて更新されます。すべてのインストール済みパッケージが、システムの制約で許可された最新バージョンに更新されます。この制約は、インストール済みパッケージの依存関係およびパブリッシャーの構成によってシステムに課せられるものです。
- 更新されたインストール済みパッケージで新しい依存関係が指定されている場合は、インストール済みパッケージが削除され、新しいパッケージがインストールされる可能性があります。

新しいブート環境の指定

75 ページの「更新操作のプレビュー」の例では、-n オプションなしでこのコマンドを実行した場合、この更新に対して新規 BE が作成されることを示しています。-n オプションなしでこのコマンドを実行した場合は、更新出力の最後に次のメッセージが表示されます。

```
A clone of currentBE exists and has been updated and activated.
On the next boot the Boot Environment newBE will be
mounted on '/'.  Reboot when ready to switch to this updated BE.
```

現在の BE は変更されません。すべての変更は新しい BE で行われます。

新しい BE を明示的に指定することは、インストールまたは更新のもっとも安全な方法です。BE が作成される状況については、122 ページの「ブート環境ポリシーイメージのプロパティー」を参照してください。--be-name オプションを使用すると、新しい BE にわかりやすい名前を付けることもできます。新しい BE がアクティブになるため、次回システムをブートしたときにはこの新しい環境がデフォルトでブートされます。次回のリブートで新規 BE をデフォルトにしないようにする場合、pkg update コマンドで --no-be-activate オプションを使用します。beadm activate コマンドを使用して、デフォルトブート BE をいつでも変更できます。ブート環境オプションの詳細については、44 ページの「ブート環境オプション」を参照してください。

新規 BE に問題がなければ、古い BE を破棄してかまいません。

ヒント - オペレーティングシステムのリリースごとに、以前の BE を保持しておいてください。必要に応じて以前の古い BE でブートし、古い BE を使用して、そのバージョンと、インストールした新しい次のバージョンの間のバージョンに更新することができます。

許可される最新バージョンより古いバージョンへの更新

場合によっては、許可される最新バージョンに更新したくないこともあります。このセクションでは、許可される最新バージョンより古いバージョンに更新する方法を説明します。

- 更新コマンドにバージョンを指定します。これは永続的な副作用のないもっとも単純な方法です。
- 更新コマンドを指定する前にバージョン制限を指定します。この方法では pkg freeze を使用するため、あとで新しいバージョンに更新する場合は必ず凍結を解除する必要があります。
- カスタムの制約パッケージを使用します。これはもっともスケーラブルで制御可能であり、追跡可能な方法です。

インストールするバージョンの指定

許可される最新バージョンより古いバージョンに更新するための簡単な方法は、`pkg update` コマンドでパッケージ名 (バージョン文字列の部分を含む) を指定することです。次の例は、新しいバージョンが許可される場合でも、Oracle Solaris 11 11/11 SRU 13 に更新するために `pkg:/entire` 制約パッケージのバージョンを指定する方法を示しています。

```
$ pkg update -nv entire@0.5.11,5.11-0.175.0.13 '*'
            Packages to remove:        2
           Packages to install:        1
            Packages to update:      486
     Estimated space available: 48.39 GB
Estimated space to be consumed:  2.50 GB
       Create boot environment:      Yes
     Activate boot environment:      Yes
Create backup boot environment:       No
          Rebuild boot archive:      Yes

Changed packages:
solaris
...
  entire
    0.5.11,5.11-0.175.0.10.0.5.0:20120803T182627Z -> 0.5.11,5.11-0.175.0.13.0.4.0:
20121106T194623Z
...
```

実際の更新を実行する前に、必ず `-nv` オプションを使用し、出力を確認してください。実際の更新を実行するときは、`--be-name` オプションを使用して、新しい BE にわかりやすい名前を付けます。

一部のインストール済みパッケージは、`entire` 制約パッケージによって制約されたパッケージに依存しない場合もあります。これらのパッケージは、`entire` パッケージだけを更新しても更新されません。これらのパッケージを同じ `pkg update` コマンドに名前で追加するか、`entire@`*version* に加えて `*` を指定できます。

注記 - コマンド行でバージョン制約を指定することは、多数のシステムを管理するための最適な方法ではありません。最適なスケーラブルなソリューションは、78 ページの「Oracle Solaris 制約パッケージの使用」および79 ページの「カスタム制約パッケージのインストール」の説明に従って制約パッケージを使用することです。

更新前のバージョン制約の指定

任意の Oracle Solaris 11.2 バージョンへの更新は許可するが、Oracle Solaris 11.3 への更新を許可しない場合は、次のコマンドに示すように `pkg:/entire` 制約パッケージを凍結できます。`0.175.2` を指定すると、`entire` パッケージを `0.175.3` にではなく、たとえば `0.175.2.15` に更新できることを示します。

```
$ pkg freeze -c "Keep this image at 11.2." entire@0.5.11,5.11-0.175.2
entire was frozen at 0.5.11,5.11-0.175.2
$ pkg freeze
NAME    VERSION             DATE                      COMMENT
entire  0.5.11,5.11-0.175.2 30 Jan 2014 15:50:01 PST Keep this image at 11.2.
$ pkg list entire
NAME (PUBLISHER)  VERSION                       IFO
entire            0.5.11,5.11-0.175.2.10.0.5.0  if-
```

パッケージの凍結の詳細は、113 ページの「指定したバージョンへのパッケージのロック」を参照してください。

Oracle Solaris 11.2 を越えて更新するには、`pkg unfreeze entire` を実行するか、`entire` をより新しいバージョンで凍結する必要があります。

注記 - パッケージを凍結することは多数のシステムを管理するための最適な方法ではありません。最適なスケーラブルなソリューションは、78 ページの「Oracle Solaris 制約パッケージの使用」および 79 ページの「カスタム制約パッケージのインストール」の説明に従って制約パッケージを使用することです。

Oracle Solaris 制約パッケージの使用

前のセクションで示した `pkg freeze` コマンドを使用するのと同じように、インストール可能な `pkg:/entire` のバージョンを制約するパッケージをインストールできます。制約パッケージと、Oracle Solaris でのその使用についての詳細は、14 ページの「制約パッケージ」を参照してください。

Oracle Solaris 11.3 には、`pkg:/entire` を Oracle Solaris 11.3 の任意のバージョンに制約するパッケージが用意されています。

```
$ pkg list -s solaris-11.3
NAME (PUBLISHER)                SUMMARY
release/constraint/solaris-11.3  Constraint Package for Oracle Solaris 11.3
```

このパッケージの内容は、このパッケージをインストールした場合、システムを Oracle Solaris 11.3 SRU など任意の Oracle Solaris 11.3 リリース (`entire@0.5.11-0.175.3`) には更新できるが、より新しい Oracle Solaris 11.*n* リリースには更新できないことを示しています。

```
$ pkg contents -m solaris-11.3
set name=pkg.fmri value=pkg://solaris/release/constraint/solaris-11.3@0,5.11:
20161021T222558Z
set name=pkg.summary value="Constraint Package for Oracle Solaris 11.3"
set name=variant.arch value=sparc value=i386
set name=pkg.depend.install-hold value=core-os
depend fmri=entire@0.5.11-0.175.3 type=incorporate
depend fmri=entire type=require
```

- `solaris-11.3` パッケージをインストールしたあとで、入手可能な最新の Oracle Solaris 11.3 リリースがすでにインストールされているときに更新を試みると、使用可能な更新が存在しないというメッセージを受け取ります。

- `solaris-11.3` パッケージをインストールしたあとで、より新しい Oracle Solaris 11.*n* リリースへの更新を試みると、システムが過度に制約されているというメッセージを受け取ります。より新しい Oracle Solaris 11.*n* リリースに更新するには、まず `solaris-11.3` 制約パッケージをアンインストールする必要があります。

カスタム制約パッケージのインストール

前のセクションで示した Oracle Solaris 制約パッケージを使用するのと同じように、独自のカスタム制約パッケージを作成して、必要な制約を指定できます。Oracle Solaris パッケージ内で使用可能な制約とは異なる制約を指定する場合もあります。

- 制約パッケージを作成し、ローカルの IPS パッケージリポジトリまたはパッケージアーカイブファイルからパッケージをインストールします。
- 制約を変更するには、カスタム制約パッケージを変更および再提供し、`pkg update` を使用して、新しい制約パッケージをインストールします。

カスタム制約パッケージを使用して、インストール可能なソフトウェアのバージョンを制御することにより、複数のパッケージリポジトリを維持しなくても、異なるシステム上でさまざまなバージョンの Oracle Solaris を容易に維持できます。各システムに、異なるバージョンのカスタム更新制御パッケージをインストールできます。それぞれのシステムで必要なすべてのソフトウェアバージョンを格納する同一のパッケージリポジトリが、すべてのシステムで共有されます。

カスタム制約パッケージの作成

イメージにインストール可能なコアのオペレーティングシステムパッケージのバージョンは、`pkg:/entire` 制約パッケージによって制御されます。システムのアップグレードを制御するには、特定バージョンの `entire` パッケージを `incorporate` 依存関係として指定するパッケージを作成します。

カスタム制約パッケージマニフェストの作成

次の例は、インストール可能な `pkg:/entire` パッケージのバージョンを制御するカスタム制約パッケージに対する `upgradectrl.p5m` という名前のマニフェストを示します。このマニフェストの設定の一部を以下に示します。

```
set name=pkg.fmri value=upgradectrl@1.0
set name=pkg.summary value="Incorporation to constrain the version of the OS"
set name=pkg.description value="This package controls the version of \
pkg://solaris/entire that can be installed."
set name=info.classification value="org.opensolaris.category.2008:Meta Packages/
Incorporations"
```

```
set name=pkg.depend.install-hold value=core-os
set name=variant.opensolaris.zone value=global value=nonglobal
set name=variant.arch value=sparc value=i386
depend fmri=feature/package/dependency/self type=parent variant.opensolaris.zone=nonglobal
depend fmri=pkg://solaris/entire type=require
depend fmri=pkg://solaris/entire@0.5.11,5.11-0.175.1.0 type=incorporate
```

`pkg.depend.install-hold`

ユーザーが `pkg update upgradectrl` コマンドを入力すると、`pkg:/entire` パッケージも自動的に更新されます。

`variant.opensolaris.zone`

このパッケージは、大域ゾーンと非大域ゾーンの両方にインストールできます。`parent` 依存関係の説明も参照してください。

`variant.arch`

このパッケージは、SPARC と x86 の両方のシステムにインストールできます。

`parent` 依存関係

このパッケージは、大域ゾーンにすでにインストール済みの場合に限り、非大域ゾーンにインストールできます。

`require` 依存関係

`upgradectrl` パッケージは、`pkg://solaris/entire` パッケージがすでにインストールされているか、これと同じ操作でインストール可能である場合に限りインストールできます。

`incorporate` 依存関係

`pkg://solaris/entire` パッケージを指定のバージョンでインストールする必要があります。指定された精度の桁数によっては、複数のバージョンが `incorporate` 依存関係を満たす場合があります。この例では、`0.175.1.0` は Oracle Solaris 11.1 SRU 0 を指定します。このアップグレード制御パッケージは、サポート更新のない Oracle Solaris 11.1 にシステムを維持します。ただし、このアップグレード制御パッケージは、`pkg:/entire` 制約パッケージによる制約を受けないパッケージの更新を許可します。

アップグレード制御パッケージの公開

`upgradectrl` パッケージをファイルベースのローカルリポジトリに公開します。このリポジトリは、この新しいパッケージを開発およびテストするためのものです。一般的な用途のリポジトリを作成する場合、リポジトリ用の別個のファイルシステムを作成するなどの追加のステップを含める必要があります。一般的な用途のパッケージリポジトリを作成することについては、『Copying and Creating Package Repositories in Oracle Solaris 11.3』を参照してください。

システム上にパッケージ開発リポジトリを作成します。pkgrepo コマンドについては、pkgrepo(1) のマニュアルページを参照してください。

```
$ pkgrepo create myrepo
```

このリポジトリのデフォルトパブリッシャーを設定します。デフォルトパブリッシャーは、リポジトリの publisher/prefix プロパティーの値です。

```
$ pkgrepo -s myrepo set publisher/prefix=site
```

upgradectrl パッケージを開発リポジトリに公開します。

```
$ pkgsend -s myrepo publish upgradectrl.p5m
pkg://site/upgradectrl@1.0,5.11:20131104T072336Z
PUBLISHED
```

リポジトリのデフォルトパブリッシャーが、パッケージ FMRI に適用されていることに注意してください。

リポジトリを調べて、パッケージが公開されたことを確認します。

```
$ pkgrepo -s myrepo list
PUBLISHER NAME                    O VERSION
site      upgradectrl               1.0,5.11:20131104T072336Z
$ pkg list -vg myrepo
FMRI                                                IFO
pkg://site/upgradectrl@1.0,5.11:20131104T072336Z    ---
```

O 列の値は、パッケージが廃止されている (o) か、または名前が変更されている (r) かを示します。

パッケージを、共有されている場所にある別個の ZFS ファイルシステム内のローカルリポジトリに提供します。

```
$ pkgrecv -s myrepo -d /var/share/pkgrepos/solaris upgradectrl
Processing packages for publisher site ...
Retrieving and evaluating 1 package(s) ...
PROCESS      ITEMS    GET (MB)    SEND (MB)
Completed      1/1     0.0/0.0      0.0/0.0
```

リポジトリ内のパッケージと、パッケージが incorporation で指定する pkg:/entire のバージョンを確認します。

```
$ pkg info -g /var/share/pkgrepos/solaris upgradectrl
          Name: upgradectrl
       Summary: Incorporation to constrain the version of the OS
   Description: This package controls the version of pkg://solaris/entire that
                can be installed.
      Category: Meta Packages/Incorporations
         State: Not installed
     Publisher: site
       Version: 1.0
 Build Release: 5.11
        Branch: None
Packaging Date: November 20, 2013 01:01:05 AM
          Size: 0.00 B
          FMRI: pkg://site/upgradectrl@1.0,5.11:20131120T010105Z
```

```
$ pkg contents -Hro fmri -t depend -a type=incorporate upgradectrl
pkg://solaris/entire@0.5.11,5.11-0.175.1.0
```

IPS パッケージの作成と提供についての詳細は、「Creating and Publishing a Package」in 『Packaging and Delivering Software With the Image Packaging System in Oracle Solaris 11.3』を参照してください。

パブリッシャーの起点の設定

site パブリッシャーの起点を設定します。システムリポジトリはこの情報で自動的に更新されるため、非大域ゾーンでは site パブリッシャーからパッケージにアクセスできるようになります。

```
$ pkg set-publisher -g /var/share/pkgrepos/solaris site
$ pkg publisher
PUBLISHER              TYPE     STATUS P LOCATION
solaris                origin   online F https://pkg.oracle.com/solaris/support/
site                   origin   online F file:///var/share/pkgrepos/solaris/
```

アップグレード制御パッケージのインストール

パッケージをインストールします。このケースでは、インストール済みの pkg:/entire のバージョンが、アップグレード制御パッケージによって incorporation 指定されたバージョンと同じであるため、変更はほとんどありません。パッケージは非大域ゾーンにもインストールされることに注意してください。

```
$ pkg list -v entire
FMRI                                                                            IFO
pkg://solaris/entire@0.5.11,5.11-0.175.1.0.0.24.2:20120919T190135Z              i--
$ zoneadm list
global
z1
$ pkg install upgradectrl
           Packages to install:  1
       Create boot environment: No
Create backup boot environment: No

Planning linked: 0/1 done; 1 working: zone:z1
Planning linked: 1/1 done
Downloading linked: 0/1 done; 1 working: zone:z1
Downloading linked: 1/1 done
PHASE                                          ITEMS
Installing new actions                           9/9
Updating package state database                 Done
Updating image state                            Done
Creating fast lookup database                   Done
Reading search index                            Done
Updating search index                            1/1
Executing linked: 0/1 done; 1 working: zone:z1
Executing linked: 1/1 done
```

次のコマンドは、インストール済みのバージョンよりも新しいバージョンの pkg:/entire が、構成済みの solaris パブリッシャーから入手できますが、アップグレー

ドの試行は新しくインストールされたアップグレード制御パッケージによって制御されていることを示しています。

```
$ pkg list -af entire
NAME (PUBLISHER)                                      VERSION                    IFO
entire                                                0.5.11-0.175.1.13.0.6.0    ---
entire                                                0.5.11-0.175.1.12.0.5.0    ---
entire                                                0.5.11-0.175.1.11.0.4.0    ---
entire                                                0.5.11-0.175.1.10.0.6.0    ---
entire                                                0.5.11-0.175.1.10.0.5.0    ---
...
$ pkg update
pkg update: No solution was found to satisfy constraints
Plan Creation: Package solver has not found a solution to update to latest available
 versions.
This may indicate an overly constrained set of packages are installed.
latest incorporations:
...
Try specifying expected results to obtain more detailed error messages.
$ pkg update -nv entire@0.5.11-0.175.1.13.0.6.0
pkg update: No matching version of entire can be installed:
  Reject:  pkg://solaris/entire@0.5.11,5.11-0.175.1.13.0.6.0:20131108T211557Z
  Reason:  This version is excluded by installed incorporation pkg://site/upgradectrl@1.
0,5.11:20131120T010105Z
```

アップグレード制御パッケージの更新

システムを新しいバージョンに更新することをユーザーに許可する準備が整ったら、`upgradectrl.p5m` マニフェストを更新し、新しい更新制御パッケージを再発行して再提供します。次のマニフェストでは、更新制御パッケージのバージョンと、`pkg:/entire` 制約パッケージのバージョンが更新されています。ユーザーへの支援として、アップグレード制御パッケージのバージョン `1.10` は、`entire` パッケージの更新済みバージョン `0.175.1.10` に対応するように設定されます。

```
set name=pkg.fmri value=upgradectrl@1.10
set name=pkg.summary value="Incorporation to constrain the version of the OS"
set name=pkg.description value="This package controls the version of \
pkg://solaris/entire that can be installed."
set name=info.classification value="org.opensolaris.category.2008:Meta Packages/
Incorporations"
set name=pkg.depend.install-hold value=core-os
set name=variant.opensolaris.zone value=global value=nonglobal
set name=variant.arch value=sparc value=i386
depend fmri=feature/package/dependency/self type=parent variant.opensolaris.zone=nonglobal
depend fmri=pkg://solaris/entire type=require
depend fmri=pkg://solaris/entire@0.5.11,5.11-0.175.1.10 type=incorporate
```

次のコマンドにより、更新制御パッケージが再発行および再提供されます。

```
$ pkgsend -s myrepo publish upgradectrl.p5m
pkg://site/upgradectrl@1.10,5.11:20131120T021902Z
PUBLISHED
$ pkgrepo -s myrepo list
PUBLISHER NAME                                        O VERSION
site      upgradectrl                                   1.10,5.11:20131120T021902Z
site      upgradectrl                                   1.0,5.11:20131120T010105Z
$ pkgrecv -s myrepo -d /var/share/pkgrepos/solaris upgradectrl
```

```
Processing packages for publisher site ...
Retrieving and evaluating 1 package(s)...
PROCESS                                           ITEMS    GET (MB)   SEND (MB)
Completed                                           1/1     0.0/0.0     0.0/0.0
$ pkg refresh site
$ pkg list -af pkg://site/upgradectrl
NAME (PUBLISHER)                                    VERSION                    IFO
upgradectrl (site)                                  1.10                       ---
upgradectrl (site)                                  1.0                        i--
```

イメージのアップグレード

次の pkg update コマンドではパッケージが指定されていないため、すべてのパッケージは入手可能で許可されている最新バージョンに更新されます。コマンドにより、アップグレード制御パッケージは入手可能な最新バージョンに更新され、これによってイメージがアップグレードされますが、この理由は、upgradectrl パッケージの pkg.depend.install-hold 設定により、upgradectrl パッケージが更新されるときに pkg:/entire パッケージが更新されるためです。イメージは、新しい upgradectrl 制約パッケージで指定されている pkg:/entire 制約パッケージのバージョンにアップグレードされます。

```
$ pkg update --be-name s11u1_10
            Packages to remove:   1
            Packages to update: 186
           Mediators to change:   1
       Create boot environment: Yes
Create backup boot environment:  No

Planning linked: 0/1 done; 1 working: zone:z1
Linked image 'zone:z1' output:
|  Packages to remove:  1
| Packages to install:  3
|  Packages to update: 73
| Mediators to change:  1
|  Services to change:  3
`
Planning linked: 1/1 done
DOWNLOAD                                  PKGS         FILES    XFER (MB)   SPEED
Completed                                 187/187   16139/16139  507.9/507.9  562k/s

Downloading linked: 0/1 done; 1 working: zone:z1
Downloading linked: 1/1 done
PHASE                                          ITEMS
Removing old actions                       1473/1473
Installing new actions                     3451/3451
Updating modified actions                 16378/16378
Updating package state database                Done
Updating package cache                       187/187
Updating image state                           Done
Creating fast lookup database                  Done
Reading search index                           Done
Building new search index                    851/851
Executing linked: 0/1 done; 1 working: zone:z1
Executing linked: 1/1 done

A clone of s11u1_0 exists and has been updated and activated.
On the next boot the Boot Environment s11u1_10 will be
mounted on '/'.  Reboot when ready to switch to this updated BE.
```

現在の BE が変更されていないこと、および新しい BE に更新済みパッケージが含まれていることを確認します。

```
$ pkg list entire upgradectrl
NAME (PUBLISHER)                                  VERSION                    IFO
entire                                            0.5.11-0.175.1.0.0.24.2    i--
upgradectrl (site)                                1.0                        i--
$ beadm mount s11u1_10 /mnt
$ pkg -R /mnt list entire upgradectrl
NAME (PUBLISHER)                                  VERSION                    IFO
entire                                            0.5.11-0.175.1.10.0.6.0    i--
upgradectrl (site)                                1.10                       i--
$ beadm unmount s11u1_10
```

イメージのダウングレード

オペレーティングシステムのリリースをダウングレードするには、ダウングレードするバージョンより古い BE にブートし、そこからアップグレードします。たとえば、Oracle Solaris 11.3 SRU 10 から Oracle Solaris 11.3 SRU 13 に更新したあとで、SRU 12 イメージが必要なことがわかった場合は、SRU 10 BE にリブートし、そこから SRU 12 に更新します。

あるいは、カーネルゾーンまたは Oracle VM Server for SPARC をインストールします。

サポート更新の適用

Oracle Solaris サポートリポジトリは、セキュリティー更新を含む重要な修正で随時更新されます。Oracle Solaris サポートリポジトリについては、86 ページの「サポート更新へのアクセス」を参照してください。

次のタイプの更新が提供されています。これらのさまざまなタイプの更新によって、システムのコンプライアンスと稼働時間の要件に最適なシステム保守の方法を選択できます。これらのオプションはいつでも簡単に切り替えることができます。

- Support Repository Updates (SRU)
- クリティカルパッチアップデート (CPU SRU)

Oracle Solaris サポートリポジトリは、セキュリティー更新を含む重要な修正で毎月更新されます。これらの毎月の更新を SRU と言います。3 回に 1 回の SRU は CPU SRU です。ほかの SRU では拡張機能や重要でないバグ修正が導入されることがありますが、CPU SRU の新しい内容は、共通脆弱性 (CVE) の修正など、重要な修正に限定されます。

システムを毎月更新できない場合は、CPU のリリーススケジュールに基づき毎四半期ごとに更新します。CPU のリリーススケジュールについては、Oracle Technology Network で Critical Patch Updates を参照してください。これらの更新パスは、新しい SRU または CPU に更新することでいつでも切り替えることができます。「Oracle Solaris バイナリおよびソース保証プログラム」(ドキュメント ID 1391762.1) は、リリースの境界を越える更新のリスクが低いことを保証します。

サポート更新へのアクセス

サポート更新を適用するには、次のいずれかのソースからシステムを更新します。

- `https://pkg.oracle.com/solaris/support/` で入手できる Oracle Solaris サポートリポジトリ。サポートリポジトリにアクセスするには、Oracle サポート資格を使用して、`https://pkg-register.oracle.com/` Oracle Solaris パッケージリポジトリの証明書要求サイトに SSL 証明書を作成します。
- 次のいずれかのソースから更新するローカルリポジトリ。
  - Oracle Solaris サポートリポジトリ。
  - My Oracle Support からダウンロードされた SRU リポジトリファイル。

    リポジトリファイルをダウンロードするには、Oracle サポートサイトにある Oracle Solaris 11.3 サポートリポジトリ更新 (SRU) 索引 (ドキュメント ID 2045311.1) を参照してください。各 SRU の Readme ファイルには、この SRU で修正されるバグ、更新されるパッケージ、および置換される IDR (Interim Diagnostic or Relief) 更新のリストが記載されています。IDR 更新の説明については、89 ページの「IDR カスタムソフトウェア更新のインストール」を参照してください。SRU のインストールガイドには、SRU Readme ファイルのコピー、SRU パッケージリポジトリファイルをインストールする方法について説明している別個の readme ファイル、チェックサムファイル、および SRU リポジトリファイルをローカルパッケージリポジトリにインストールするスクリプトが含まれています。リポジトリダウンロードには、SRU リポジトリファイルが含まれています。

    修正は SRU ごとに累積されます。ただし、各 SRU には、その更新で変更されたパッケージのみが含まれています。したがって、ローカルリポジトリに追加した SRU にのみシステムを更新できます。4 月の SRU を追加できなかった場合に、5 月の SRU を追加すると、5 月、4 月、およびそれより前の SRU のすべての修正を入手したことになりますが、システムを 4 月の SRU レベルには更新できません。

  ローカルの IPS パッケージリポジトリを作成および保持する方法については、『Copying and Creating Package Repositories in Oracle Solaris 11.3』を参照してください。

更新を実行するには、オペランドなしで `pkg update` コマンドを使用します。最新の SRU リリースよりも古い SRU に更新するには、76 ページの「許可される最新

バージョンより古いバージョンへの更新」で説明されている方法のいずれかを使用します。

クリティカルパッチ更新パッケージ

次のクリティカルパッチ更新パッケージが各月次 SRU で提供されます。

`pkg:/support/critical-patch-update/solaris-11-cpu@`*YYYY*`.`*MM*`-`*version*

YYYY	SRU がリリースされた年。
MM	SRU がリリースされた月。この値は 1 桁または 2 桁です。先頭のゼロは使用されません。
version	CPU パッケージが同じ月に再度リリースされたときに増分される整数。

この `solaris-11-cpu` パッケージには次の 2 つの目的があります。

- SRU で提供される CVE バグ修正についてメタデータを提供します。
- システムを、指定された SRU レベルに更新します。その制約パッケージからロック解除されるコンポーネントも含め、すべてのコンポーネントが指定された SRU レベルに移動されます。

次のコマンドは、このシステムにインストールされているすべての CVE 修正を一覧表示します。

```
$ pkg search -Hlo value info.cve:
```

追加の修正が使用可能かどうかを確認するには、次のコマンドを使用して、インストールしたバージョンより新しいバージョンの `solaris-11-cpu` パッケージが使用可能かどうかを表示します。

```
$ pkg list -n solaris-11-cpu
```

新しいパッケージが使用可能な場合は、次のコマンドを使用して、新しいパッケージから使用できる CVE 修正を一覧表示し、そのリストをインストール済みの CVE 修正のリストと比較します。

```
$ pkg contents -ro value -t set -a name=info.cve solaris-11-cpu@YYYY.MM
```

`pkg update` コマンドを使用して、新しい修正をインストールし、最新の SRU に更新します。

次のコマンドは、指定された CVE の修正を提供する `solaris-11-cpu` パッケージのすべてのバージョンを表示します。

```
$ pkg search -Hpo pkg.shortfmri CVE-YYYY-NNNN:
```

この出力は、この CVE の修正が最初に提供された `solaris-11-cpu` パッケージのバージョンと、この修正が最後に提供されたバージョンを示します。たとえば、10 月は 9 月より古いようにソートされるため、これらのパッケージは必ずしも日付順に一覧表示されません。

特定の CVE 識別子については、次のコマンドは、その CVE を修正するために変更されたすべてのパッケージを一覧表示します。

```
$ pkg search -Ho value CVE-YYYY-NNNN:
```

CVE の詳細については、『Oracle Solaris 11.3 Security Compliance Guide』を参照してください。

SPARC システムのプラットフォームファームウェアの更新

選択された SPARC システムのプラットフォームファームウェアの更新は、My Oracle Support (`http://support.oracle.com`) からの `.zip` のダウンロードに加えて、Oracle Solaris IPS サポートリポジトリで入手できます。`solaris` パッケージパブリッシャーが Oracle Solaris サポートリポジトリからパッケージを取得するように構成されている場合は、`.zip` ファイルをダウンロードして展開する代わりに、`pkg install` コマンドを使用できます。更新にはシステム管理者による手動の手順とサーバーの電源再投入が必要であるため、IPS パッケージをインストールしてもサーバーのファームウェアは自動的には更新されません。ファームウェア更新パッケージをインストールまたは更新すると、ファイルが `/var/firmware/`*ServerType* に配信されます。ファームウェアの更新をインストールするには、含まれる README ファイルの手順に従います。

次のコマンドを使用して、使用しているプラットフォームのファームウェア更新パッケージを識別します。

```
$ pkg list -af 'firmware/system/*'
```

パッケージには、`pkg:/firmware/system/`*ServerType* という名前が付けられています。

`pkg contents` および `pkg info` コマンドを使用して、プラットフォームのパッケージに関する詳細情報を取得します。たとえば、`firmware/system/T5-4` グループパッケージは、`firmware/system/T5-4/sysfw9-5`、`firmware/system/T5-4/hbafw`、および `firmware/system/T5-4/hwprogrammables` パッケージをインストールします。`pkg info` コマンドは、`firmware/system/T5-4/sysfw9-5` パッケージが Version 9.5.3 Patch 22270913 を提供することを示します。この例では、`/var/firmware/system/T5-4/sysfw9-5/p22270913_953/README.html` の手順に従って、ファームウェアの更新をインストールします。

プラットフォームの `pkg:/firmware/system/`*ServerType* パッケージをインストールしたあと、`pkg update` コマンドを使用して、新しいファームウェアの更新をダウン

ロードできます。最新のファームウェア更新パッケージをダウンロードして、最新の SRU もインストールするには、`pkg update` コマンドを FMRI なしで使用します。

ファームウェアの更新の詳細は、『*Oracle ILOM* 構成および保守用管理者ガイド、ファームウェアリリース *3.2.x*』を参照してください。

IDR カスタムソフトウェア更新のインストール

IDR (Interim Diagnostic or Relief) 更新は、特定の問題に関するカスタムの一時的な回避方法や診断を提供します。IDR はパッチではなく、一般にすべてのお客様が使用できるものではありません。IDR は、最終修正を提供できるようになるまで、問題の暫定的な回避方法を提供することがあります。また、IDR は、特定の問題に関連する診断データを収集することがあります。IDR によって問題が軽減されている場合でも、永続的な解決方法が使用可能になったらすぐに、永続的な解決方法に移る必要があります。IDR は SRU では提供されません。ただし、問題の永続的な修正が、後続の SRU で提供されることはあります。SRU については、85 ページの「サポート更新の適用」を参照してください。

IDR によって対処された問題が更新で修正されていない場合、IDR はシステムの更新を防ぎ、IDR が削除されないようにします。たとえば、IDR が `system/network` パッケージを変更し、SRU が `system/network` パッケージを更新する場合、IDR によって対処された問題に対する修正が SRU で提供されていないと、更新は失敗します。

IDR によって対処された問題に対する修正が SRU で提供されている場合、その SRU にはその IDR の置換用パッケージも含まれており、IDR は更新中に自動的に削除されます。置換用 IDR パッケージにより、明示的に IDR を削除せずに SRU に更新できます。

IDR は `p5p` という拡張子を持つパッケージアーカイブファイルとして配布されます。パッケージアーカイブは、パブリッシャーの情報と、そのパブリッシャーによって提供された 1 つ以上のパッケージを含むファイルです。

IDR のインストール

IDR をインストールする前に、IDR によって対処される問題や IDR で提供されている内容など、その IDR に関する情報を取得します。

例 10 IDR パッケージアーカイブに関する情報を取得する

次のコマンドは、アーカイブには、`solaris` パブリッシャーによって発行された 3 つのパッケージが含まれていることを示しています。

```
$ pkgrepo info -s idr1929.2.p5p
PUBLISHER PACKAGES STATUS           UPDATED
solaris   3        online           2015-06-24T09:01:20Z
```

`pkgrepo list` および `pkg list` コマンドは、パッケージアーカイブ内のパッケージに関する同様の、ただし異なる情報を提供します。インストールまたは更新するパッケージの名前は `idr1929` です。パッケージアーカイブの名前は、アーカイブが IDR 1929 のバージョン 2 であることを示しています。次の出力は、例15「IDR の名前が変更されているかどうかを表示する」 に説明するように、`idr1834` が名前変更されていることを示しています。

```
$ pkgrepo list -s idr1929.2.p5p
PUBLISHER NAME                            O VERSION
solaris   idr1834                         r 2,5.11:20150624T090119Z
solaris   idr1929                           2,5.11:20150624T090120Z
solaris   system/install/media/internal     0.5.11,5.11-0.175.2.9.0.3.2.1929.2:
20150624T090120Z
$ pkg list -afg idr1929.2.p5p
NAME (PUBLISHER)                            VERSION                        IFO
idr1834                                     2                              --r
idr1929                                     2                              ---
system/install/media/internal               0.5.11-0.175.2.9.0.3.2.1929.2 ---
```

例 11 この IDR で提供されている内容を表示する

次のコマンドは、`idr1929` パッケージに `system/install/media/internal` パッケージのカスタムバージョンが組み込まれていることを示しています。`system/install/media/internal` パッケージのバージョンはこの IDR を指定しています。バージョン文字列の枝番の部分の末尾にある `1929.2` フィールドを見てください。パッケージのバージョン文字列のフィールドについては、16 ページの「障害管理リソース識別子」を参照してください。例10「IDR パッケージアーカイブに関する情報を取得する」のコマンドは、IDR パッケージアーカイブには、IDR パッケージで提供されているこのカスタムバージョンのパッケージが含まれていることを示しました。

```
$ pkg contents -g idr1929.2.p5p -o type,fmri -t depend idr1929
TYPE        FMRI
incorporate pkg:/system/install/media/internal@0.5.11,5.11-0.175.2.9.0.3.2.1929.2
```

次のコマンドは、このカスタムバージョンの `system/install/media/internal` パッケージは `idr1929@2` パッケージに対して `require` 依存関係を持っていることを示しています。つまり、このバージョンの `system/install/media/internal` はこの IDR がインストールされている場合にのみインストールされます。

```
$ pkg contents -g idr1929.2.p5p -o type,fmri -t depend media/internal
TYPE    FMRI
require idr1929@2
```

例 12 IDR によって対処される問題を表示する

次のコマンドは、`idr1929` が対処している問題レポートの番号を表示します。複数の問題レポートが表示されることもあります。

```
$ pkg contents -Hg idr1929.2.p5p -o value -t set -a name='*bug*' idr1929
```

```
16857802
```

例 13 IDR をインストールできる Oracle Solaris リリースを判定する

次のコマンドは、この IDR がどの Oracle Solaris リリース向けに構築されたかを表示します。

```
$ pkg contents -Hg idr1929.2.p5p -o value -t set -a name="*description*" idr1929
i386 IDR built for release : Solaris 11.2 SRU # 10.5.0
```

この IDR は、構築された対象のリリースだけでなく、ほかの Oracle Solaris リリースにもインストールできる場合があります。例11「この IDR で提供されている内容を表示する」に示すように、この IDR で提供されているすべてのパッケージのバージョンを一覧表示し、そのリストを、それらのパッケージの使用可能なすべてのバージョンのリストと比較します。IDR で提供されているものと同じバージョンのパッケージをインストールできる、任意の Oracle Solaris リリース上に IDR をインストールできます。この例では、`idr1929` は、バージョン `5.11-0.175.2.9.0.3.2` の `system/install/media/internal` パッケージをインストールできる任意の Oracle Solaris リリース上にインストールできます。

例 14 IDR のリリースノートの場所を表示する

次のコマンドは、インストールされている IDR のリリースノートがある場所を表示します。

```
$ pkg contents -Ht file idr1929
usr/share/doc/release-notes/idr1929.txt
```

また、`pkg history` コマンドを使用してリリースノートを表示することもできます。130 ページの「操作履歴の表示」の説明に従って、`-n`、`-t`、および `-N` オプションを使用してください。

IDR がインストールされていない場合は、92 ページの「IDR をインストールする方法」に示すように、`-nv` オプションを指定して `pkg install` コマンドを使用することで、IDR をインストールせずにリリースノートを表示できます。

例 15 IDR の名前が変更されているかどうかを表示する

例10「IDR パッケージアーカイブに関する情報を取得する」の `pkgrepo list` および `pkg list` コマンドは、`idr1834` パッケージの名前が変更されていることを示しています。次のコマンドは、`idr1834` が `idr1929` に名前変更されていることを示しています。明示的に `idr1834` をインストールすると、`idr1929` が代わりにインストールされ、`idr1834` がすでにインストールされている場合は削除されます。

```
$ pkg info -g idr1929.2.p5p idr1834
          Name: idr1834
       Summary: Superseding pkg for idr1834.1
         State: Not installed (Renamed)
    Renamed to: pkg://solaris/idr1929@2,5.11
```

```
     Publisher: solaris
       Version: 2
 Build Release: 5.11
        Branch: None
Packaging Date: June 24, 2015 09:01:19 AM
          Size: 5.46 kB
          FMRI: pkg://solaris/idr1834@2,5.11:20150624T090119Z
```

次のコマンドは、`idr1834` パッケージの内容は `idr1929` パッケージに対する依存関係のみであることを示しています。`signature` アクションは簡潔にするために省略されています。

```
$ pkg contents -mg idr1929.2.p5p idr1834
set name=pkg.fmri value=pkg://solaris/idr1834@2,5.11:20150624T090119Z
set name=pkg.summary value="Superseding pkg for idr1834.1"
set name=pkg.renamed value=true
depend fmri=pkg://solaris/idr1929@2,5.11 type=require
```

IDR パッケージでは、名前の変更を使用することにより、インストール済みの IDR を管理者が明示的に削除しなくても、その IDR がインストールされているシステムを新しい SRU バージョンに更新できるようになっています。

▼ IDR をインストールする方法

1. この同じ問題に対処するほかの IDR がインストールされていないことを確認します。

次のコマンドは、インストールする IDR によって対処されるすべての問題の識別子を一覧表示します。

```
$ pkg contents -g idr1929.2.p5p -Ho value -t set -a name='*bug*' idr1929
['problemID1, 'problemID2']
```

次のコマンドは、これらの同じ問題に対処するすべてのインストール済みパッケージを一覧表示します。

```
$ pkg search -lo value,pkg.name problemID1 OR problemID2
```

インストールする IDR によって対処される問題が、現在インストール済みの IDR でも対処されている場合は、この同じ問題に対処するインストール済みの IDR が、インストールする IDR によって削除されるかどうかを確認してください。新しい IDR をインストールすることによって古い IDR が削除されるかどうかを確認するには、`pkg list` および `pkg info` コマンドを使用するか、`-nv` オプションを指定して `pkg install` コマンドを使用します。両方の IDR がインストールされることになる場合は、サポート担当者に連絡してください。

2. パッケージアーカイブをパブリッシャーの起点として追加します。

この手順は、非大域ゾーンがこのリポジトリにアクセスできるようにするために必要です。このシステムに非大域ゾーンが含まれていない場合でも、構成済みのパブリッシャーの起点にこれらのパッケージを追加することがベストプラクティスです。

```
$ pkg set-publisher -g idr1929.2.p5p solaris
```

```
$ pkg publisher
PUBLISHER   TYPE     STATUS P LOCATION
solaris     origin   online F file:///var/share/repos/idr1929.2.p5p/
solaris     origin   online F https://pkg.oracle.com/solaris/support/
```

使用しているすべての IDR パッケージのローカルリポジトリを作成することをお勧めします。『Copying and Creating Package Repositories in Oracle Solaris 11.3』を参照してください。`pkg5srv` ユーザーがリポジトリの内容にアクセスできることを確認します。非大域ゾーンがリポジトリの内容にアクセスできることを確認します。

3. **この IDR がこのシステムにインストール可能であることを確認します。**
次の `pkg contents` コマンドは、IDR にバージョン `5.11-0.175.2.9.0.3.2.1929.2` の `media/internal` パッケージが組み込まれていることを示しています。

```
$ pkg contents -ro type,fmri -t depend idr1929
TYPE        FMRI
incorporate pkg:/system/install/media/internal@0.5.11,5.11-0.175.2.9.0.3.2.1929.2
```

バージョン `5.11-0.175.2.9.0.3.2` の `media/internal` パッケージがシステムにインストールされている場合は、この IDR をインストールできます。
次のコマンドは、何がインストールされるかを示しますが、実際にはインストールを実行しません。このイメージに非大域ゾーンがある場合は、`-r` オプションを指定します。インストール操作では、IDR のリリースノートが表示されます。リリースノートには、IDR によって対処されるバグ、IDR パッケージによってインストールされるパッケージ、および IDR の削除手順が記載されています。

```
$ pkg install -nvr idr1929
           Packages to install:         1
     Estimated space available:  31.42 GB
Estimated space to be consumed: 122.34 MB
       Create boot environment:        No
Create backup boot environment:        No
          Rebuild boot archive:        No

Changed packages:
solaris
  idr1929
    None -> 2,5.11:20150624T090120Z
Release Notes:
  # Release Notes for IDR : idr1929
  # ------------------------------

  Release         : Solaris 11.2 SRU # 10.5.0

  Platform         : i386

  Bug(s) addressed     :
      16857802 : AI zlib download timeout should be longer

  Package(s) included     :
      pkg:/system/install/media/internal

  Removal instruction     :
  # /usr/bin/pkg uninstall -r idr1929
  If this IDR is installed on a system running Solaris 11.2 SRU9 or less, use
  this command to remove:
  # /usr/bin/pkg update --ignore-missing -r --reject
pkg:/system/install/media/internal@0.5.11,5.11-0.175.2.9.0.3.2:20150321T014312Z
```

```
Generic Instructions     :

1) If system is configured with 'Zones', ensure that
IDR is available in a configured repository.

Special Instructions for : idr1929

None.
```

4. **IDR をインストールします。**

   前のコマンドからの出力は、`idr1929` パッケージがインストールされることを示しています。あるバージョンの `idr1834` がシステムにすでにインストールされている場合、`idr1929` のインストールはブロックされることに注意してください。その場合は、`idr1929` をインストールする代わりに `idr1834` を更新する必要があります。`idr1834` の更新の結果として、例16「置換用 IDR をインストールする」に示すように `idr1929` がインストールされます。

   IDR をインストールするときは、新しい BE またはバックアップ BE が作成されることを確認してください。IDR をインストールしたあとで問題が発生した場合は、古い BE またはバックアップ BE にリブートできます。

   前のコマンドの出力は、このインストールでは新しい BE もバックアップ BE も作成されないことを示しています。次のコマンドは、現在の BE に IDR をインストールする前に、バックアップ BE を作成します。

```
$ pkg install --backup-be-name pre-idr1929 idr1929
```

置換用 IDR とサポート更新のインストール

IDR は、同じ問題に対処する新しい IDR、または問題を修正する SRU で置換できます。IDR パッケージの内容に関する情報を取得し、`-nv` オプションを使用して、`pkg update` コマンドまたは `pkg install` コマンドの結果がどのようになるかを理解します。

- 置換用 IDR。IDR でほかの IDR を置換できます。通常、既存の IDR は新しい修正によって更新されるため、置換用 IDR は必要ありません。場合によっては、既存の IDR の変更は実行できず、以前の IDR を置換する新しい IDR が作成されることがあります。

  置換用 IDR が必要になるのは、すでに別の IDR によって変更されているパッケージを変更するような場合です。たとえば、IDR1 は、`system/core-os` パッケージの一部である `usr/sbin/cron` に対する修正を提供します。IDR2 は、ほかのパッケージに対する無関係な修正を提供し、`system/core-os` の一部である `usr/sbin/svcadm` に対する修正を含んでいます。IDR1 および IDR2 の両方が `system/core-os` パッケージを変更することはできません。置換用 IDR パッケージが必要になり、それは次のいずれかの形式を取ることができます。

  - IDR2 に IDR1 の修正が含まれています。IDR2 は IDR1 を置換します。IDR1 の名前は IDR2 に変更されます。そのため、明示的に IDR1 パッケージをインストールまたは更新すると、IDR2 パッケージが代わりにインストールされ、IDR1 パッケージがシステムにすでにインストールされていた場合は削除されます。
  - IDR3 に IDR1 の修正と IDR2 の修正が含まれています。IDR3 は IDR1 と IDR2 の両方を置換します。
- 置換用 SRU。IDR によって対処された問題が SRU で修正されている場合は、その IDR パッケージの名前が、修正が行われたパッケージの名前に変更されます。IDR パッケージの変更後の名前に等しいバージョンのパッケージをインストールする SRU の場合、その SRU に更新すると IDR パッケージは削除されます。

  修正されたパッケージのより新しいバージョンを含んではいるが問題を修正しない SRU の場合、その SRU への更新はブロックされます。更新を実行するには、IDR をアンインストールするか、または IDR パッケージを拒否します。ただし、これらの方法を使用して IDR を削除すると、システムには IDR で対処されていた問題が残ることに注意してください。

例 16 置換用 IDR をインストールする

次の出力は、このイメージに `idr1834` のバージョン 1 がインストールされていることを示しています。`idr1834` のバージョン 2 は、92 ページの「IDR をインストールする方法」で `solaris` パブリッシャーの起点として追加された `idr1929.2.p5p` から使用できます。

```
$ pkg list -af 'idr*'
NAME (PUBLISHER)      VERSION IFO
idr1834               2       --r
idr1834               1       i--
idr1929               2       ---
```

例15「IDR の名前が変更されているかどうかを表示する」は、バージョン 2 では `idr1834` パッケージの名前が `idr1929` に変更されていることを示しています。`idr1834` を更新すると、インストールされているバージョン 1 は削除され、`idr1929` がインストールされます。更新後は、`idr1834` パッケージのどのバージョンもインストールされていません。

```
$ pkg update -v idr1834
            Packages to remove:         1
           Packages to install:         1
     Estimated space available:  30.42 GB
Estimated space to be consumed: 718.51 MB
       Create boot environment:        No
Create backup boot environment:        No
          Rebuild boot archive:        No

Changed packages:
solaris
  idr1834
    1,5.11:20150708T000258Z -> None
  idr1929
    None -> 2,5.11:20150624T090120Z
```

```
...

$ pkg list -af 'idr*'
NAME (PUBLISHER)      VERSION IFO
idr1834               2       --r
idr1834               1       ---
idr1929               2       i--
```

`idr1834` を更新する代わりに `idr1929` のインストールを試みると、`idr1834` が引き続きシステム上のパッケージを制約して `idr1929` が移動するバージョンより低いバージョンに保つため、インストール操作は失敗します。

例 17 IDR の問題を修正する SRU に更新する

次のコマンドは、インストール済みの IDR によって対処されたすべての問題を一覧表示します。

```
$ pkg contents -Ho value,pkg.name -t set -a name='*bug*' 'idr*'
```

SRU で修正された問題を調べるには、SRU の Readme ファイルを参照してください。インストール済みの IDR によって対処されたすべての問題が SRU で修正されている場合は、その SRU に更新できます。構成済みのパブリッシャーから入手可能な最新の SRU に更新するには、制約を受けない `pkg update` コマンドを使用します。インストール済みの IDR によって対処されたすべての問題が修正されている中間 SRU に更新するには、76 ページの「許可される最新バージョンより古いバージョンへの更新」を参照してください。

修正された IDR は更新中に自動的に削除されます。

例 18 IDR の問題を修正しない SRU に更新する

IDR が変更しているどのパッケージも SRU で更新されない場合、`pkg update` コマンドは SRU への更新を行い、IDR はインストールされたままになります。

IDR が変更しているいずれかのパッケージが SRU で更新されるが、IDR が対処している問題が修正されない場合、更新操作は失敗します。IDR で提供されている修正が失われてもよい場合は、次のコマンドを使用して IDR を削除し、更新を続行することもできます。

```
$ pkg update --reject idr1929
```

IDR によって提供されている修正を失いたくない場合は、サポート担当者に連絡し、その SRU リビジョンの新しい IDR をリクエストしてください。既存の IDR を置換する新しい IDR が生成されるため、修正を失わずにシステムを新しい SRU および IDR に更新できます。

入手可能な最新の SRU ではない SRU に更新する場合は、更新で置換用 IDR パッケージを指定する必要があります。その方法の 1 つを次の例に示します。

```
$ pkg update entire@intermediate-version  superseding-idr  '*'
```

IDR の削除

ほとんどの場合、IDR で対処されていた問題に対してシステムが無防備になるため、IDR を明示的に削除するべきではありません。IDR の削除は、IDR で対処された問題に対する修正を含んでいる SRU に更新するプロセスの一環として行われる必要があります。

IDR を明示的に削除できる 1 つの事例は、IDR が診断作業のみを実行し、その診断情報を収集する必要がなくなったときです。

▼ IDR を削除する方法

1. **IDR パッケージのサマリーにパッケージの削除手順があるかどうかを確認します。**

   IDR パッケージのサマリーには、このパッケージを削除する方法の説明が含まれています。

   古い IDR の場合は、2 とおりの手順が指定されています。1 つのコマンドは、Oracle Solaris 11.2 SRU 10 以上を実行しているシステム用で、もう 1 つのコマンドは、SRU 10 よりも前の Oracle Solaris 11.2 リリースを実行しているシステム用です。

   `info`、`contents`、および `search` サブコマンドのどれでも、パッケージのサマリーを表示できます。

   - **`pkg info`**

     ```
     $ pkg info idr1929
        Name: idr1929
     Summary: To back out This IDR : # /usr/bin/pkg uninstall -r idr1929;
     on Solaris systems < 11.2 SRU 10 : # /usr/bin/pkg update --ignore-missing -r
     --reject idr1929 pkg:/system/install/media/
     internal@0.5.11,5.11-0.175.2.9.0.3.2:20150321T014312Z
     ...
     ```

   - **`pkg contents`**

     ```
     $ pkg contents -Ho value -a name=pkg.summary idr1929
     ```

   - **`pkg search`**

     ```
     $ pkg search -Hlo value idr1929::pkg.summary:
     ```

   新しい IDR の場合は、1 とおりの手順が指定されています。

   ```
   $ pkg search -Hlo value idr2353::pkg.summary:
   To back out This IDR : # /usr/bin/pkg uninstall -r idr2353
   ```

   例14「IDR のリリースノートの場所を表示する」に示すコマンドを使用して、インストールされているリリースノートファイル内のこの情報を表示することもできます。

2. **適切なコマンドを実行して IDR パッケージを削除します。**

- **Oracle Solaris 11.2 SRU 10 以上。**

  Oracle Solaris 11.3 を含む、Oracle Solaris 11.2 SRU 10 以上を実行しているシステムでは、次の例に示すように `pkg uninstall` コマンドを使用します。

  ```
  $ pkg uninstall -r idr1929
  ```

  `-r` オプションは、この大域ゾーンに関連付けられたすべての非大域ゾーン内で削除を実行します。

- **古い Oracle Solaris 11.2 システム。**

  SRU 10 よりも前の Oracle Solaris 11.2 リリースを実行しているシステムでは、次の例に示すように `pkg update` コマンドを使用します。

  ```
  $ pkg update -r --ignore-missing \
  --reject pkg:/system/install/media/internal@0.5.11,5.11-0.175.2.9.0.3.2:20150321T014312Z
  ```

  `-r` オプションは、この大域ゾーンに関連付けられたすべての非大域ゾーン内で削除を実行します。

  `--ignore-missing` オプションは、更新されるパッケージの一部がインストールされていない場合に `update` 操作を失敗させないことを意味します。

  `--reject` オプションは、IDR パッケージによってインストールされたパッケージを削除します。

♦ ♦ ♦ 第 5 章

5 インストールされるイメージの構成

この章では、パッケージパブリッシャーの構成、インストール可能なパッケージの制限、パッケージ署名ポリシーの設定、ブート環境 (BE) ポリシーの構成などのイメージ全体に適用する特性を構成する方法について説明します。

- 起点、検索順序、鍵と証明書、およびプロキシの設定などのパブリッシャーの構成
- バリアントおよびファセットの設定によるオプションコンポーネントのインストールの制御
- 指定したバージョンへのパッケージのロック
- 制約パッケージによって指定されたバージョン制約の緩和
- メディエーションの使用によるアプリケーションのデフォルト実装の指定
- グループパッケージに含まれる一部のパッケージのインストールの回避
- BE 作成ポリシーやパッケージ署名ポリシーなどのイメージプロパティーおよびパブリッシャープロパティーの構成
- イメージの作成
- パッケージ操作履歴の表示

これらの操作の多くでは、強力な権限が必要です。詳細は、21 ページの「インストール権限」を参照してください。

この章で説明されているコマンドのすべてのオプションの完全なリストについては、pkg(1) のマニュアルページを参照してください。

パブリッシャーの構成

ソフトウェアをインストールして更新するには、pkg クライアントがパッケージリポジトリに接続できる必要があります。

Ops Center を使用している場合は、139 ページの「Oracle Enterprise Manager Ops Center のパブリッシャーの構成」を参照してください。

パブリッシャー情報の表示

このイメージに対して構成されているパッケージパブリッシャーに関する情報を表示するには、pkg publisher コマンドを使用します。パブリッシャーは、パッケージ FMRI にパブリッシャーが指定されていない場合に、パッケージを見つけるために検索される順番で一覧表示されます。

デフォルトで、solaris パブリッシャーは、新しくインストールされた Oracle Solaris 11 システムに構成されます。パブリッシャーの起点を確認するには、pkg publisher コマンドを使用します。

```
$ pkg publisher
PUBLISHER                  TYPE     STATUS P LOCATION
solaris                    origin   online F http://pkg.oracle.com/solaris/release/
isvpub         (non-sticky) origin   online F file:///var/share/pkgrepos/isvrepo/
devtool        (disabled)   origin   online F http://pkg.example1.com/
```

TYPE 列は、LOCATION 値が起点かミラーかを示します。詳細は、19 ページの「リポジトリの起点とミラー」を参照してください。

STATUS 列と LOCATION 列の間にある P 列は、その場所がプロキシ設定されているかどうかを指定します。この列の値は true (T) または false (F) です。ファイルリポジトリはプロキシ設定されません。値が T になっている HTTP リポジトリは、pkg set-publisher コマンドで起点が追加されたときに --proxy オプションで指定されたプロキシを使用してプロキシ設定されています。pkg publisher に -F tsv オプションを指定すると、その場所に設定されているプロキシが PROXY 列に表示されます。

```
$ pkg publisher -F tsv
PUBLISHER STICKY SYSPUB ENABLED TYPE   STATUS URI                                    PROXY
solaris   true   false  true    origin online http://pkg.oracle.com/solaris/release/ -
isvpub    false  false  true    origin online file:///var/share/pkgrepos/isvrepo/    -
devtool   true   false  false   origin online http://pkg.example1.com/              -
```

P 列の F あるいは PROXY 列の - は、その場所が pkg set-publisher コマンドを使用してプロキシ設定されなかったことを示します。その場所が、http_proxy 環境変数を設定することによってプロキシ設定されている場合、pkg publisher からの出力は、F または - を表示したままになります。プロキシを設定するためのさまざまな方法については、105 ページの「プロキシの指定」を参照してください。

それらのパブリッシャーの詳細構成を表示するには、名前でパブリッシャーを指定します。

```
$ pkg publisher solaris
       Publisher: solaris
           Alias:
      Origin URI: http://pkg.oracle.com/solaris/release/
         SSL Key: None
        SSL Cert: None
     Client UUID: e15e3228-eada-11df-80ab-8023183d954b
 Catalog Updated: March 4, 2014 11:48:02 PM
         Enabled: Yes
      Properties:
                  proxied-urls = []
```

クライアント UUID はこのイメージの一意識別子です。

パブリッシャーの検索順序の先頭のパブリッシャーのみを表示する場合は、-P オプションを使用します。

```
$ pkg publisher -P
PUBLISHER                    TYPE    STATUS P LOCATION
solaris                      origin  online F http://pkg.oracle.com/solaris/release/
```

有効になっているパブリッシャーのみを表示する場合は、-n オプションを使用します。

```
$ pkg publisher -n
PUBLISHER                    TYPE    STATUS P LOCATION
solaris                      origin  online F http://pkg.oracle.com/solaris/release/
isvpub         (non-sticky) origin  online F file:///var/share/pkgrepos/isvrepo/
```

パッケージパブリッシャーの追加、変更、削除

次の操作を実行するには、pkg set-publisher コマンドを使用します。

- 新しいパブリッシャーを構成する
- パブリッシャーの起点とミラーを設定する
- パブリッシャーのスティッキネスを設定する
- パブリッシャーの検索順序を設定する
- パブリッシャープロパティーを設定または設定解除したり、パブリッシャープロパティー値を追加または削除する
- パブリッシャーの SSL 鍵および証明書を指定する
- パブリッシャーのプロキシを設定する
- パブリッシャーを有効または無効にする
- パブリッシャーを削除する

pkg set-publisher コマンドには 2 つの形式があります。詳細は、pkg(1) のマニュアルページを参照してください。

- 1 つの形式では、パブリッシャーの名前が必須オペランドになります。
- 別の形式では、リポジトリ URI が -p オプションの引数として指定され、パブリッシャー情報はその指定されたリポジトリから取得されます。パブリッシャー名がオプションのオペランドになるため、複数のパブリッシャーがそのリポジトリにパッケージを公開する場合は、名前が付いたパブリッシャーのみ構成することができます。

パブリッシャーの追加

以下の例は、パブリッシャーを追加するための 2 つの方法を示します。

例 19 新しいパブリッシャーの指定

次のコマンドは、-g オプションで指定した起点 URI を持つ devtool という新しいパブリッシャーを追加し、このパブリッシャーが検索順序の先頭になるように設定します。指定したパブリッシャーを検索順序の先頭に設定するには、-P オプションまたは --search-first オプションを使用します。

```
$ pkg set-publisher -P -g http://pkg.example1.com/release/ devtool
```

例 20 パブリッシャー構成のインポート

指定したリポジトリ URI からパブリッシャーの構成情報を取得するには、-p オプションを使用します。パブリッシャーを指定した場合、一致するパブリッシャーのみが追加または更新されます。パブリッシャーを指定しない場合、すべてのパブリッシャーが必要に応じて追加または更新されます。

```
$ pkg publisher
PUBLISHER                   TYPE     STATUS P LOCATION
solaris                     origin   online F http://pkg.oracle.com/solaris/release/
$ pkg set-publisher -p /var/share/pkgrepos/myrepo
$ pkg publisher
PUBLISHER                   TYPE     STATUS P LOCATION
solaris                     origin   online F http://pkg.oracle.com/solaris/release/
site                        origin   online F file:///var/share/pkgrepos/myrepo/
```

パブリッシャーの起点の追加と変更

次のコマンドは、solaris パブリッシャーに起点を追加する方法を示しています。イメージ内の特定のパブリッシャーに対して複数の起点が構成されている場合、IPS クライアントは、パッケージデータの取得元として最適な起点を選択しようとします。たとえば、地理的に異なる場所にあるシステムに対してパブリッシャーの起点の同じセットを使用する場合、IPS クライアントは、その特定の時点で各システムに最適な起点を選択しようとします。

```
$ pkg publisher
PUBLISHER                   TYPE     STATUS P LOCATION
solaris                     origin   online F http://pkg.oracle.com/solaris/release/
$ pkg set-publisher -g /var/share/pkgrepos/solaris solaris
$ pkg publisher
PUBLISHER                   TYPE     STATUS P LOCATION
solaris                     origin   online F http://pkg.oracle.com/solaris/release/
solaris                     origin   online F file:///var/share/pkgrepos/solaris/
```

指定したパブリッシャーの起点として URI を削除するには、-G オプションを使用します。

パブリッシャーの起点 URI を変更するには、新しい URI を追加し、古い URI を削除します。

```
$ pkg set-publisher -G '*' -g file:///var/share/pkgrepos/isvrepo/ isvpub
```

パブリッシャーのミラーの追加と変更

指定したパブリッシャーのミラーとして URI を追加するには、-m オプションを使用します。起点とミラーの違いについては、19 ページの「リポジトリの起点とミラー」を参照してください。同じパッケージの同じバージョンが、同じパブリッシャーの起点リポジトリ内にも存在しないかぎり、ミラーリポジトリの内容にアクセスすることはできません。

```
$ pkg set-publisher -m http://pkg.example3.com/ devtool
$ pkg publisher
PUBLISHER                   TYPE    STATUS P LOCATION
devtool                     origin  online F http://pkg.example1.com/
devtool                     mirror  online F http://pkg.example3.com/
```

指定したパブリッシャーのミラーとして URI を削除するには、-M オプションを使用します。

パブリッシャーのミラー URI を変更するには、新しい URI を追加し、古い URI を削除します。

パブリッシャーの検索順序およびスティッキネスの設定

新しく追加されたパブリッシャーは、デフォルトで固定です。パブリッシャーが非固定の場合は、このパブリッシャーからインストールされたパッケージを別のパブリッシャーから更新できます。パブリッシャーのスティッキネスを設定するには、--sticky および --non-sticky オプションを使用します。

新しく追加されたパブリッシャーは、デフォルトで検索順序の最後になります。パブリッシャーの検索順序は、インストールするパッケージを検索するために使用されます。パッケージが最初にインストールされたときのパブリッシャーが非固定の場合は、更新するパッケージを検索するためにパブリッシャーの検索順序が使用されます。パブリッシャーの検索を変更するには、--search-before、--search-after、および --search-first オプションを使用します。-P オプションは、--search-first オプションの同義語です。

一致するパッケージを提供する最初のパブリッシャーがインストール元として使用されます。該当するパブリッシャーがこのイメージ内にインストール可能なパッケージのバージョンを提供しない場合、インストール操作は失敗します。この検索順序でさらに下にあるパブリッシャーからインストールするには、パブリッシャーの名前やパッケージのバージョン文字列などの詳細情報をパッケージ FMRI に指定します。

パブリッシャープロパティーの構成

パブリッシャープロパティーを設定および設定解除したり、パブリッシャープロパティーの値を追加および削除したりするには、次のオプションを使用します。

- `--set-property` *property=value*
- `--add-property-value` *property=value*
- `--remove-property-value` *property=value*
- `--unset-property` *property*

`publisher-search-order` および `signature-required-names` プロパティーは、複数の値を取ることができます。

125 ページの「パッケージの署名プロパティーの構成」の `pkg set-publisher` の例を参照してください。

パブリッシャーの鍵と証明書の構成

例 21 パブリッシャーの鍵と証明書の指定

クライアントの SSL 鍵を指定するには、`-k` オプションを使用します。クライアントの SSL 証明書を指定するには、`-c` オプションを使用します。このパブリッシャーに対する `pkg publisher` コマンドの出力に、鍵と証明書のハッシュが一覧表示されます。100 ページの「パブリッシャー情報の表示」および 140 ページの「SSL 証明書の問題」を参照してください。

```
$ pkg set-publisher -k /tmp/keyfile -c /tmp/certfile publisher-name
```

各パブリッシャーには 1 つだけの鍵と証明書を指定できます。パブリッシャーに複数のセキュアな起点が構成されている場合、すべてのセキュアな起点で 1 つの鍵と証明書を共有します。

例 22 パブリッシャーの鍵と証明書の取り消し

指定した証明書を取り消された証明書として扱うには、`--revoke-ca-cert` オプションを使用します。ユーザーが取り消した CA 証明書のハッシュが、このパブリッシャーに対する `pkg publisher` コマンドの出力に一覧表示されます。

承認された証明書の一覧および取り消された証明書の一覧から指定した証明書を削除するには、`--unset-ca-cert` オプションを使用します。

パブリッシャープロキシの構成

指定した起点またはミラーの内容を取得する永続的なプロキシ URI を指定するには、`--proxy` オプションを使用します。プロキシの値は *protocol*`://`*host-name*`[:`*port*`]` で、*protocol* は `http` または `https` で、`:`*port* はオプションです。プロキシを設定するためのさまざまな方法については、105 ページの「プロキシの指定」を参照してください。

パブリッシャーの有効化および無効化

新しく追加されたパブリッシャーは、デフォルトで有効になります。無効なパブリッシャーは、パッケージリストの作成時や `install`、`uninstall`、または `update` 更新パッケージ操作で使用されません。無効なパブリッシャーのプロパティーを設定または表示することはできます。1 つだけのパブリッシャーが有効な場合、そのパブリッシャーは無効にできません。

次のコマンドは、`isvpub` パブリッシャーを有効にし、それを検索順序で `devtool` パブリッシャーの前に設定します。

```
$ pkg set-publisher --enable --search-before devtool isvpub
```

パブリッシャーを無効にするには `--disable` オプションを使用します。パブリッシャーの起点が一時的に使用できないなどの場合にパブリッシャーを無効にする場合があります。すべてのパブリッシャーが使用できない場合、パッケージのインストールおよび更新操作は失敗します。

パブリッシャーの削除

パブリッシャーを削除するには、`pkg unset-publisher` コマンドを使用します。

```
$ pkg unset-publisher devtool
```

プロキシの指定

プロキシを設定する方式により、効果と利点は異なります。たとえば、`pkg set-publisher` コマンドは、パブリッシャー構成の一部としてプロキシを格納し、`http_proxy` 環境変数を使用すると、認証済みプロキシを設定できます。

プロキシを設定するための `pkg set-publisher` コマンドの設定

`pkg set-publisher` コマンドの `--proxy` オプションは、指定されたパブリッシャーの起点とミラー URI の永続的プロキシ URI を設定します。プロキシ値は、パブリッシャーの構成の一部として格納されます。パブリッシャーの構成の一部としてプロキシ値を格納することで、子イメージによって使用されるシステムリポジトリが自動的に更新されます。パブリッシャーの構成の一部としてプロキシ値を格納することは、異なるパブリッシャーに対して異なるプロキシを使用できるということも意味します。

```
$ pkg publisher
PUBLISHER                 TYPE     STATUS P LOCATION
solaris                   origin   online F file:///var/share/pkgrepos/solaris/
```

```
$ pkg publisher -F tsv
PUBLISHER STICKY SYSPUB ENABLED TYPE   STATUS URI                                    PROXY
solaris   true   false  true    origin online file:///var/share/pkgrepos/solaris/ -
$ pkg set-publisher -g http://pkg.oracle.com/solaris/release/ --proxy proxyURI solaris
$ pkg publisher solaris
      Publisher: solaris
          Alias:
     Origin URI: file:///var/share/pkgrepos/solaris/
        SSL Key: None
       SSL Cert: None
     Origin URI: http://pkg.oracle.com/solaris/release/
          Proxy: proxyURI
        SSL Key: None
       SSL Cert: None
    Client UUID: e15e3228-eada-11df-80ab-8023183d954b
Catalog Updated: July 11, 2013 11:32:46 PM
        Enabled: Yes
     Properties:
                  proxied-urls = []
$ pkg publisher
PUBLISHER                   TYPE    STATUS P LOCATION
solaris                     origin  online F file:///var/share/pkgrepos/solaris/
solaris                     origin  online T http://pkg.oracle.com/solaris/release/
$ pkg publisher -F tsv
PUBLISHER STICKY SYSPUB ENABLED TYPE   STATUS URI                                    PROXY
solaris   true   false  true    origin online file:///var/share/pkgrepos/solaris/ -
solaris   true   false  true    origin online http://pkg.oracle.com/solaris/release/ proxyURI
```

このイメージに非大域ゾーンがある場合、システムリポジトリはこのプロキシ情報を使用して自動的に更新されます。`system-repository` サービスにプロパティーを設定する必要はありません。非大域ゾーン内のパブリッシャープロキシを調べた場合、大域ゾーンに表示されるものと同じプロキシ URI は表示されません。大域ゾーンでは、システムリポジトリはプロキシ URI を使用します。非大域ゾーンでは、システムリポジトリはプロキシそのものとして機能し、非大域ゾーンが大域ゾーンのシステムリポジトリと通信できるようになります。64 ページの「大域ゾーンと非大域ゾーンの関係」に、非大域ゾーン内でシステムリポジトリパブリッシャーが表示された場合の例が示されています。

`pkg set-publisher` コマンドの `--proxy` オプションを使用して、認証済みポリシーを設定することはできません。`--proxy` オプションの値を、*protocol*`://`*user*`:`*password*`@`*host-name* の形式にすることはできません。

プロキシを設定するための環境変数の使用

プロキシ環境変数値は、そのプロトコルのすべての URI に適用されます。実行時は、`http_proxy` 環境変数の値が、`pkg set-publisher` コマンドの `--proxy` オプションで設定された値をオーバーライドします。プロキシ環境変数の詳細については、`curl`(1) のマニュアルページの「環境」セクションを参照してください。

非大域ゾーンを持つイメージ内で `http_proxy` 環境変数を設定した場合、`svc:/application/pkg/system-repository` SMF サービス内のプロキシプロパティーをこれらの同じ値に設定して、サービスをリフレッシュします。

```
$ svccfg -s system-repository:default setprop config/http_proxy = astring: proxyURI
$ svccfg -s system-repository:default listprop config/*proxy
config/https_proxy astring
config/http_proxy  astring     proxyURI
$ svcprop system-repository:default | grep proxy
config/https_proxy astring ""
config/http_proxy astring ""
$ svcadm refresh system-repository:default
$ svcprop system-repository:default | grep proxy
config/https_proxy astring ""
config/http_proxy astring proxyURI
```

`pkg publisher` コマンドは、環境変数または SMF サービスプロパティーを設定することによって設定されたプロキシを表示しません。

`http_proxy` 環境変数の値を変更した場合、`system-repository` サービスプロパティーを更新してサービスをリフレッシュするようにしてください。

オプションのコンポーネントのインストールの制御

ソフトウェアには、オプションのコンポーネントや、相互に排他的なコンポーネントが含まれることがあります。オプションのコンポーネントの例には、ロケールやドキュメントがあります。相互に排他的なコンポーネントの例には、SPARC または x86 アーキテクチャーが含まれます。オプションのコンポーネントはファセットと呼ばれ、相互に排他的なコンポーネントはバリアントと呼ばれます。

ファセットおよびバリアントの値がパッケージのインストールに与える影響

ファセットとバリアントのどちらも次の 2 つのコンポーネントで構成されます。

- イメージに設定されたプロパティー
- パッケージマニフェスト内のアクションに設定されたタグ

パッケージマニフェスト内のアクションに設定されたファセットおよびバリアントタグの値と、イメージに設定されたファセットおよびバリアントプロパティーの値の比較により、そのアクションがインストール可能かどうかを判別します。

ファセットとバリアントのプロパティーとタグにはそれぞれ名前と値があります。1 つのパッケージアクションに複数のファセットタグおよびバリアントタグを付けることができます。

ファセットとバリアントのプロパティーとタグの値は明示的に設定する必要はありません。値はデフォルト値または継承された値を指定できます。たとえば、非大域ゾー

ンはその親の大域ゾーンから値を継承できます。一部のバリアントプロパティー値は、システムの初期インストール時にイメージに設定され、変更できません。

次のアルゴリズムは、イメージに設定されたファセットおよびバリアントのプロパティー値が、特定のアクションがインストールされるかどうかにどのように影響するかを示しています。値については、以降のセクションで詳しく説明します。

- パッケージマニフェストにファセットタグまたはバリアントタグがないアクションは常にインストールされます。
- パッケージマニフェストにファセットタグがあるアクションは、イメージに次の条件が存在する場合にインストールされます。
  - アクションのファセットタグのうち、値 `all` を持つものはすべて、デフォルトまたは明示的な設定によって、イメージ内で `true` のファセットプロパティー値を持ちます。
  - アクションのファセットタグのいずれかが値 `true` を持つ場合、それらのファセットプロパティー値の少なくとも 1 つは、デフォルトまたは明示的な設定によってイメージ内でも `true` です。

  次の例で、ファイル `test.txt` は、イメージ内で `devel` と `optional.test` の両方が `true` であり、イメージ内で `doc.info` または `doc.help` のいずれかが `true` である場合にのみインストールされます。

  ```
  file path=usr/share/doc/test.txt facet.devel=all facet.optional.test=all
  facet.doc.info=true facet.doc.help=true
  ```

- パッケージマニフェスト内にバリアントタグのあるアクションは、すべてのバリアントタグの値がイメージに設定されている対応するバリアントプロパティーの値と同じ場合にのみインストールされます。

  次の例では、ファイル `x86test.txt` はインストールされません。次のパッケージマニフェストの抜粋は、ファイル `x86test.txt` に 2 つのバリアントタグが設定されていることを示しています。

  ```
  file path=usr/share/doc/x86test.txt variant.arch=i386 variant.debug.osnet=true
  ```

  次のコマンドは、イメージ内のこれらのバリアントプロパティーの値を示しています。

  ```
  $ pkg variant arch debug.osnet
  VARIANT                 VALUE
  arch                    i386
  debug.osnet             false
  ```

- ファセットタグとバリアントタグの両方があるアクションは、ファセットとバリアントの両方のプロパティー値でアクションのインストールが許可されている場合にインストールされます。

バリアント値の表示と変更

ほとんどのバリアントは説明的な値を持ちます。たとえば、variant.arch の値は、i386 や sparc などになります。アクションがインストールされるには、パッケージマニフェスト内のそのアクションのバリアントタグの値が、イメージ内の対応するバリアントプロパティーの値と一致している必要があります。arch および zone バリアントプロパティー値は、イメージを作成し、その初期コンテンツをインストールするプログラムによって設定されます。これらの値は変更できません。

variant.debug.* バリアントの値と不明なバリアントの値は true または false のみ指定でき、デフォルトでイメージ内で false です。不明なバリアントとは、パッケージマニフェスト内のアクションのバリアントタグとして導入され、イメージの作成時には不明だったバリアントです。

アクションのデバッグバージョンと非デバッグバージョンの両方を配布する場合、次の例に示すように、両方のバージョンに該当する debug バリアントタグが明示的に設定されている必要があります。この例では、debug.osnet がイメージ内で false の場合は、このファイルの非デバッグバージョンがインストールされ、debug.osnet がイメージ内で true の場合は、このファイルのデバッグバージョンがインストールされます。通常ファイルのデバッグバージョンは variant.arch タグも付けられます。

```
file payload chash=hash group=sys mode=0644 overlay=allow owner=root
path=etc/motd pkg.csize=116 pkg.size=106 preserve=true variant.debug.osnet=true
file payload chash=hash group=sys mode=0644 overlay=allow owner=root
path=etc/motd pkg.csize=70 pkg.size=50 preserve=true variant.debug.osnet=false
```

イメージに設定されているバリアントプロパティーの値を表示するには、pkg variant コマンドを使用します。

```
$ pkg variant
VARIANT                   VALUE
arch                      i386
opensolaris.zone          global
```

イメージに明示的に設定されたすべてのバリアントとインストール済みパッケージに設定されたすべてのバリアントの値を表示するには、-a オプションを使用します。

```
$ pkg variant -a
VARIANT                   VALUE
arch                      i386
debug.container           false
debug.osnet               false
opensolaris.zone          global
```

インストール済みパッケージに設定できるすべてのバリアント値を表示するには、-v オプションを使用します。

```
$ pkg variant -v
VARIANT                   VALUE
arch                      i386
arch                      sparc
debug.container           false
```

```
debug.container           true
debug.osnet               false
debug.osnet               true
opensolaris.zone          global
opensolaris.zone          nonglobal
```

バリアントプロパティーの値を変更するには、`pkg change-variant` コマンドを使用します。設定する値を選択するには、`pkg variant -v` コマンドを使用します。

注記 - バリアントプロパティー値を変更すると、多数のパッケージが更新され、新しい BE が必要になることがあります。変更する前に、どのような変更が行われるかを確認するには、`-nv` オプションを使用します。

次のコマンドでは、新しい BE が作成されます。BE が作成される状況については、122 ページの「ブート環境ポリシーイメージのプロパティー」を参照してください。新しい BE が作成されても、現在の BE は変更されません。この例ではデバッグファイルを使用するように、新しい BE でブートします。`-n` オプションを使用すると、この操作を `-n` なしで実行した場合に何が変更されるかが表示されますが、コマンドによって実際の変更は行われません。この例では新しい BE は作成されていません。

```
$ pkg change-variant -nv variant.debug.osnet=true
           Packages to change:       232
     Variants/Facets to change:         1
     Estimated space available: 306.74 GB
Estimated space to be consumed:   1.49 GB
       Create boot environment:       Yes
     Activate boot environment:       Yes
Create backup boot environment:        No
          Rebuild boot archive:       Yes

Changed variants/facets:
  variant debug.osnet: true

Changed packages:
solaris
  ...

Editable files to change:
  Update:
    etc/motd
```

ファセット値の表示と変更

ファセットプロパティーは、イメージにデフォルト値があり、明示的に設定する必要はありません。`facet.debug` または `facet.optional` で始まる名前を持つファセットプロパティーは、デフォルトで `false` の値を持ちます。ほかのすべてのファセットプロパティーは、デフォルトで `true` の値を持ちます。

パッケージマニフェスト内のアクションのファセットタグは、`true` または `all` の値のみ指定できます。説明と例については、107 ページの「ファセットおよびバリア

ントの値がパッケージのインストールに与える影響」を参照してください。`facet.debug` または `facet.optional` で始まる名前を持つファセットでタグ付けされたアクションは、デフォルトでインストールされません。そのようなアクションをインストールするには、イメージ内のファセットプロパティー値を `true` に変更する必要があります。ほかのすべてのファセットでタグ付けされたアクションは、デフォルトでインストールされます。そのようなアクションのインストールを避けるには、イメージ内のファセットプロパティー値を `false` に変更する必要があります。

version-lock ファセットの値 (`version-lock.`*package-name*) を変更しても、アクションはインストールまたはアンインストールされません。代わりに、version-lock ファセットは、`incorporate` タイプの `depend` アクションを有効または無効にします。147 ページの「制約パッケージによって制約されたパッケージの更新」を参照してください。

ファセットプロパティー値には、次の 3 つのいずれかのソースがあります。これらのソース名は、`pkg facet` コマンド出力の SRC 列に表示されます。

- `system` – システムによって割り当てられた値。これらの値は、通常システムパッケージに指定されたファセットタグの値です。
- `local` – `pkg change-facet` コマンドを使用するか、IPS API を使用して設定された値。
- `parent` – 親イメージから継承された値。たとえば、非大域ゾーンは大域ゾーンからのファセット設定を継承します。

このイメージにローカルで設定されたか、親イメージから継承されたすべてのファセットプロパティーの現在の値とその値のソースを表示するには、`pkg facet` コマンドを使用します。

イメージに明示的に設定されたすべてのファセットとインストール済みパッケージに設定されたすべてのファセットの値を表示するには、`pkg facet -a` コマンドを使用します。

次の `pkg facet` コマンドは、ローカルに設定されているか継承された `doc.` ファセットプロパティーがないことを示しています。

```
$ pkg facet doc.*
FACET                     VALUE SRC
pkg facet: no matching facets found
$ pkg facet -a doc.*
FACET                     VALUE SRC
doc.help                  True  system
doc.html                  True  system
doc.info                  True  system
doc.man                   True  system
doc.pdf                   True  system
doc.ps                    True  system
```

ファセットプロパティーの値を変更するには、`pkg change-facet` コマンドを使用します。ファセットプロパティー値には、`true`、`false`、または `none` を設定できます。ファセットプロパティーを `none` に設定すると、そのファセットにシステムデフォル

ト値 (true または false のいずれか) が適用され、ソースは system として表示されます。

注記 - ファセットプロパティー値を変更すると、多数のパッケージが更新され、新しい BE が必要になることがあります。変更する前に、どのような変更が行われるかを確認するには、-nv オプションを使用します。

イメージに設定されるファセットは、doc.man などの完全なファセットか、locale.* などのパターンになります。この柔軟性は、ファセット名前空間の一部を無効にし、その名前空間内の個々のファセットのみを有効にする場合に役立ちます。たとえば、次の例に示すように、すべてのロケールを無効にしてから、1 つか 2 つの特定のロケールのみを有効にすることができます。

```
$ pkg change-facet locale.*=false locale.en_US=true
```

次の pkg change-facet コマンドでは、新しい BE が作成されませんでしたが、バックアップ BE が作成されています。BE が作成される状況については、122 ページの「ブート環境ポリシーイメージのプロパティー」を参照してください。バックアップ BE が作成されると、元のイメージがバックアップに保存され、現在の BE が変更されます。「Changed variants/facets」の下の None の元の値は、前述の pkg facet コマンドによって示されるように、このコマンドの実行前にローカル設定が存在せず、システム設定しか存在していなかったことを示します。

```
$ pkg change-facet -v doc.*=false doc.man=true
            Packages to change:        97
     Variants/Facets to change:         2
            Services to change:         1
     Estimated space available: 306.34 GB
Estimated space to be consumed: 936.87 MB
       Create boot environment:        No
Create backup boot environment:       Yes
          Rebuild boot archive:        No

Changed variants/facets:
    facet doc.* (local): None -> False
    facet doc.man (local): None -> True

Changed packages:
solaris
  ...

Services:
  restart_fmri:
    svc:/application/texinfo-update:default

PHASE                                          ITEMS
...
```

次の pkg facet コマンドは、pkg change-facet コマンドの結果の doc. ファセットプロパティーの変更を示しています。

```
$ pkg facet doc.*
FACET                   VALUE SRC
doc.*                   False local
doc.man                 True  local
```

```
$ pkg facet -a doc.*
FACET                     VALUE SRC
doc.*                     False local
doc.help                  False local
doc.html                  False local
doc.info                  False local
doc.man                   True  local
doc.pdf                   False local
doc.ps                    False local
```

指定したバージョンへのパッケージのロック

パッケージのバージョンを制限する場合は、`pkg freeze` コマンドを使用します。

パッケージオペランドにバージョンを指定しない場合、名前付きのパッケージをインストールする必要があり、システムにインストールされているバージョンに制限されます。パッケージオペランドにバージョンを指定した場合、この制約 (つまり、凍結) は、`fmri` 属性が指定されたパッケージバージョンの値を持った状態で `incorporate` 依存関係がインストールされているかのように機能します。

凍結されているパッケージをインストールまたは更新するときは、凍結された時点のバージョンと一致するバージョンである必要があります。たとえば、パッケージが 1.2 で凍結された場合、1.2.1、1.2.9、1.2.0.0.1 などのバージョンに更新することはできます。そのパッケージは 1.3 または 1.1 で終了することはできません。

パッケージオペランドに指定されたパブリッシャーを使用して、一致するパッケージが検索されます。ただし、パブリッシャーの情報は凍結の一環として記録されません。パッケージはパブリッシャーではなくバージョンのみに関して凍結されます。

すでに凍結されているパッケージを凍結すると、新しく指定されたバージョンによって凍結バージョンが置き換えられます。

パッケージを指定しない場合、現在凍結されているパッケージについての情報 (パッケージ名、凍結バージョン、パッケージがいつ凍結されたか、パッケージが凍結された理由) が表示されます。

パッケージを凍結しても、そのパッケージを削除できなくなるわけではありません。パッケージが削除される場合に警告は表示されません。

次の例で、パッケージは現在インストールされているバージョンで凍結されます。`-c` オプション引数は、パッケージが凍結される理由です。凍結が原因でインストールまたは更新に失敗する場合、その理由が示されます。パッケージのリストの「f」は、そのパッケージが凍結されていることを示します。

```
$ pkg freeze -c "Downgrade to avoid bug" library/security/openssl
library/security/openssl was frozen at 1.0.0.10-0.175.1.0.0.18.0:20120611T201116Z
$ pkg freeze
NAME                    VERSION                                    DATE
  COMMENT
```

```
library/security/openssl 1.0.0.10-0.175.1.0.0.19.0:20120625T171753Z 29 Jul 2012 17:45:44
 PDT Downgrade to
avoid bug
$ pkg list library/security/openssl
NAME (PUBLISHER)                    VERSION                      IFO
library/security/openssl            1.0.0.10-0.175.1.0.0.18.0  if-
```

凍結されたパッケージの別のバージョンをインストールしようとすると、凍結に関するメッセージが表示されます。

```
$ pkg update library/security/openssl@1.0.0.10-0.175.1.0.0.20.0
Creating Plan (Solver setup):
pkg update: No matching version of library/security/openssl can be installed:
  Reject: pkg://solaris/library/security/openssl@1.0.0.10,5.11-0.175.1.0.0.20.0:
20120709T180243Z
  Reason:  This version is excluded by a freeze on library/security/openssl at version
  1.0.0.10,5.11-0.175.1.0.0.18.0:20120611T201116Z.
  The reason for the freeze is: Downgrade to avoid bug
```

凍結がパッケージシステムによって自動的に解除されることはありません。凍結によって適用される制約を、指定されたパッケージから削除するには、`pkg unfreeze` コマンドを使用します。バージョンを提供しても無視されます。

制約パッケージによって指定されたバージョン制約の緩和

制約パッケージは、インストール可能なパッケージのバージョンを指定します。これらのバージョン制約は、更新後もシステムをサポート可能な状態に保つのに役立ちます。制約パッケージとバージョン制約についての詳細は、14 ページの「制約パッケージ」を参照してください。

`incorporate` 依存関係である一部のパッケージは、制約パッケージの `incorporate` 依存関係で指定されたバージョンとは異なるバージョンでダウングレードやアップグレードを安全に行うことができる場合があります。バージョン制約は、制約パッケージ内の `depend` アクションに指定された `version-lock.`*package* ファセットタグによって表現されます。`version-lock.`*package* ファセットプロパティーのデフォルト値は `true` です。パッケージのバージョン制約を緩和するには、その `version-lock.` ファセットの値を `false` に設定します。

注記 - パッケージをロック解除すると、サポートされていない構成になる可能性があります。システムを最新に維持し、バージョンロックを常に `true` に設定しておくことがベストプラクティスです。85 ページの「サポート更新の適用」を参照してください。

次の例では、パッケージを以前のバージョンにダウングレードしようとしています。`pkg update` コマンドは、パッケージのアップグレードだけでなくダウングレードも行います。

```
$ pkg list -af library/security/openssl
```

```
NAME (PUBLISHER)                    VERSION                    IFO
library/security/openssl            1.0.1.5-0.175.2.0.0.24.0  i--
library/security/openssl            1.0.1.5-0.175.2.0.0.23.0  ---
$ pkg update library/security/openssl@1.0.1.5-0.175.2.0.0.23.0
Creating Plan (Solver setup):
pkg update: No matching version of library/security/openssl can be installed:
  Reject: pkg://solaris/library/security/openssl@1.0.1.5,5.11-0.175.2.0.0.23.0:
20130916T191702Z
  Reason:  This version is excluded by installed incorporation
  pkg://solaris/consolidation/userland/userland-incorporation@0.5.11,5.11-0.175.2.0.0.24.0:
20131001T160408Z
```

`pkg contents` コマンドは、このバージョン制約が設定されている方法を示します。このパッケージのバージョン制約を緩和するには、その `version-lock` ファセットを `false` に設定します。その後、ダウングレードをもう一度試みます。新しい BE は作成されませんが、バックアップ BE が作成されます。BE が作成される状況については、122 ページの「ブート環境ポリシーイメージのプロパティー」を参照してください。

```
$ pkg contents -m userland-incorporation | grep 'library/security/openssl'
depend facet.version-lock.library/security/openssl=true
fmri=pkg:/library/security/openssl@1.0.1.5-0.175.2.0.0.24.0 type=incorporate
$ pkg change-facet facet.version-lock.library/security/openssl=false
            Packages to update: 850
     Variants/Facets to change:   1
       Create boot environment:  No
Create backup boot environment: Yes

PHASE                                          ITEMS
Removing old actions                             1/1
Updating image state                            Done
Creating fast lookup database                   Done
Reading search index                            Done
Building new search index                     850/850
$ pkg update library/security/openssl@1.0.1.5-0.175.2.0.0.23.0
            Packages to update:   1
       Create boot environment:  No
Create backup boot environment: Yes

DOWNLOAD                                    PKGS       FILES    XFER (MB)   SPEED
Completed                                    1/1       10/10      1.6/1.6    0B/s

PHASE                                          ITEMS
Removing old actions                             3/3
Installing new actions                           3/3
Updating modified actions                       14/14
Updating package state database                 Done
Updating package cache                           1/1
Updating image state                            Done
Creating fast lookup database                   Done
Reading search index                            Done
Updating search index                            1/1
$ pkg list library/security/openssl
NAME (PUBLISHER)                    VERSION                    IFO
library/security/openssl            1.0.1.5-0.175.2.0.0.23.0  i--
```

このパッケージのダウングレードやアップグレードを防止するには、パッケージを現在のバージョンで凍結します。パッケージのリストの「f」は、そのパッケージが凍結されていることを示します。

```
$ pkg freeze -c "Downgrade to avoid bug" library/security/openssl
```

```
library/security/openssl was frozen at 1.0.1.5,5.11-0.175.2.0.0.23.0:20130916T191702Z
$ pkg list library/security/openssl
NAME (PUBLISHER)                  VERSION                    IFO
library/security/openssl          1.0.1.5-0.175.2.0.0.23.0  if-
```

アップグレードやダウングレードを再度有効にするには、`pkg unfreeze` コマンドを使用してバージョンの凍結を解除します。制約パッケージで指定されているバージョンより低いバージョンでパッケージがインストールされている場合、このパッケージの `version-lock` ファセットを `true` に設定すると、制約パッケージで指定されているバージョンがインストールされます。

ダウングレードまたはアップグレードしようとしているパッケージに対して、ほかのインストール済みパッケージが `require` 依存関係を持っている場合、そのような関連パッケージのバージョン制約の緩和も必要になることがあります。次の例では、`hexedit` パッケージのバージョン制約は解除されていますが、`system/library` パッケージのバージョン制約のためインストールは拒否されます。

```
$ pkg install editor/hexedit@1.2.12-0.175.2.0.0.25.0
Creating Plan (Solver setup):
pkg install: No matching version of editor/hexedit can be installed:
  Reject: pkg://solaris/editor/hexedit@1.2.12-0.175.2.0.0.25.0:20131014T170634Z
  Reason:  All versions matching 'require' dependency
  pkg:/system/library@0.5.11,5.11-0.175.2.0.0.24.0 are rejected
    Reject: pkg://solaris/system/library@0.5.11,5.11-0.175.2.0.0.24.0:20131001T152820Z

pkg://solaris/system/library@0.5.11,5.11-0.175.2.0.0.25.0:20131014T161136Z
    Reason:  This version is excluded by installed incorporation
    pkg://solaris/consolidation/osnet/osnet-incorporation@0.5.11,5.11-0.175.2.0.0.24.0:
20131001T150429Z
```

個々のコンポーネントパッケージのほかに、制約パッケージのバージョン制約も緩和できます。この場合、`version-lock` ファセットを `false` に設定すると、制約パッケージをシステムの残りの部分からロック解除できます。制約パッケージはロック解除されますが、`incorporate` 依存関係パッケージの同期は保たれます。

デフォルトのアプリケーション実装の指定

アプリケーションまたはツールの複数のバージョンが同じイメージから利用できる場合があります。ユーザーはフルパスを指定することによって、アプリケーションの各バージョンを利用できます。使いやすくするために、優先バージョンと呼ばれる 1 つのバージョンが、`/usr/bin` などの共通ディレクトリから利用できます。以下に示すように、すべてのバージョンが同じメディエーションに入っている場合、優先バージョンであるバージョンを簡単に設定し直すことができます。この管理上の選択は、パッケージの更新後も保持されます。

メディエーションとは、セット内のすべてのリンクでリンクパスが同じで、各リンクのターゲットが異なるリンクのセットです。たとえば、リンクパスが `/usr/bin/`

myapp で、リンクのターゲットが /usr/myapp/myapp1/bin/myapp および /usr/myapp/myapp2/bin/myapp を含む場合があります。メディエーションの各リンクは、メディエーションの参加者と呼ばれます。/usr/bin/myapp が現在 myapp1 を呼び出す場合、/usr/bin/myapp が myapp2 を呼び出すように選択を簡単に変更することができます。現在リンクのターゲットであるソフトウェアのバージョンは、優先バージョンです。

メディエーション内の参加者の識別

イメージ内のすべてのメディエーション対象リンクの優先バージョンを表示するには、pkg mediator コマンドを使用します。

次の出力で、MEDIATOR は、同じ優先リンクパスを共有するリンクのセットの名前です。VER. SRC. および IMPL. SRC. は、優先バージョンが、システムによって選択されたか、割り当て済みの優先度に応じて選択されたか (vendor または site)、管理者によって設定されたか (local) を表示します。VERSION は、選択されたメディエーション参加者のバージョンで、リンクが表すソフトウェアのバージョンに類似しているはずです。VERSION はパッケージ開発者によって設定されます。IMPLEMENTATION は、バージョンストリングに加えて、あるいはバージョンストリングの代わりにパッケージ開発者によって設定できるストリングです。

```
$ pkg mediator
MEDIATOR    VER. SRC. VERSION IMPL. SRC. IMPLEMENTATION
gcc-runtime system    4.7     system
java        system    1.7     system
php         system    5.2     system
python      vendor    2.6     vendor
ruby        system    1.9     system
```

-a オプションは、すべてのメディエーション参加者を表します。異なる優先バージョンを選択する場合、このオプションを使用して、選択内容を表示します。次の例は、java メディエーションのすべての参加者を示しています。system キーワードは、このメディエーションの優先バージョンがパッケージの優先設定で指定されておらず、管理者によって設定されなかったことを示します。パッケージシステムは、高い VERSION 値を持つバージョンを優先バージョンとして選択しました。

```
$ pkg mediator -a java
MEDIATOR    VER. SRC. VERSION IMPL. SRC. IMPLEMENTATION
java        system    1.7     system
java        system    1.6     system
```

次の出力では、2 つの異なるバージョンの Java Runtime Environment がこのイメージにインストールされ、バージョン 1.7.0_51 が現在選択されている優先バージョンであることを確認できます。

```
$ pkg list -s '*jre*'
NAME (PUBLISHER)        SUMMARY
```

```
runtime/java/jre-6      Java(TM) Platform Standard Edition Runtime Environment (1.6.0_71-
b12)
runtime/java/jre-7      Java Platform Standard Edition Runtime Environment (1.7.0_51-b13)
$ java -version
java version "1.7.0_51"
Java(TM) SE Runtime Environment (build 1.7.0_51-b13)
Java HotSpot(TM) Server VM (build 24.51-b03, mixed mode)
```

jre-6 パッケージと jre-7 パッケージの両方に、パスが /usr/bin/java であるシンボリックリンクが定義されています。jre-6 パッケージでは、リンクのターゲットは jdk1.6.0 です。jre-7 パッケージでは、リンクのターゲットは jdk1.7.0 です。上記の pkg mediator コマンドおよび java -version コマンドによって、現在の優先バージョンはバージョン 1.7 で、/usr/bin/java リンクのターゲットであることが表示されています。

優先アプリケーションの変更

指定したメディエーションのデフォルトまたは優先のバージョンを設定し直すには、pkg set-mediator コマンドを使用します。

pkg mediator -a からの出力を使用して、-V 引数に対するバージョンか、-I 引数に対する実装を選択します。タイプミスをしたか、現在入手できないメディエータのバージョンまたは実装を指定した場合、指定されたメディエータを使用するリンクは削除されます。

バックアップ BE が作成されるかどうかを表示するには、set-mediator サブコマンドに -n オプションを使用します。バックアップ BE が作成されない場合、set-mediator サブコマンドに --require-backup-be オプションを指定することができます。メディエータの変更が現在の BE で行われます。メディエータの変更後に現在の BE に問題がないことが確認できたら、beadm destroy を使用してバックアップ BE を破棄できます。

前回の出力では、現在選択されている java メディエーションの優先バージョンがバージョン 1.7 であると表示されました。次のコマンドは、バージョン 1.6 を優先バージョンとして設定することを示し、つまり /usr/bin/java を呼び出すと JRE バージョン 1.6 が呼び出されることを意味します。ユーザーが JRE version 1.7 のフルパスを指定すれば、このバージョンも引き続きシステムで使用可能です。2 つの pkg mediator コマンドの出力を比較してください。メディエーションの優先バージョンを変更すると、VER. SRC. も local に変更され、選択が管理者によって指定されたことを示しています。この選択は、リブートおよびパッケージ更新後も持続します。

```
$ pkg mediator java
MEDIATOR VER. SRC. VERSION IMPL. SRC. IMPLEMENTATION
java     system    1.7     system
$ pkg mediator -a java
MEDIATOR    VER. SRC. VERSION IMPL. SRC. IMPLEMENTATION
java        system    1.7     system
```

```
java        system    1.6     system
$ pkg set-mediator -V 1.6 java
            Packages to update:  3
           Mediators to change:  1
       Create boot environment: No
Create backup boot environment: No

PHASE                                          ITEMS
Removing old actions                             2/2
Updating modified actions                        3/3
Updating image state                            Done
Creating fast lookup database                   Done
Reading search index                            Done
Updating search index                            3/3
$ pkg mediator java
MEDIATOR VER. SRC. VERSION IMPL. SRC. IMPLEMENTATION
java     local     1.6     system
```

この管理上の選択は、選択された実装がインストールされていなくても、パッケージの更新後も保持されます。選択された実装がインストールされていない場合、メディエーションされたリンクのターゲットは存在しません。優先される実装をリセットするには、次のいずれかの方法を使用します。

- `pkg set-mediator` コマンドをふたたび使用して、`pkg mediator -a` によって表示された更新済みリストから別の実装を選択します。
- `pkg unset-mediator` コマンドを使用して、システムが新しい実装を選択できるようにします。

  ```
  $ pkg unset-mediator java
  ```

グループパッケージに含まれる一部のパッケージのインストールの回避

指定したパッケージが `group` 依存関係のターゲットである場合に、それらのインストールを回避するには、`pkg avoid` コマンドを使用します。回避リストに含まれているパッケージでも、イメージとの互換性があるパッケージはいつでも明示的にインストールできます。回避リストに含まれているパッケージを明示的にインストールすると、そのパッケージは回避リストから削除されます。`pkg avoid` コマンドを使用すると、グループパッケージをインストールするときに、そのグループパッケージに含まれている指定したパッケージのインストールを回避できます。

引数を使用しない場合、`pkg avoid` コマンドは、回避される各パッケージとそのパッケージへのグループ依存関係のあるすべてのパッケージを表示します。

パッケージを指定した場合、`pkg avoid` コマンドは、指定されたパターンに現在一致するパッケージ名を回避リストに配置します。現在インストールされていないパッケージのみを回避できます。パッケージが現在グループ依存関係のターゲットである

場合、パッケージをアンインストールするとそのパッケージは回避リストに登録されます。

回避リストに登録されているパッケージは、require 依存関係を満たすために必要であればインストールされ、したがって回避リストから削除されます。その require 依存関係が削除されると、パッケージはアンインストールされて回避リストにふたたび配置されます。

指定したパッケージを回避リストから明示的に削除する場合は、pkg unavoid コマンドを使用します。

回避リストに登録されており、インストール済みパッケージのグループ依存関係のターゲットであるパッケージは、pkg unavoid コマンドを使用して回避リストから削除することはできません。そのようなパッケージを回避リストから削除するには、そのパッケージをインストールします。

例 23 回避リストへのパッケージの追加と回避リストからのパッケージの削除

次のコマンド出力は、group/feature/amp グループパッケージがインストールされていないことを示しています。グループパッケージの一部であるパッケージの一部は、明示的に、またはほかのパッケージの require 依存関係としてインストールされたため、インストールされています。指定されたパッケージはインストールされていないため、pkg contents コマンドに -r オプションを使用します。

```
$ pkg list -a group/feature/amp
NAME (PUBLISHER)                                  VERSION                    IFO
group/feature/amp                                 0.5.11-0.175.2.0.0.33.0    ---
$ pkg list -a `pkg contents -o fmri -Hrt depend -a type=group group/feature/amp`
NAME (PUBLISHER)                                  VERSION                    IFO
database/mysql-51                                 5.1.37-0.175.2.0.0.34.0    ---
web/php-52                                        5.2.17-0.175.2.0.0.34.0    i--
web/php-52/extension/php-apc                      3.0.19-0.175.2.0.0.34.0    i--
web/php-52/extension/php-mysql                    5.2.17-0.175.2.0.0.34.0    i--
web/server/apache-22                              2.2.26-0.175.2.0.0.34.0    i--
web/server/apache-22/module/apache-dtrace         0.3.1-0.175.2.0.0.34.0     ---
web/server/apache-22/module/apache-fcgid          2.3.9-0.175.2.0.0.34.0     ---
web/server/apache-22/module/apache-php5           5.2.17-0.175.1.0.0.18      --r
```

次のコマンドは、まだインストールされておらずこのグループパッケージに属するパッケージの 1 つを回避リストに配置します。このグループパッケージはインストールされていないため、回避リストにグループパッケージは示されません。

```
$ pkg avoid apache-fcgid
$ pkg avoid
    web/server/apache-22/module/apache-fcgid
```

次のコマンドは、回避されたパッケージがグループパッケージのインストール時にインストールされないことを示しています。指定されたパッケージがインストール済みであるため、pkg contents コマンドで -r オプションは使用されません。

```
$ pkg install group/feature/amp
$ pkg list -a `pkg contents -o fmri -Ht depend -a type=group group/feature/amp`
NAME (PUBLISHER)                                  VERSION                    IFO
```

```
database/mysql-51                                 5.1.37-0.175.2.0.0.34.0    i--
web/php-52                                        5.2.17-0.175.2.0.0.34.0    i--
web/php-52/extension/php-apc                      3.0.19-0.175.2.0.0.34.0    i--
web/php-52/extension/php-mysql                    5.2.17-0.175.2.0.0.34.0    i--
web/server/apache-22                              2.2.26-0.175.2.0.0.34.0    i--
web/server/apache-22/module/apache-dtrace         0.3.1-0.175.2.0.0.34.0     i--
web/server/apache-22/module/apache-fcgid          2.3.9-0.175.2.0.0.34.0     ---
web/server/apache-22/module/apache-php5           5.2.17-0.175.1.0.0.18      i-r
```

グループパッケージのインストール後は、回避リストにグループパッケージが示されます。

```
$ pkg avoid
    web/server/apache-22/module/apache-fcgid (group dependency of 'group/feature/amp')
```

インストール済みのグループパッケージに含まれているパッケージは、`pkg unavoid` コマンドによって回避リストから削除されることはありません。そのようなパッケージを回避リストから削除するには、そのパッケージをインストールします。

```
$ pkg unavoid apache-fcgid
pkg unavoid: The following packages are a target of group dependencies; use install to
 unavoid these:
    web/server/apache-22/module/apache-fcgid
$ pkg install apache-fcgid
$ pkg avoid
$
```

パッケージがすでにインストールされている場合、そのパッケージを回避リストに登録することはできません。パッケージをアンインストールすると、パッケージが回避リストに登録されます。

```
$ pkg avoid apache-fcgid
pkg avoid: The following packages are already installed in this image; use uninstall to
 avoid these:
    web/server/apache-22/module/apache-fcgid
$ pkg uninstall apache-fcgid
$ pkg avoid
    web/server/apache-22/module/apache-fcgid (group dependency of 'group/feature/amp')
```

グループパッケージをアンインストールすると、回避されたパッケージは回避リストに残りますが、グループパッケージとの関連は回避リストに示されなくなります。

```
$ pkg uninstall group/feature/amp
$ pkg avoid
    database/mysql-51
    web/server/apache-22/module/apache-fcgid
$ pkg unavoid database/mysql-51 apache-fcgid
$ pkg avoid
$
```

イメージとパブリッシャーのプロパティーの構成

イメージポリシーを実装するには、イメージのプロパティーを設定します。このセクションでは、イメージとパブリッシャーのプロパティーと、これらのプロパティーの

設定方法について説明します。イメージプロパティーの説明については、pkg(1) のマニュアルページのイメージプロパティーに関するセクションも参照してください。

ブート環境ポリシーイメージのプロパティー

イメージは、IPS パッケージをインストールでき、その他の IPS 操作を実行できる場所です。ブート環境 (BE) は、イメージのブート可能なインスタンスです。システム上に複数の BE を維持することができ、各 BE にそれぞれ異なるソフトウェアバージョンをインストールすることもできます。システムをブートするとき、システム上の任意の BE にブートすることを選択できます。

パッケージ操作の結果として、新しい BE が自動的に作成されることがあります。明示的に新しい BE を作成することもできます。新しい BE が作成されるかどうかは、このセクションで説明するようにイメージポリシーに依存します。

デフォルトで、次のいずれかの操作を実行すると、新しい BE が自動的に作成されます。

- 一部のドライバやその他のカーネルコンポーネントなど、特定のキーシステムパッケージをインストールまたは更新する。重要なシステムコンポーネントは、バリアントやファセットを変更したり、パッケージをインストール、アンインストール、および更新するときに更新できます。
- `--be-name`、`--require-new-be`、`--backup-be-name`、`--require-backup-be` のいずれかのオプションを指定する。
- `be-policy` イメージポリシーを `always-new` に設定する。このポリシーでは、次のブート時にアクティブに設定されている新しい BE で、すべてのパッケージ操作が実行されます。

新しい BE が作成される場合、システムは次の処理を実行します。

1. 現在の BE のクローンを作成します。

   クローンの BE には、元の BE のメインのルートデータセットより下の階層にあるすべての要素が格納されます。共有ファイルシステムは、ルートデータセットの下にはなく、複製されません。代わりに、新しい BE は元の共有ファイルシステムにアクセスします。
2. クローン BE 内でパッケージを更新します。現在の BE 内でパッケージを更新しないでください。

   現在の BE で非大域ゾーンが構成されている場合、これらの既存のゾーンが新しい BE で構成されます。
3. `--no-be-activate` が指定されていないかぎり、システムの次回ブート時に、新しい BE をデフォルトのブート選択肢に設定します。現在の BE は代替のブート選択肢として残ります。

バックアップ BE が作成される場合、システムは次の処理を実行します。

1. 現在の BE のクローンを作成します。
2. 現在の BE 内でパッケージを更新します。クローン BE 内でパッケージを更新しないでください。

新しい BE が必要でも、それを作成するための十分な空き領域がない場合、既存の不要な BE を削除できる場合があります。BE の詳細は、『Creating and Administering Oracle Solaris 11.3 Boot Environments』を参照してください。

次のイメージプロパティーの設定方法については、128 ページの「イメージのプロパティーの設定」を参照してください。

`be-policy`

パッケージ操作中にいつ BE が作成されるかを指定します。次の値が許可されます。

`default`

デフォルトの BE 作成ポリシー `create-backup` を適用します。

`always-new`

次のブート時にアクティブに設定されている新しい BE でパッケージ操作を実行するため、すべてのパッケージ操作に対してリブートを必要とします。明示的に要求されないかぎり、バックアップ BE は作成されません。

このポリシーはもっとも安全ですが、リブートしないとパッケージを追加できないため、ほとんどのサイトの要求よりも厳格です。

`create-backup`

リブートを必要とするパッケージ操作の場合、このポリシーにより、次のブート時にアクティブに設定される新しい BE が作成されます。パッケージが変更されるか、カーネルに影響する可能性のあるコンテンツがインストールされて操作がライブ BE に影響する場合、バックアップ BE は作成されますがアクティブには設定されません。バックアップ BE を明示的に要求することもできます。

このポリシーは、新しくインストールされたソフトウェアによりシステムが不安定になっている場合にのみ潜在的に危険です。この可能性はありますが、比較的まれです。

`when-required`

リブートを必要とするパッケージ操作の場合、このポリシーにより、次のブート時にアクティブに設定される新しい BE が作成されます。明示的に要求されないかぎり、バックアップ BE は作成されません。

ライブ BE へのパッケージングの変更によって、それ以上の変更が不可能になる場合、最新の代替 BE が存在しない可能性があるため、このポリシーには最大のリスクがあります。

パッケージの署名のプロパティー

署名付きパッケージをインストールする場合は、このセクションで説明するイメージプロパティーとパブリッシャープロパティーを、パッケージの署名を検証するように設定します。

署名付きパッケージのイメージプロパティー

次のイメージプロパティーを、署名付きパッケージを使用するように構成します。

`signature-policy`

このプロパティーの値により、イメージ内のパッケージのインストール、更新、変更、または検証時に、マニフェストに実行されるチェックが決まります。パッケージに適用される最終的なポリシーは、イメージポリシーとパブリッシャーポリシーの組み合わせに依存します。この組み合わせの厳格さは、少なくとも、この 2 つのポリシーが個別に適用された場合の厳格な方と同じです。デフォルトでは、パッケージクライアントは証明書が失効済みかどうかをチェックしません。そのようなチェック (パッケージクライアントから外部 Web サイトへのアクセスが必要な場合がある) を有効にするには、`check-certificate-revocation` イメージプロパティーを `true` に設定します。次の値が許可されます。

`ignore`

すべてのマニフェストの署名を無視します。

`verify`

署名が含まれているすべてのマニフェストが有効に署名されていることを確認しますが、インストール済みパッケージがすべて署名されている必要はありません。

これがデフォルト値です。

`require-signatures`

新しくインストールされたすべてのパッケージに、有効な署名が少なくとも 1 つ含まれている必要があります。インストール済みパッケージに有効な署名が含まれていない場合は、`pkg fix` および `pkg verify` コマンドでも警告が表示されます。

`require-names`

`require-signatures` と同じ要件に従いますが、`signature-required-names` イメージプロパティーで一覧表示される文字列が、署名の信頼のチェーンを検証するために使用される証明書の共通名としても表示される必要があります。

`signature-required-names`
: このプロパティーの値は、パッケージの署名の検証中に、証明書の共通名として表示される必要のある名前の一覧です。

署名付きパッケージのパブリッシャープロパティー

次のパブリッシャープロパティーを、特定のパブリッシャーからの署名付きパッケージを使用するように構成します。

`signature-policy`
: このプロパティーの機能は、このプロパティーが指定したパブリッシャーからのパッケージにのみ適用される点を除き、`signature-policy` イメージプロパティーの機能と同じです。

`signature-required-names`
: このプロパティーの機能は、このプロパティーが指定したパブリッシャーからのパッケージにのみ適用される点を除き、`signature-required-names` イメージプロパティーの機能と同じです。

パッケージの署名プロパティーの構成

パッケージの署名プロパティーを構成するには、`set-property`、`add-property-value`、`remove-property-value`、および `unset-property` サブコマンドを使用します。

特定のパブリッシャーの署名ポリシーと必要な名前を指定するには、`set-publisher` サブコマンドの `--set-property`、`--add-property-value`、`--remove-property-value`、および `--unset-property` オプションを使用します。

次の例は、このイメージで、すべてのパッケージが署名されることを必須にするように構成します。また、この例では、信頼のチェーン内のいずれかの証明書の共通名として文字列「oracle.com」が表示されることも必須にします。

```
$ pkg set-property signature-policy require-names oracle.com
```

次の例は、このイメージで、署名されたすべてのパッケージが検証されることを必須にするように構成します。

```
$ pkg set-property signature-policy verify
```

次の例は、このイメージで、パブリッシャー `example.com` からインストールされたすべてのパッケージが署名されることを必須にするように構成します。

```
$ pkg set-publisher --set-property signature-policy=require-signatures example.com
```

次の例は、必要な署名を追加します。この例では、有効と見なされるには、署名の信頼のチェーンに表示される必要のあるイメージの共通名の一覧に文字列 `trustedname` を追加します。

```
$ pkg add-property-value signature-required-names trustedname
```

次の例は、必要な署名を削除します。この例では、有効と見なされるには、署名の信頼のチェーンに表示される必要のあるイメージの共通名の一覧から文字列 `trustedname` を削除します。

```
$ pkg remove-property-value signature-required-names trustedname
```

次の例は、指定したパブリッシャーの必要な署名を追加します。この例では、有効と見なされるには、署名の信頼のチェーンに表示される必要のあるパブリッシャー `example.com` の共通名の一覧に文字列 `trustedname` を追加します。

```
$ pkg set-publisher --add-property-value \
signature-required-names=trustedname example.com
```

追加のイメージのプロパティー

`ca-path`

SSL 操作用の CA 証明書が格納されたディレクトリを指すパス名を指定します。このディレクトリの形式は、ベースとなる SSL 実装に固有です。信頼できる CA 証明書のために別の場所を使用するには、別のディレクトリを指すようにこの値を変更します。CA ディレクトリの要件については、`SSL_CTX_load_verify_locations`(3openssl) の `CApath` に関する項目を参照してください。

デフォルト値は `/etc/openssl/certs` です。

`check-certificate-revocation`

`true` に設定すると、パッケージクライアントは、署名検証のために使用される証明書の CRL 配布ポイントへのアクセスを試み、発行時よりもあとに証明書が失効していないかどうかを調べます。

デフォルト値は `False` です。

`content-update-policy`

パッケージシステムがパッケージ操作中に編集不能なファイルを更新する時期を指定します。次の値が許可されます。

`default`

デフォルトのコンテンツ更新ポリシーを常に適用します。

`always`
: 変更された編集不能なファイルを常にダウンロードおよび更新します。

`when-required`
: 更新が必要だということがパッケージシステムで決定された場合に限り、変更された編集不能なファイルをダウンロードして更新します。

デフォルト値は `always` です。

`flush-content-cache-on-success`
: これが `true` に設定されている場合、パッケージクライアントはイメージ変更操作が正常に完了したときに内容キャッシュ内のファイルを削除します。BE を作成する操作の場合は、ソースと出力先の両方の BE から内容が削除されます。

このプロパティーを使用して、ディスク容量の限られたシステムで内容キャッシュを小さく保つことができます。このプロパティーを使用すると、操作が完了するまでの時間が長くなる可能性があります。

デフォルト値は `True` です。

`mirror-discovery`
: このプロパティーは、mDNS および DNS-SD を使用してリンクローカル内容ミラーを検出するようにパッケージクライアントに命令します。このプロパティーを `true` に設定すると、パッケージクライアントはミラーを動的に検出し、そのミラーからパッケージ内容のダウンロードを試みます。mDNS を介してその内容を通知するミラーを実行するには、`pkg.depotd(1M)` を参照してください。

デフォルト値は `False` です。

`send-uuid`
: ネットワーク操作の実行時にこのイメージの汎用一意識別子 (UUID) を送信します。ユーザーはこのオプションを無効にできますが、一部のネットワークリポジトリは UUID を供給しないイメージとのやり取りを拒否する場合があります。

デフォルト値は `True` です。

`trust-anchor-directory`
: このプロパティーの値は、イメージのトラストアンカーを含むディレクトリのパス名です。このパスはイメージのルートに相対的です。

デフォルト値は `etc/certs/CA` です。

`use-system-repo`
: このプロパティーではシステムリポジトリを、イメージおよびパブリッシャーの構成のソースとして、および提供されたパブリッシャーと通信するためのプロキシとしてイメージで使用するべきかどうかを指定します。システムリポジトリについては、`pkg.sysrepo(1M)`を参照してください。

デフォルト値は `ignore` です。

イメージのプロパティーの設定

イメージプロパティー設定を表示するには pkg property コマンドを使用します。イメージプロパティーを構成するには、set-property、add-property-value、remove-property-value、および unset-property サブコマンドを使用します。

イメージプロパティーの値の表示

イメージのプロパティーを表示するには、pkg property コマンドを使用します。

```
$ pkg property
PROPERTY                        VALUE
be-policy                       default
ca-path                         /etc/openssl/certs
check-certificate-revocation    False
flush-content-cache-on-success  False
mirror-discovery                False
preferred-authority             solaris
publisher-search-order          ['solaris', 'isvpub']
send-uuid                       True
signature-policy                verify
signature-required-names        []
trust-anchor-directory          etc/certs/CA
use-system-repo                 False
```

publisher-search-order プロパティーを設定するために、pkg set-publisher コマンドの検索順序オプションを使用する場合もあります。103 ページの「パブリッシャーの検索順序およびスティッキネスの設定」を参照してください。

イメージプロパティーの値の設定

イメージプロパティーの値を設定したり、プロパティーを追加して設定したりするには、pkg set-property コマンドを使用します。

次の例では、mirror-discovery プロパティーの値を設定します。

```
$ pkg set-property mirror-discovery true
$ pkg property -H mirror-discovery
mirror-discovery True
```

イメージプロパティーの値のリセット

指定したプロパティーの値をそのデフォルト値にリセットするには、pkg unset-property コマンドを使用します。

```
$ pkg unset-property mirror-discovery
$ pkg property -H mirror-discovery
mirror-discovery False
```

イメージの作成

イメージは、IPS パッケージとそれらの関連ファイル、ディレクトリ、リンク、および依存関係をインストールでき、その他の IPS 操作を実行できる場所です。

`pkg image-create` コマンドを使用して作成されたイメージは、ブート可能ではありません。ブート可能なイメージを作成するには、`pkg` コマンドを使用して `--be-name` または `--require-new-be` オプションを付けたり、`beadm` または `zonecfg` および `zoneadm` コマンドを使用したりします。`pkg image-create` コマンドは、パッケージおよびオペレーティングシステムのディストリビューションを維持するといったタスク向けに使用されています。

`pkg image-create` コマンドには、イメージが作成されるディレクトリとなるオペランドが必要です。作成されるイメージのデフォルトタイプはユーザーイメージです。次のいずれかのイメージタイプを指定できます。

フル
: フルイメージは完全なシステムを提供できます。フルイメージでは、イメージ自体の中ですべての依存関係が解決され、IPS が一貫した方法で依存関係を維持します。Oracle Solaris OS のインストールを完了した時点で、ルートファイルシステムとその内容がフルイメージに含まれています。フルイメージを指定するには、`-F` または `--full` オプションを使用します。

部分
: 部分イメージは、所定の *dir* パスを囲むフルイメージ (親イメージ) にリンクされます。部分イメージは、それ自体では完全なシステムを提供しません。部分イメージを指定するには、`-P` または `--partial` オプションを指定します。

  非大域ゾーンは部分イメージです。非大域ゾーンコンテキストでイメージを使用するには、`-z` または `--zone` オプションを指定して、適切なバリアントを設定します。ゾーンイメージでは、IPS は、パッケージ内の依存関係によって定義されているとおりに、非大域ゾーンと大域ゾーンの一貫性を維持します。ゾーンについての詳細は、20 ページの「イメージとブート環境」を参照してください。

ユーザー
: ユーザーイメージは、再配置可能なパッケージのみを含みます。これはイメージの型を指定しない場合に作成されるデフォルト型のイメージです。ユーザーイメージを指定するには、`-U` または `--user` オプションを指定します。

パッケージリポジトリ URI を指定するには、-p または --publisher オプションを使用します。パブリッシャーの名前も提供した場合、イメージの作成時にそのパブリッシャーのみが追加されます。パブリッシャーの名前を提供しない場合、指定されたリポジトリによって認識されているすべてのパブリッシャーがイメージに追加されます。このパブリッシャーに関連付けられたカタログは、初期作成操作に続いて取得が試みられます。

クライアント SSL 認証を使用するパブリッシャーの場合、-c オプションおよび -k オプションを使用して、クライアント鍵とクライアント証明書を登録します。この鍵と証明書は、イメージ作成中に追加されるすべてのパブリッシャーのために使用されます。

バリアント値、ファセット値、およびイメージプロパティー値を設定するには、--variant、--facet、および --set-property オプションを使用します。

操作履歴の表示

イメージを変更する pkg コマンドの履歴を表示するには、pkg history コマンドを使用します。pkg search、pkg refresh、pkg publisher、pkg facet、pkg property などのコマンドは pkg history では記録されませんが、pkg set-publisher、pkg change-facet、pkg set-property などのコマンドは pkg history で記録されます。pkg history は、入力された 1 つのコマンドに対して実行される複数の操作を表示することもあります。たとえば、pkg refresh --full は pkg history 出力に refresh-publishers と rebuild-image-catalogs を表示します。

デフォルトでは、pkg history コマンドは次の情報を表示します。

- 操作の開始時間
- install などの操作の名前
- pkg などのクライアント
- 操作の結果: Succeeded または Failed

追加情報やさらに詳細な情報を表示するにはオプションを使用します。

-l

デフォルトの情報に加えて次の情報を表示します。

- クライアントのバージョン
- 操作を実行したユーザーの名前
- 新しい BE が作成されたかどうか
- 操作が終了した時間
- 発行された完全なコマンド

- コマンドの実行中に発生したエラー
- update などの操作について、変更されたパッケージの完全な FMRI

-n *number*

最新のものから順に指定された数の操作のみを表示します。

```
$ pkg history -n4
START                   OPERATION               CLIENT              OUTCOME
2013-08-06T16:32:03     fix                     pkg                 Succeeded
2013-08-06T16:41:47     revert                  pkg                 Succeeded
2013-08-06T17:56:22     set-property            pkg                 Succeeded
2013-08-06T17:56:53     unset-property          pkg                 Succeeded
```

-o *column*[,*column*]...

列名を指定するコンマ区切りリストを使用して出力を表示します。pkg(1) のマニュアルページの列名のリストを参照してください。

```
$ pkg history -o start,time,operation,outcome -n4
START                   TIME      OPERATION               OUTCOME
2013-08-06T16:32:03     0:00:27   fix                     Succeeded
2013-08-06T16:41:47     0:00:43   revert                  Succeeded
2013-08-06T17:56:22     0:00:00   set-property            Succeeded
2013-08-06T17:56:53     0:00:00   unset-property          Succeeded
```

-t *time* | *time*-*time*[,*time* | *time*-*time*]...

%Y-%m-%dT%H:%M:%S 形式のタイムスタンプのコンマ区切りリストのログレコード (strftime(3C) のマニュアルページを参照)。日時の範囲を指定するには、開始と終了のタイムスタンプの間にハイフン (-) を使用します。キーワード now は、現在の時間の別名です。指定されたタイムスタンプに重複したタイムスタンプや重なり合う日付範囲が含まれている場合、重複する履歴イベントの単一インスタンスのみが表示されます。

-N

操作に関するリリースノートテキストを表示するには、-N オプションを使用します。-N オプションを -o オプションとともに使用することはできません。インストールされる一部のパッケージがリリースノートを持つ場合のインストールまたは更新の操作で -v オプションを指定した場合、操作の出力にリリースノートが表示されます。新しい BE へのインストール操作の場合、操作の出力では現在の BE の /tmp 内に、リリースノートファイルへのパスが指定されます。新しい BE でブートすると、リリースノートは /usr/share/doc/release-notes にありますが、あるいは次のコマンドに示すように -N オプションを使用して、リリースノートを表示することができます。

```
$ pkg history -N -n 1
```

リリースノートをインストールした操作が、この BE で実行した最後の pkg 操作でない場合、-n 引数で使用する数値を大きくするか、次のコマンドに示すように -t オプションを使用して、リリースノートをインストールした pkg 操作を識別します。

```
$ pkg history -N -t 2013-07-17T08:31:23
```

例 24 イメージ変更に関する情報の取得

次のコマンドは、指定された時間範囲内に発生した pkg 操作を表示します。

```
$ pkg history -t 2016-07-22T10:54:00-2016-07-22T10:55:00
START                   OPERATION               CLIENT             OUTCOME
2016-07-22T10:54:05     update-publisher        pkg                Succeeded
2016-07-22T10:54:06     refresh-publishers      pkg                Succeeded
2016-07-22T10:54:53     rebuild-image-catalogs  pkg                Succeeded
```

次のコマンドは、前のリストにある特定の操作に関する詳細情報を示します。

```
$ pkg history -lt 2016-07-22T10:54:05
         Operation: update-publisher
           Outcome: Succeeded
            Reason: None
            Client: pkg
           Version: 77d785eb851b
              User: admin (100)
         Boot Env.: 103_104.34291
    Boot Env. UUID: 1831b116-361f-4ef0-9406-851c446ec4c2
     New Boot Env.: None
New Boot Env. UUID: (None)
          Snapshot: (None)
        Start Time: 2016-07-22T10:54:05
          End Time: 2016-07-22T10:55:25
        Total Time: 0:01:20
           Command: /usr/bin/pkg set-publisher -g /var/share/pkg/repositories/solaris
 solaris
     Release Notes: No
       Start State:
None
         End State:
None
```

例 25 履歴情報の削除

すべての既存の履歴情報を削除するには、pkg purge-history コマンドを使用します。

```
$ pkg purge-history
```

♦ ♦ ♦ 付 録 A

パッケージのインストールおよび更新のトラブルシューティング

この付録では、パッケージをインストールまたは更新する際に見られることがある次のようなエラーを処理する方法を示します。

- パッケージをインストールできない
- 制約を満たすことができない

この付録では、まず、トラブルシューティングを始めるときに最初にチェックすべき情報について説明します。これらの推奨事項に従えば、大幅に時間を節約できます。

この付録では、パフォーマンスを向上し、格納されるメタデータを最小化するためのヒントも示します。

トラブルシューティングの初期手順

インストールするパッケージが構成済みのパブリッシャーから入手可能で、このイメージにインストールできるかどうかを確認します。このチェックを実行するには、次の情報が必要になります。

- インストールするパッケージが構成済みのパブリッシャーから入手可能かどうかを確認するには、インストールするパッケージの名前を `pkg list -af` コマンドの引数として指定します。このイメージに構成されているパブリッシャーの起点の情報が必要です。`pkg:/entire` パッケージの更新が必要な場合もあります。パブリッシャーの起点の変更が必要な場合もあります。
- このイメージにインストールされている `pkg:/entire` パッケージのバージョン。

```
$ pkg list -v entire
```

パッケージのインストールにおけるほとんどすべての問題について、次の 2 つの確認を最初に行う必要があります。

- インストールされている `pkg:/entire` 制約パッケージのバージョンを確認します。
- パッケージパブリッシャーの起点を確認します。

必要なパッケージが構成済みのパブリッシャーから入手可能であることが判別できたら、インストールの進捗に応じて次のステップを使用します。

- インストールまたは更新を行うときは常に `-nv` オプションを使用して、行われる変更内容 (どのパッケージのどのバージョンがインストールまたは更新されるのかや、新しい BE が作成されるかどうかなど) を確認してください。`-v` オプションは、この特定のインストールまたは更新操作に適用されるリリースノートも示します。
- さらに詳細なエラーメッセージを受け取るには、インストールするパッケージの FMRI を、バージョンおよびパブリッシャーも含めて詳しく指定します。

更新する場合、システムに現在インストールされているパッケージを提供するパッケージリポジトリにシステムがアクセスできる必要があります。たとえば、Oracle Solaris 11.2 から Oracle Solaris 11.3 に更新する場合は、`solaris` パブリッシャーに、インストールされている Oracle Solaris 11.2 パッケージと目的の Oracle Solaris 11.3 パッケージの両方へのアクセスが構成されている必要があります。パッケージが `ha-cluster` や `solarisstudio` などの別のパブリッシャーからインストールされている場合は、それらのパブリッシャーにも現在インストールされているパッケージに加えて目的の新しいパッケージへのアクセスが構成されている必要があります。

ほとんどの更新エラーの原因は不完全なパッケージリポジトリです。「Best Practices for Creating and Using Local IPS Package Repositories」 in 『Copying and Creating Package Repositories in Oracle Solaris 11.3』を参照してください。

`pkg:/entire` のインストール済みバージョンの確認

IPS パッケージのバージョン管理および 2 つの異なるパッケージのバージョンの比較方法について理解することは、多くの場合、パッケージのインストールおよび更新の問題のトラブルシューティングのために非常に重要です。詳細については、16 ページの「障害管理リソース識別子」および「Package Version」 in 『Packaging and Delivering Software With the Image Packaging System in Oracle Solaris 11.3』を参照してください。

現在インストールされている `pkg:/entire` 制約パッケージのバージョンを確認するには、`pkg info` または `pkg list` コマンドを使用します。

```
$ pkg list -Hv entire
pkg://solaris/entire@0.5.11,5.11-0.175.2.8.0.5.0:20150325T200338Z      i--
```

この `pkg:/entire` 制約パッケージのバージョンは、このシステムで Oracle Solaris 11.2 SRU 8.5 が実行されていることを示しています。

`pkg:/entire` 制約パッケージは、使用中のサポート可能なイメージを容易に維持できるようにするために、ほかの多くのパッケージのバージョンを制約します。制約パッケージによって制約されるパッケージの詳細は、14 ページの「制約パッケージ」を参照してください。

pkg:/entire 制約パッケージによって制約されているパッケージを直接インストールまたは更新することはできません。pkg:/entire によって制約されているパッケージをインストールまたは更新するには、pkg:/entire パッケージを更新する必要があります。詳細は、146 ページの「制約を満たすことができない」を参照してください。場合によっては、147 ページの「制約パッケージによって制約されたパッケージの更新」に示すように制約を削除できます。

構成済みのパブリッシャーの起点についての内容の確認

パッケージパブリッシャーの起点を確認するには、pkg publisher コマンドを使用します。

```
$ pkg publisher
PUBLISHER  TYPE    STATUS P LOCATION
solaris    origin  online F http://pkg.oracle.com/solaris/release/
```

セキュアな URI の場合、必要な鍵および証明書が正しくインストールされていることを確認し、パブリッシャーを構成するときに -k および -c オプションを使用します。

外部の場所へのプロキシがサイトで必要な場合、pkg set-publisher コマンドの --proxy オプションを使用して、そのプロキシを設定します。手順については、105 ページの「プロキシの指定」を参照してください。

鍵、証明書、プロキシなどのパブリッシャーに関する詳細な情報を表示するには、pkg publisher *publisher* コマンドを使用します。

有効化されたパブリッシャーのいずれかの起点 URI にアクセスできない場合、必要な場所がアクセス可能であったとしても、インストールまたは更新操作は失敗します。場所にアクセスできない原因となっている問題を修復できない場合、pkg set-publisher -G を使用してアクセスできない起点を削除するか、pkg set-publisher --disable を使用して、そのパブリッシャーを無効にすることができます。このパブリッシャーが不要になった場合、pkg unset-publisher を使用してパブリッシャーを削除します。

パッケージパブリッシャーの起点に、必要なパッケージが含まれていることを確認します。たとえば、solaris パブリッシャーの起点が公開リリースリポジトリに設定されている場合、サポートリポジトリからのみ利用可能なバージョンにパッケージを更新することはできません。

必須のインストール済みパッケージが使用可能かどうかの確認

インストール済みのパッケージを更新したり、インストール済みパッケージに依存するパッケージをインストールしたり、または非大域ゾーンをインストールしたりす

るには、パブリッシャーの起点として設定しているリポジトリが、イメージに現在インストールされているものと同じソフトウェアを最低限含んでいる必要があります。リポジトリには古いソフトウェアまたは新しいソフトウェアを含めることもできますが、イメージにインストールされているものと同じソフトウェアを含む必要があります。

インストール済みパッケージを確認するときは、`pkgrepo list` コマンドを使用し、`pkg list` コマンドを使用しないでください。`pkg list` コマンドは、インストール済みパッケージが構成済みのパブリッシャーの起点から使用可能でない場合でも、パッケージを常に表示します。

次のコマンドは、`pkg:/entire` のインストール済みバージョンがパブリッシャーの起点から使用できないため、指定されたリポジトリはこのイメージについて適切なパブリッシャーの起点ではないことを示しています。

```
$ pkg list entire
NAME (PUBLISHER)                                  VERSION                    IFO
entire                                            0.5.11-0.175.2.8.0.5.0     i--
$ pkg publisher
PUBLISHER  TYPE    STATUS P LOCATION
solaris    origin  online F http://pkg.oracle.com/solaris/release/
$ pkgrepo list -Hs http://pkg.oracle.com/solaris/release entire@0.5.11-0.175.2.8.0.5.0
pkgrepo list: The following pattern(s) did not match any packages:
        entire@0.5.11-0.175.2.8.0.5.0
```

必要なパッケージが一覧表示されない場合、`pkgrepo refresh` コマンドをふたたび実行して、`pkgrepo list` コマンドを再試行してください。

次のコマンドは、`pkg:/entire` のインストール済みバージョンが、指定されたリポジトリから使用できることを示しています。

```
$ pkgrepo list -Hs /var/share/pkgrepos/solaris entire@0.5.11-0.175.2.8.0.5.0
solaris   entire                  0.5.11,5.11-0.175.2.8.0.5.0:20150325T200338Z
```

必要なパッケージが構成済みのパブリッシャーから使用できないが、別のリポジトリの起点から使用できる場合、次のいずれかのアクションを実行します。

- `solaris` パブリッシャーについてこの起点を追加するには、`pkg set-publisher` コマンドの `-g` オプションを使用します。
- `solaris` パブリッシャーについての起点を変更するには、`pkg set-publisher` コマンドの `-g` および `-G` オプションを使用します。
- 必要なパッケージが別のパブリッシャーから提供される場合、`pkg set-publisher` コマンドを使用して、そのパブリッシャーを追加します。
- 検索するリポジトリのリストの最後にリポジトリを一時的に追加するには、インストールコマンド (`install`、`uninstall`、`update`、`change-variant`、および `change-facet`) の `-g` オプションを使用します。
- 必要なパッケージを含めるには、パッケージリポジトリを更新します。「Best Practices for Creating and Using Local IPS Package Repositories」 in 『Copying and Creating Package Repositories in Oracle Solaris 11.3』を参照してください。

インストールするパッケージが使用可能かどうかの確認

インストールするパッケージが構成済みのパブリッシャーから入手可能かどうかを確認するには、次のコマンドを使用します。パッケージを更新する場合、現在インストールされているパッケージのバージョンと、更新後のバージョンの両方が使用可能である必要があります。

```
$ pkg list -af package
```

必要なパッケージが一覧表示されない場合、`pkg refresh` コマンドをふたたび実行して、`pkg list` コマンドを再試行してください。

引き続き必要なパッケージがリストされていない場合は、新しいパブリッシャーまたは新しいパブリッシャー起点を追加するか、パッケージリポジトリを更新します。

インストールするパッケージがこのイメージにインストール可能かどうかの確認

`-af` オプションを使用したときに、必要なパッケージバージョンが一覧表示された場合、`-f` オプションを使用せずに同じコマンドをふたたび使用します。

```
$ pkg list -a package
```

必要なバージョンが依然として一覧表示される場合、このパッケージは制約を受けておらず、ほかのパッケージをインストールまたは更新しなくても、このパッケージをインストールできるはずです。

必要なバージョンが一覧表示されない場合、このバージョンは構成済みパブリッシャーから使用できますが、このイメージにインストール可能ではありません。パッケージがインストールできない理由として、次のことが考えられます。

- パッケージがバリアントまたはファセットの設定によって制約されている。
- パッケージのバージョンが、制約パッケージによって制約されている。制約するパッケージを更新したり、場合によっては制約を緩和したりできます。詳細は、146 ページの「制約を満たすことができない」を参照してください。
- パッケージのバージョンが、凍結操作によって制約されている。`pkg freeze` コマンドを実行します。インストールする異なるバージョンのパッケージに対して `require` 依存関係を持つパッケージに凍結が適用されている可能性があり、両方のバージョンを同時にインストールすることはできません。

インストールの再試行

インストールまたは更新するパッケージを指定し、パブリッシャーを指定しない場合、パッケージ FMRI またはパターンに一致するパッケージを提供する、パブリッ

シャーの検索順序で最初のパブリッシャーがインストールソースとして使用されます。該当するパブリッシャーがこのイメージ内にインストール可能なパッケージのバージョンを提供しない場合、使用可能な別のパブリッシャーがこのイメージ内にインストール可能なパッケージのバージョンを提供したとしても、インストール操作は失敗します。この問題に対処するには次のいずれかのアクションを実行します。

- パッケージ FMRI でパブリッシャーを指定します。たとえば、フルパッケージ名の前に `pkg://solaris/` を指定します。
- 必要なパッケージバージョンを提供するパブリッシャーを検索順序の最初のパブリッシャーに設定するには、`pkg set-publisher` コマンドの `-P` オプションを使用します。

2 つのパブリッシャーが同じ名前を持つパッケージを提供しないようにしてください。この状況が発生した場合は、次のいずれかの処置を取ってください。

- パブリッシャーの 1 つのパッケージのいずれかが不要になった場合は、そのパブリッシャーを無効にするか設定解除します。現在そのパブリッシャーのパッケージがインストールされている場合は、パブリッシャーを無効にするか設定解除する前にそれらをアンインストールします。
- 両方のパブリッシャーが必要な場合は、次のいずれかを実行します。
  - 優先されないパブリッシャーが固定的ではない (`--non-sticky`) ことを確認し、インストールするパッケージを提供するパブリッシャーが同じ名前のパッケージを提供するほかのパブリッシャーの前に来るように、パブリッシャーの検索順序を設定します。
  - インストールするパッケージの完全な名前を、パブリッシャーを含めて指定します。

凍結またはバージョンロックされているパッケージがインストールまたは更新を妨げているかどうかを確認するには、`pkg freeze` および `pkg facet` コマンドを使用します。

パッケージ化された内容が変更されているかどうかを確認するには、`pkg verify -v` コマンドを使用します。見つかったエラーを修正するには、`pkg fix` コマンドを使用します。

インストールまたは更新を行うときは常に `-nv` オプションを使用して、行われる変更内容 (どのパッケージのどのバージョンがインストールまたは更新されるのかや、新しい BE が作成されるかどうかなど) を確認してください。`-v` オプションは、この特定のインストールまたは更新操作に適用されるリリースノートも示します。

- `-nv` オプションを使用してもエラーメッセージを受け取らない場合は、`-n` オプションなしでコマンドを再度実行して、インストールまたは更新を実際に実行します。新しい BE 内でインストールを実行するか、新しい BE またはバックアップ BE がデフォルトで作成されない場合にバックアップ BE を作成するためのオプションを指定するかどうかを検討します。
- エラーメッセージを受け取る場合は、次のアクションを行います。

- 問題の診断および修正に役立つ多くの情報を取得するために、パッケージ FMRI に多くのバージョンを指定します。
- 詳細オプション `-v` を指定します (`-nvv` など)。
- `pkg history` コマンドを使用します。`-l` オプションによって、変更されたパッケージの完全な FMRI が表示されます。130 ページの「操作履歴の表示」を参照してください。

インストールまたは更新する複数のパッケージを指定するか、更新操作でパッケージ指定を省略したとき、いずれかのパッケージをこのイメージにインストールできない場合はインストールまたは更新操作が失敗します。1 つのパッケージをインストールできない場合、パッケージは一切インストールされません。詳細は、インストールできないパッケージのみ指定し、そのパッケージの完全な FMRI を指定し、1 つ以上の `-v` オプションを指定して、コマンドをふたたび呼び出してください。

パブリッシャーまたはリポジトリにアクセスできない

このセクションで説明するエラーは、パブリッシャーの URI にアクセスできないことに関連しています。

Oracle Enterprise Manager Ops Center のパブリッシャーの構成

Ops Center を使用している場合、IPS パッケージリポジトリは、Oracle Solaris ソフトウェア更新ライブラリと呼ばれます。ライブラリを更新する方法については、Oracle Enterprise Manager Ops Center の構成リファレンスマニュアルでソフトウェアライブラリのセクションの Oracle Solaris のセクションを参照してください。

`solaris` パブリッシャーに新しい証明書を関連付けるには、Ops Center BUI でライブラリ -> Oracle Solaris セクションの、親リポジトリの構成およびコンテンツの追加アクションを使用します。

システムが外部 IPS リポジトリにアクセスできるようなファイアウォールルールについては、Oracle Enterprise Manager Ops Center のポートおよびプロトコルに関するガイドでファイアウォールルールのセクションを参照してください。このリストは、Ops Center を使用していない場合にも役立ちます。

パッケージリポジトリにアクセスできない

エラーメッセージ:

- `Couldn't resolve host`
- `Unable to contact any configured publishers`
- `Unable to contact valid package repository`
- `Origin URIs do not appear to point to a valid pkg repository`
- `Framework error: code: E_COULDNT_CONNECT (7) reason: Failed to connect`

パブリッシャーの URI を表示するには、`pkg publisher` コマンドを使用します。次の例の LOCATION 列または URI 行を参照してください。

```
$ pkg publisher
PUBLISHER                   TYPE     STATUS P LOCATION
solaris                     origin   online F http://pkg.oracle.com/solaris/release/
$ pkg publisher solaris
            Publisher: solaris
                Alias:
           Origin URI: http://pkg.oracle.com/solaris/release/
```

使用しなくなったパブリッシャーが一覧表示される場合、これらのパブリッシャーを無効にするか、削除します。

```
$ pkg set-publisher --disable publisher
$ pkg unset-publisher publisher
```

使用しているパブリッシャーについては、URI が正しいことを確認してください。

- 各パブリッシャーの起点の場所をブラウザで表示するか、その場所に ping を実行してみてください。
- `pkgrepo list` コマンドを使用して、その起点のパッケージを一覧表示してみてください。

起点の場所が正しくない場合、`pkg set-publisher` コマンドの `-G` および `-g` オプションを一緒に使用して、URI を変更します。

パブリッシャーに複数の起点がある場合は、すべての起点の場所にアクセスできることが必要です。いずれかの起点の場所にアクセスできない場合、`pkg set-publisher` コマンドの `-G` オプションを使用して、その起点を削除します。

パブリッシャーが非大域ゾーン内で構成された場合、そのパブリッシャーが大域ゾーン内で構成されていなくても、そのパブリッシャーについてのすべての場所が大域ゾーンからアクセス可能である必要があります。

イメージに非大域ゾーンが含まれている場合は、143 ページの「場所が見つかりません」で非大域ゾーンの場合の手順を参照してください。

SSL 証明書の問題

エラーメッセージ:

- `Framework error: code: 35 reason: SSL routines`
- `Framework error: code: 60 reason: SSL certificate problem, verify that the CA cert is OK`
- `Framework error: code: 60 reason: SSL certificate problem: self signed certificate`

注記 - クライアント SSL 証明書が必要なリポジトリは、`pkg` インストールコマンドに `-g` オプションを使用して指定することができません。

`pkg` コマンドによって、SSL の問題に関するメッセージが表示される場合、次の 1 つ以上のアクションを実行してください。

- システムの時間と日付が正しいことを確認してください。
- パブリッシャーの起点で必要な鍵および証明書がインストールされていて、期限切れではないことを確認します。

  鍵および証明書を取得する方法についての情報は、セキュアリポジトリに関するほかの情報に含まれています。たとえば、`https://pkg-register.oracle.com/` のサイトを使用して、`https://pkg.oracle.com/solaris/support/` Oracle Solaris サポートリポジトリの鍵および証明書を取得します。

  このパブリッシャーの鍵および証明書ファイルをインストールするには、`pkg set-publisher` コマンドで `-k` および `-c` オプションを使用します。各パブリッシャーには 1 つだけの鍵と証明書を指定できます。パブリッシャーに複数のセキュアな起点が構成されている場合、すべてのセキュアな起点で 1 つの鍵と証明書を共有します。

  ```
  $ pkg set-publisher -k /tmp/keyfile -c /tmp/certfile publisher-name
  ```

  鍵および証明書ファイルがインストールされており、期限切れでないことを確認するには、パブリッシャーに対して `pkg publisher` コマンドを使用します。

  ```
  $ pkg publisher solaris

              Publisher: solaris
                  Alias:
             Origin URI: https://pkg.oracle.com/solaris/support/
                SSL Key: /var/pkg/ssl/keyfile
               SSL Cert: /var/pkg/ssl/certfile
   Cert. Effective Date: July  1, 2015 04:47:13 PM
  Cert. Expiration Date: July  8, 2017 04:47:13 PM
            Client UUID: client-uuid
        Catalog Updated: May 11, 2016 03:28:43 PM
                Enabled: Yes
             Properties:
                          proxied-urls = []
                          signature-policy = require-signatures
  ```

鍵または証明書の有効期限が切れている場合は、次のようなエラーメッセージが表示されます。

```
Certificate '/var/pkg/ssl/certfile' has expired.
  Please install a valid certificate.
```

- 鍵および証明書が指定された起点で有効であることを確認します。

  次のコマンドは、指定された起点に鍵と証明書が必要であるため失敗します。

  ```
  $ pkgrepo info -s https://pkg.oracle.com/solaris/support/
  ```

  次のコマンドは、`pkg publisher` *publisher-name* の出力からコピーされた鍵および証明書ファイル名が有効であるために成功します。

  ```
  $ pkgrepo info -s https://pkg.oracle.com/solaris/support/ \
  > --key /var/pkg/ssl/keyfile --cert /var/pkg/ssl/certfile
  PUBLISHER PACKAGES STATUS           UPDATED
  solaris   6711     online           2016-05-19T19:00:10.152688Z
  ```

- CA 証明書が破損していないことを確認します。

  `crypto/ca-certificates` パッケージを確認します。

  ```
  $ pkg verify crypto/ca-certificates
  ```

  問題が報告されている場合は、次の手順を実行します。

  - `crypto/ca-certificates` パッケージを修正します。

    ```
    $ pkg fix crypto/ca-certificates
    ```

  - `system/ca-certificates` SMF サービスをリフレッシュします。

    ```
    $ svcadm refresh svc:/system/ca-certificates:default
    ```

- プロキシおよびファイアウォールを確認します。

  外部の場所へのプロキシがサイトで必要な場合、`pkg set-publisher` コマンドの `--proxy` オプションを使用して、そのプロキシを設定します。手順については、105 ページの「プロキシの指定」を参照してください。

  システムが外部 IPS リポジトリにアクセスできるようなファイアウォールルールについては、Oracle Enterprise Manager Ops Center のポートおよびプロトコルに関するガイドでファイアウォールルールのセクションを参照してください。このリストは、Ops Center を使用していない場合にも役立ちます。

- 自己署名付き証明書を使用している場合は、「Creating a Self-Signed Server Certificate Authority」 in 『Copying and Creating Package Repositories in Oracle Solaris 11.3』の説明に従って、CA 証明書をシステムに追加します
- Ops Center を使用している場合は、139 ページの「Oracle Enterprise Manager Ops Center のパブリッシャーの構成」を参照してください。

場所が見つかりません

エラーメッセージ: `http protocol error: code: 404 reason: Not Found`

139 ページの「パッケージリポジトリにアクセスできない」の説明に従って、パブリッシャー URI を確認します。場所を表示したり ping を実行したりできる場合、`pkgrepo list` コマンドを使用して、リポジトリ内のいずれかのパッケージを表示してみてください。

URI がファイルベースのディレクトリの場合、ファイルおよびディレクトリが `pkg5srv` ユーザーによって読み取り可能であることを確認してください。`pkgrepo verify` コマンドを使用して、リポジトリが `pkg5srv` ユーザーによって読み取り可能であるかどうかを確認できます。

Web サーバー構成を確認します。詳細は、Chapter 5, 「Running the Depot Server Behind a Web Server」 in 『Copying and Creating Package Repositories in Oracle Solaris 11.3』を参照してください。

- Apache Web サーバーインスタンスの背後にパッケージデポサーバーを実行している場合、エンコードされたスラッシュをデコードしないように、次の設定を `httpd.conf` ファイルに含めます。

  ```
  AllowEncodedSlashes NoDecode
  ```

- デポサーバー `pkg/proxy_base` を、Apache サーバー上のリポジトリの URL に設定します。

  ```
  $ svccfg -s pkg/server:repo setprop pkg/proxy_base = astring: http://pkg.example.com/myrepo
  $ svcadm refresh pkg/server:repo
  ```

非大域ゾーンで問題が発生する場合、次のトラブルシューティングステップを実行します。非大域ゾーンはシステムリポジトリと呼ばれる特殊なパッケージリポジトリを使用します。システムリポジトリの詳細は、`pkg.sysrepo(1M)` のマニュアルページを参照してください。

- 非大域ゾーンを持つイメージ内で `-g` オプションを使用することはできません。代わりに、`pkg set-publisher` コマンドを使用して、そのパブリッシャーおよび起点を明示的に追加します。
- ファイルベースリポジトリ内のファイルおよびディレクトリが `pkg5srv` ユーザーによって読み取り可能であることを確認してください。`pkg5srv` ユーザーは `system-repository` Apache インスタンスを実行します。システムリポジトリの場所を見つける方法を示す例については、64 ページの「大域ゾーンと非大域ゾーンの関係」を参照してください。
- 外部の場所にアクセスするためにサイトでプロキシが必要な場合、大域ゾーンのパブリッシャーに対してプロキシが正しく指定されていることを確認します。プロキシを指定するには、`pkg set-publisher` コマンドの `--proxy` オプションを使用

します。手順については、105 ページの「プロキシの指定」を参照してください。プロキシの確認方法の 1 つとして、`pkg refresh --full` コマンドからアクセスエラーメッセージを受け取らないことを確認する方法があります。

- サービス `svc:/application/pkg/system-repository:default` が大域ゾーンでオンラインであることを確認します。
- サービス `svc:/application/pkg/zones-proxyd:default` が大域ゾーンでオンラインであること、およびサービス `svc:/application/pkg/zones-proxy-client:default` が非大域ゾーンでオンラインであることを確認します。
- 大域ゾーンで、`/var/log/pkg/sysrepo/*` 内のログファイルを調べ、ファイルの読み取りを試行するときに権限エラーが報告されていないかどうかを確認します。404 または 503 エラーが `/var/log/pkg/sysrepo/access_log` に報告されていないか確認します。`/var/log/pkg/sysrepo/error_log` にエラーが報告されていないか確認します。
- 大域ゾーンにおいて、`/etc/hosts` ファイルで `localhost` が `127.0.0.1` に設定されていることを確認します。`/system/volatile/pkg/sysrepo/sysrepo_httpd.conf` ファイルで、`Listen` が `127.0.0.1:1008` に設定され、`ServerName` が `127.0.0.1` に設定されていることを確認します。
- 大域ゾーンで、ファイル `/system/volatile/pkg/sysrepo/sysrepo_httpd.conf` が次の形式の Alias 行を含むかどうかを確認します。

```
$ grep Alias /system/volatile/pkg/sysrepo/sysrepo_httpd.conf
WSGIScriptAlias /wsgi_p5p /etc/pkg/sysrepo/sysrepo_p5p.py
```

`sysrepo_httpd.conf` ファイルに Alias 行がない場合、`sysrepo` サービスを再起動します。

```
$ svcadm restart svc:/application/pkg/system-repository:default
```

サービスが使用不可

エラーメッセージ: `http protocol error: code: 503 reason: Service Unavailable`

`pkg publisher` コマンドを使用して、使用しようとしているパッケージリポジトリの場所を検出し、そのシステム上の SMF サービスを調べます。次のコマンドを使用して、使用可能であるが実行中でないパッケージリポジトリ SMF サービスインスタンスと、使用可能な別のインスタンスの実行を妨げているインスタンスを識別します。

```
$ svcs -xv pkg/server
svc:/application/pkg/server: default (image packaging repository)
 State: online since July 25, 2013 07:53:50 AM PDT
   See: /var/svc/log/application-pkg-server:default.log
Impact: None.
```

いずれかのサービスが問題を報告している場合、`svcs` 出力に一覧表示されるログファイルを確認して、特定の問題を判別します。

inst_root プロパティー、port プロパティー、およびその他のプロパティーが正しく設定されていることを確認します。

```
$ svcprop -p pkg pkg/server:default
$ svcprop -p pkg/inst_root -p pkg/port pkg/server:default
/var/share/pkgrepos/solaris
80
```

必要な場合、次の例に示すように、svccfg コマンドを使用してプロパティー値をリセットします。

```
$ svccfg -s pkg/server:default setprop pkg/port=1008
```

必要に応じてサービスインスタンスをクリア、リフレッシュ、再起動、および使用可能にするには、svcadm コマンドを使用します。

使用可能な更新がない

エラーメッセージ: No updates available for this image

特定のパッケージを更新する場合、次のコマンドを使用して、このイメージに現在インストールされているパッケージのバージョンを表示します。すべてのインストール済みパッケージを更新する場合 (pkg update でパッケージを指定しないか、パッケージ名として '*' を指定)、これらのコマンドの *package* に pkg:/entire を使用します。

```
$ pkg list -v package
```

現在インストールされているバージョンが存在しない場合、pkg update ではなく pkg install コマンドを使用してください。

あるバージョンの *package* が現在インストールされている場合、次のコマンドを使用して、構成済みのパブリッシャーから使用できる *package* のバージョンを表示します。

```
$ pkg list -afv package
```

最上位のバージョン番号を持つパッケージがすでにインストールされている場合、それより新しいバージョンはおそらく存在しません。

新しいバージョンが存在する場合、新しいバージョンが使用可能なパッケージリポジトリの場所を判別し、pkg set-publisher コマンドを使用して起点 URI をリセットするか、適切なパブリッシャーの起点 URI を追加します。必要な場合、必要な鍵および証明書をインストールし、-k および -c オプションを使用して、これらを指定します。pkgrepo list コマンドを使用して、現在インストールされているバージョンのパッケージが、構成済みのパブリッシャーからも利用できることを確認します。

次の例に示すように、インストールするパッケージの FMRI にバージョン (または キーワード `latest`) を含めて `-nv` オプションを指定して、`pkg update` コマンドを再実行します。パッケージ名に指定する情報が多くなれば、エラー出力の情報も多くなります。

```
$ pkg update -nv package@latest
```

パッケージをインストールできない

エラーメッセージ: `No matching version of` *package* `can be installed`

次のコマンドを使用して、構成済みのパッケージパブリッシャーから使用できる *package* のバージョンを表示します。

```
$ pkg list -afv package
```

インストールするパッケージの FMRI を詳細に指定します。最初に検出された一致はこのイメージにインストール可能でないかもしれませんが、インストールする特定のバージョンはインストール可能な場合があります。FMRI をさらに特定しても依然としてインストール可能でない場合、FMRI を詳しく指定すると、パッケージをインストールできない理由が詳しく表示されます。

インストールまたは更新するパッケージのバージョンが凍結されていないことを確認してください。バージョンが凍結されたすべてのパッケージを一覧表示するには、引数を指定せずに `pkg freeze` コマンドを使用します。

インストールされている `pkg:/entire` パッケージのバージョンを表示するには、次のコマンドを使用します。

```
$ pkg list -v entire
```

`pkg:/entire` 制約パッケージまたはほかの制約パッケージによって制約されているパッケージをインストールまたは更新することはできません。制約パッケージを更新する必要があります。詳細は、146 ページの「制約を満たすことができない」を参照してください。

制約を満たすことができない

エラーメッセージ: `No solution was found to satisfy constraints`

このメッセージは、あるバージョンのパッケージをインストールしようとしたが、そのパッケージのバージョンが、制約パッケージによる制約で指定されたパッケージの

バージョンと一致しないことを示しています。制約パッケージとバージョン制約については、14 ページの「制約パッケージ」を参照してください。

制約パッケージによって、パッケージのセットは一緒に機能するバージョンに制約されるため、サポート可能なイメージを保持するのに役立ちます。このため、制約パッケージによって制約される 1 つのパッケージを更新しないでください。代わりに、制約パッケージを更新する必要があり、この結果、制約されたすべてのパッケージは、一緒にテストされた新しいバージョンのセットに更新されます。

パッケージが変更されるとき (たとえば更新されるとき)、`pkg` クライアントは関連するパッケージとそれらの依存関係を調べます。いずれかの依存パッケージをインストールまたは更新できない場合、インストールまたは更新できないパッケージに依存するパッケージごとに、個別のエラーメッセージが生成されます。大量のエラーメッセージを処理するもっとも効果的な方法は、もっともインデントされたエラーメッセージを最初に調べるやり方です。

158 ページの「非大域ゾーンをインストールできない」も参照してください。

制約パッケージによって制約されたパッケージの更新

エラーメッセージ:

- `No suitable version of installed package` *package* `found`
- `All versions matching 'incorporate' dependency` *package* `are rejected`
- `This version excluded by specified installation version`
- `This version is excluded by installed incorporation`

制約パッケージによって制約されたパッケージの場合、制約パッケージを更新し、制約されたすべてのパッケージを、一緒にテストされたセットとして保持することがベストプラクティスです。

制約パッケージからの 1 つのパッケージのみ更新する場合、パッケージの `version-lock` ファセットが `true` に設定されているかどうかを確認します。パッケージに `version-lock` ファセットが関連付けられている場合、そのパッケージを制約パッケージからロック解除できます。`version-lock` ファセットを `false` に設定して制約を除去し、その後パッケージのインストールまたは更新を再試行します。`-nv` オプションと、パッケージ名の FMRI に必要なパッケージのバージョンを指定します。114 ページの「制約パッケージによって指定されたバージョン制約の緩和」も参照してください。

例 26 Java Runtime Environment のロック解除と更新

次の例は、`runtime/java/jre-7` パッケージを更新する方法を示しています。`jre-7` パッケージは `consolidation/java/java-incorporation` パッケージによって制約さ

れ、java-incorporation パッケージは同様に pkg:/entire 制約パッケージによって制約されます。

次のコマンドは、0.175.2.0.0.9.0 バージョンの jre-7 が現在インストールされており、構成済みのパッケージリポジトリから新しいバージョンが使用可能であることを示しています。

```
$ pkg list -af runtime/java/jre-7
NAME (PUBLISHER)                                  VERSION                    IFO
runtime/java/jre-7                                1.7.0.21-0.175.2.0.0.13.0  ---
runtime/java/jre-7                                1.7.0.17-0.175.2.0.0.9.0   i--
```

-f オプションを削除すると、可能な更新後のバージョンが表示されます。次の pkg list 出力には、このイメージにインストール可能な新規バージョンがないことが示され、pkg update コマンド出力でこの状態が確認できます。-n オプションは、実行される変更を示しますが、変更を一切加えません。

```
$ pkg list -a runtime/java/jre-7
NAME (PUBLISHER)                                  VERSION                    IFO
runtime/java/jre-7                                1.7.0.17-0.175.2.0.0.9.0   i--
$ pkg update -nv runtime/java/jre-7
No updates available for this image.
```

このパッケージを更新できない理由を詳細に示すには、更新後のバージョンを指定します。次の例に示す出力では、インストール済みの java-incorporation@0.5.11,5.11-0.175.2.0.0.9.0 パッケージは、jre-7@1.7.0.21-0.175.2.0.0.13.0 パッケージのインストールを許可しないことを示しています。java-incorporation@0.5.11,5.11-0.175.2.0.0.13.0 パッケージは、jre-7@1.7.0.21-0.175.2.0.0.13.0 パッケージのインストールを許可しますが、インストール済みの entire@0.5.11,5.11-0.175.2.0.0.12.0 制約パッケージは jre-7@1.7.0.21-0.175.2.0.0.13.0 パッケージのインストールを許可しません。

```
$ pkg update -nv runtime/java/jre-7@1.7.0.21-0.175.2.0.0.13.0
pkg update: No solution was found to satisfy constraints

maintained incorporations:
  [output omitted]
  pkg://solaris/entire@0.5.11,5.11-0.175.2.0.0.12.0:20130415T172730Z

Plan Creation: dependency error(s) in proposed packages:
  [output omitted]
  No suitable version of required package pkg://solaris/consolidation/java/java-
incorporation@0.5.11,5.11-0.175.2.0.0.9.0:20130304T213946Z found:
    Reject:  pkg://solaris/consolidation/java/java-incorporation@0.5.11,5.11-0.1
75.2.0.0.9.0:20130304T213946Z
    Reason:  All versions matching 'incorporate' dependency pkg:/runtime/java/jr
e-7@1.7.0.17,5.11-0.175.2.0.0.9.0 are rejected
      Reject:  pkg://solaris/runtime/java/jre-7@1.7.0.17,5.11-0.175.2.0.0.9.0:20
130304T214022Z
      Reason:  This version excluded by specified installation version
        Reject:  pkg://solaris/runtime/java/jre-7@1.7.0.17,5.11-0.175.2.0.0.9.0:
20130304T214022Z
        Reason:  This version excluded by specified installation version
      Reject:  pkg://solaris/consolidation/java/java-incorporation@0.5.11,5.11-0
.175.2.0.0.13.0:20130429T145534Z
      Reason:  This version is excluded by installed incorporation pkg://solaris
/entire@0.5.11,5.11-0.175.2.0.0.12.0:20130415T172730Z
```

```
Plan Creation: Errors in installed packages due to proposed changes:
  [output omitted]
  No suitable version of installed package pkg://solaris/consolidation/java/java
-incorporation@0.5.11,5.11-0.175.2.0.0.9.0:20130304T213946Z found
    Reject:  pkg://solaris/consolidation/java/java-incorporation@0.5.11,5.11-0.1
75.2.0.0.9.0:20130304T213946Z
    Reason:  All versions matching 'incorporate' dependency pkg:/runtime/java/jr
e-7@1.7.0.17,5.11-0.175.2.0.0.9.0 are rejected
      Reject:  pkg://solaris/runtime/java/jre-7@1.7.0.17,5.11-0.175.2.0.0.9.0:20
130304T214022Z
      Reason:  This version excluded by specified installation version
    Reject:  pkg://solaris/consolidation/java/java-incorporation@0.5.11,5.11-0.1
75.2.0.0.13.0:20130429T145534Z
    Reason:  This version is excluded by installed incorporation pkg://solaris/e
ntire@0.5.11,5.11-0.175.2.0.0.12.0:20130415T172730Z
```

ベストプラクティスは、`entire` パッケージを更新することです。`entire` パッケージを更新すると、`java-incorporation` パッケージが更新され、その結果として `jre-7` パッケージが更新されます。この例では、Java パッケージを更新する必要があり、イメージを `entire` の更新済みバージョンに変更することはできません。

インストール可能な Java ソフトウェアのバージョンは、Java 制約パッケージの `version-lock` ファセットを設定することによって制約されます。ほかのソフトウェアを更新せずに Java ソフトウェアを更新するには、Java 制約パッケージの `version-lock` ファセットをロック解除し、そのあとで Java 制約パッケージを更新します。version-lock ファセットの詳細は、114 ページの「制約パッケージによって指定されたバージョン制約の緩和」を参照してください。

次のコマンドによって、インストール済みの `java-incorporation` パッケージの `version-lock` ファセットの値が `false` に変更されます。イメージ内の各パッケージがこのファセットに対して検査されるため、更新するパッケージ数は、このイメージにインストールされているパッケージの数です。

```
$ pkg change-facet \
facet.version-lock.consolidation/java/java-incorporation=false
            Packages to update: 856
     Variants/Facets to change:   1
       Create boot environment:  No
Create backup boot environment: Yes

Planning linked: 1/1 done
PHASE                                          ITEMS
Removing old actions                             1/1
Updating image state                            Done
Creating fast lookup database                   Done
Reading search index                            Done
Building new search index                     856/856
```

次のコマンドは、ファセット値が変更されたことを示しています。

```
$ pkg facet
FACETS                                                  VALUE
facet.version-lock.consolidation/java/java-incorporation False
```

次のコマンドでは `-n` オプションが指定されているため、このコマンドによって変更される内容が示されますが、イメージを実際に変更するわけではありません。

```
$ pkg update -nv java-incorporation
            Packages to update:         2
     Estimated space available:  80.91 GB
Estimated space to be consumed: 687.28 MB
       Create boot environment:        No
Create backup boot environment:       Yes
          Rebuild boot archive:        No

Changed packages:
solaris
  consolidation/java/java-incorporation
    0.5.11,5.11-0.175.2.0.0.9.0:20130304T213946Z -> 0.5.11,5.11-0.175.2.0.0.13.0
:20130429T145534Z
  runtime/java/jre-7
    1.7.0.17,5.11-0.175.2.0.0.9.0:20130304T214022Z -> 1.7.0.21,5.11-0.175.2.0.0.
13.0:20130429T145626Z
```

次のコマンドは実際の更新を実行します。このコマンドは、現在のイメージ内で更新を実行します。新しいブート環境で更新を実行するために、`--be-name` オプションを使用する場合もあります。

```
$ pkg update -v java-incorporation
            Packages to update:         2
     Estimated space available:  80.91 GB
Estimated space to be consumed: 687.28 MB
       Create boot environment:        No
Create backup boot environment:       Yes
          Rebuild boot archive:        No

Changed packages:
solaris
  consolidation/java/java-incorporation
    0.5.11,5.11-0.175.2.0.0.9.0:20130304T213946Z -> 0.5.11,5.11-0.175.2.0.0.13.0
:20130429T145534Z
  runtime/java/jre-7
    1.7.0.17,5.11-0.175.2.0.0.9.0:20130304T214022Z -> 1.7.0.21,5.11-0.175.2.0.0.
13.0:20130429T145626Z
DOWNLOAD                                  PKGS         FILES    XFER (MB)   SPEED
Completed                                  2/2       171/171    61.9/61.9    0B/s

PHASE                                          ITEMS
Removing old actions                             7/7
Installing new actions                           6/6
Updating modified actions                    170/170
Updating package state database                 Done
Updating package cache                           2/2
Updating image state                            Done
Creating fast lookup database                   Done
Reading search index                            Done
Updating search index                            2/2
```

次のコマンドは、このイメージ内で `jre-7` パッケージが更新されたことを確認しています。新しいブート環境で更新を実行する場合、`beadm mount` および `pkg -R` を使用して、新しいブート環境で次の検査を行います。

```
$ pkg list jre-7
NAME (PUBLISHER)                                  VERSION                    IFO
runtime/java/jre-7                                1.7.0.21-0.175.2.0.0.13.0  i--
```

適切な依存関係が見つからないときの制約パッケージの更新

エラーメッセージ: `A version for 'incorporate' dependency cannot be found`

制約パッケージとそれらの `incorporate` 依存関係については、14 ページの「制約パッケージ」を参照してください。

制約パッケージの更新に失敗することがある理由の例として、制約パッケージの `incorporate` 依存関係のインストール状態が、次のようになっていることがあります。

- 依存パッケージが異なるバージョンで凍結されている。
- 依存パッケージが高いバージョンですでにインストールされている。
- 依存パッケージが別のパブリッシャーからインストールされており、そのパブリッシャーが固定である。

例 27 依存関係がロック解除されていて別個に更新される場合の `pkg:/entire` の更新

次の例ではパッケージ名が指定されていないため、インストール済みのすべてのパッケージの更新が試行されます。この操作で更新を試行するインストール済みパッケージの 1 つが、`pkg:/entire` 制約パッケージです。この例は、すでに高いバージョンでインストールされている `pkg:/entire` の `incorporate` 依存関係を示しています。

```
$ pkg update --be-name s11.2
Creating Plan (Solver setup): /
pkg update: No solution was found to satisfy constraints
Plan Creation: Package solver has not found a solution to update to latest available
 versions.
This may indicate an overly constrained set of packages are installed.

latest incorporations:
[output omitted]
pkg://solaris/entire@0.5.11,5.11-0.175.2.0.0.10.0:20130318T181506Z

The following indicates why the system cannot update to the latest version:

No suitable version of required package pkg://solaris/entire@0.5.11,5.11-0.175.2
.0.0.10.0:20130318T181506Z found:
 Reject:  pkg://solaris/entire@0.5.11,5.11-0.175.2.0.0.10.0:20130318T181506Z
 Reason:  A version for 'incorporate' dependency on pkg:/consolidation/ub_javavm
/ub_javavm-incorporation@0.5.11,5.11-0.175.2.0.0.9.0 cannot be found
```

このメッセージでは、システムが更新対象としていた `pkg:/entire` 制約パッケージのバージョンが、インストールできない `ub_javavm-incorporation` パッケージのバージョンを指定しています。1 つのパッケージをインストールできないため、パッケージは一切インストールされず、更新は失敗します。

次の方法を使用すれば、`ub_javavm-incorporation` パッケージをインストールできない理由の詳細を提供できます。

- -v オプションを使用します。たとえば、詳細な出力を受け取るために、-v または -vv を使用します。
- 更新するパッケージを指定します。一般的に、詳細な情報を入力すると、詳細なメッセージが表示されます。たとえば、パッケージ FMRI に、パッケージ名に加えてバージョンを指定します。

次のコマンドでは -v オプションを指定し、上記のメッセージからコピーされた entire@0.5.11,5.11-0.175.2.0.0.10.0 パッケージへの更新が指定されています。このコマンドでは、--be-name オプションの代わりに -n オプションが指定されています。-n オプションは、実行される内容を表示しますが、このイメージを実際に変更するわけではありません。

```
$ pkg update -nv entire@0.5.11,5.11-0.175.2.0.0.10.0
Creating Plan (Solver setup): /
pkg update: No matching version of entire can be installed:
 Reject:  pkg://solaris/entire@0.5.11,5.11-0.175.2.0.0.10.0:20130318T181506Z
 Reason:  All versions matching 'require' dependency pkg:/consolidation/ub_javav
m/ub_javavm-incorporation are rejected
  Reject:  pkg://solaris/consolidation/ub_javavm/ub_javavm-incorporation@0.5.11,
5.11-0.151.0.1:20101105T053418Z
  pkg://solaris/consolidation/ub_javavm/ub_javavm-incorporation@0.5.11,5.11-0.17
5.0.0.0.2.0:20111019T144756Z
  pkg://solaris/consolidation/ub_javavm/ub_javavm-incorporation@0.5.11,5.11-0.17
5.0.10.1.0.0:20120920T143020Z
  Reason:  Excluded by proposed incorporation 'entire'
  Newer version pkg://solaris/consolidation/ub_javavm/ub_javavm-incorporation@0.
5.11,5.11-0.175.2.0.0.13.0:20130429T145201Z is already installed
  Reject:  pkg://solaris/consolidation/ub_javavm/ub_javavm-incorporation@0.5.11,
5.11-0.175.2.0.0.9.0:20130304T213739Z
  Reason:  Newer version pkg://solaris/consolidation/ub_javavm/ub_javavm-incorpo
ration@0.5.11,5.11-0.175.2.0.0.13.0:20130429T145201Z is already installed
 Reject:  pkg://solaris/consolidation/ub_javavm/ub_javavm-incorporation@0.5.11,5
.11-0.175.2.0.0.13.0:20130429T145201Z
 Reason:  Excluded by proposed incorporation 'entire'
```

これらのメッセージは、このイメージに現在インストールされている ub_javavm-incorporation パッケージのバージョンが、更新操作によってインストールされる pkg:/entire 制約パッケージによって指定されたバージョンよりも新しいことを示しています。

次のコマンドは、現在インストールされている ub_javavm-incorporation パッケージのバージョンを表示します。

```
$ pkg list ub_javavm-incorporation
NAME (PUBLISHER)                                  VERSION                    IFO
consolidation/ub_javavm/ub_javavm-incorporation   0.5.11-0.175.2.0.0.13.0    i--
```

次のコマンドは、システムが更新対象として指定しようとした pkg:/entire 制約パッケージによって指定されている ub_javavm-incorporation パッケージのバージョンを示しています。システムが更新対象として指定しようとした pkg:/entire 制約パッケージのバージョンは、上記の出力の先頭にある「Reject」メッセージからのコピーです。-r オプションは、インストール済みイメージではなく構成済みのパッケージリポジトリからこのパッケージを検索します。

```
$ pkg contents -Hrt depend \
```

```
-a facet.version-lock.consolidation/ub_javavm/ub_javavm-incorporation=true \
-o fmri entire@0.5.11,5.11-0.175.2.0.0.10.0
consolidation/ub_javavm/ub_javavm-incorporation@0.5.11-0.175.2.0.0.9.0
```

この問題を修正するために、指定された特定のパッケージを除くすべてのインストール済みパッケージを更新するように更新操作に指示できます。`pkg update` コマンドで 1 つ以上の `--reject` オプションを使用して、`--reject` オプションで指定されたパッケージの更新を試行せずに更新を実行します。`--reject` 引数にはワイルドカードを使用できます。次のコマンドで、拒否するパッケージは上記の「Reason: Newer version is already installed」メッセージの前にある、「Reject」メッセージからコピーされたものです。

```
$ pkg update -v --be-name s11.2 \
--reject 'consolidation/ub_javavm/ub_javavm-incorporation@0.5.11,5.11-0.151*' \
--reject 'consolidation/ub_javavm/ub_javavm-incorporation@0.5.11,5.11-0.175.0*' \
--reject 'consolidation/ub_javavm/ub_javavm-incorporation@0.5.11,5.11-0.175.2.0.0.9.0*'
```

ヒント - `--reject` 引数を指定するときは注意が必要です。すでにインストール済みで、`--reject` 引数に一致するすべてのパッケージはアンインストールされます。

インストール済みの依存関係が許容されないときの制約パッケージの更新

エラーメッセージ:

- `The installed package` *package* `is not permissible`
- `Excluded by proposed incorporation`

これらのエラーメッセージを受け取った場合、*package* はおそらく、ほかのコアのオペレーティングシステムパッケージと同期して保持する必要があるコアのオペレーティングシステムパッケージです。`pkg facet` コマンドを使用して、`facet.version-lock.`*package* ファセットの値を確認してください。この `version-lock` ファセットの値が `false` の場合、`pkg change-facet` コマンドを使用して、このファセットの値を `true` に変更し、更新操作を再試行してください。

必要なパッケージが見つからない

エラーメッセージ: `A version for 'require' dependency cannot be found`

必要なパッケージが見つからないという次のメッセージのようなメッセージを受け取る場合、構成済みのパブリッシャーからパッケージが使用可能であるかどうかを確認します。

```
pkg update: No solution was found to satisfy constraints
```

```
Plan Creation: Package solver has not found a solution to update to
latest available versions.
This may indicate an overly constrained set of packages are installed.
[output omitted]
No suitable version of required package package1 found:
Reject: package1
Reason: A version for 'require' dependency package2 cannot be found
```

package2 が構成済みのパブリッシャーから使用可能であるかどうかを表示するには、次のコマンドを使用します。

```
$ pkg list -afv package2
```

パブリッシャーの起点の場所を確認するには、`pkg publisher` コマンドを使用します。このパッケージのパブリッシャーは、完全な FMRI の `pkg://` の後に示されます。パブリッシャーの起点の場所を変更することが必要な場合もあります。場所がローカルパッケージリポジトリのとき、リポジトリの更新が必要な場合もあります。

必要なパッケージが拒否された

エラーメッセージ:

- `No solution was found to satisfy constraints`
- `All versions matching 'require' dependency` *package* `are rejected`

`pkg update` コマンドからの次の出力で、最初のエラーメッセージでは、必要なパッケージ `desktop-incorporation` の適切なバージョンが見つからないことが示されています。`desktop-incorporation` パッケージが適切でないのは、そのいずれかの依存パッケージが見つからないためです。適切なバージョンが見つからないため、`desktop-incorporation` パッケージは拒否されます。`desktop-incorporation` パッケージが `pkg:/entire` 制約パッケージによって必要とされているため、`pkg update` は失敗します。次のコマンドでは、`desktop-incorporation` パッケージが `pkg:/entire` 制約パッケージによって必要とされていることが示されています。

```
$ pkg search -Hlo pkg.name require:consolidation/desktop/desktop-incorporation
```

その次のエラーメッセージでは、必要なパッケージ `python-extra-26` の適切なバージョンが見つからないことが示されています。`python-extra-26` パッケージが適切でない理由は、`python-extra-26` パッケージには `desktop-incorporation` パッケージが必要ですが、`desktop-incorporation` の適切なバージョンが見つからなかったためです。

この情報により、この更新の失敗の解決策は、`desktop-incorporation` パッケージの必要なバージョンをインストールする方法を見つけることであるということがわかります。

```
pkg update: No solution was found to satisfy constraints

maintained incorporations:
```

```
   [output omitted]

Plan Creation: dependency error(s) in proposed packages:
  [output omitted]
  No suitable version of required package pkg://solaris/consolidation/desktop
/desktop-incorporation@0.5.11,5.11-0.175.2.0.0.26.0:20131028T145233Z found:
    Reject: pkg://solaris/consolidation/desktop/desktop-incorporation
@0.5.11,5.11-0.175.2.0.0.26.0:20131028T145233Z
    Reason: A version for 'incorporate' dependency on pkg:/library/python-2
/python-sexy-26@0.1.9-0.175.0.0.0.1.0 cannot be found
  No suitable version of required package pkg://solaris/library/python-2
/python-extra-26@2.6.4-0.175.1.0.0.15.0:201205014T200156Z found:
    Reject:  pkg://solaris/library/python-2/python-extra-26@
2.6.4-0.175.1.0.0.15.0:201205014T200156Z
    Reason: All versions matching 'require' dependency pkg:/consolidation
/desktop/desktop-incorporation are rejected
      Reject: pkg://solaris/consolidation/desktop/desktop-incorporation
@0.5.11,5.11-0.175.0.0.0.2.0:20111019T132128Z
  [output omitted]
pkg://solaris/consolidation/desktop/desktop-incorporation
@0.5.11,5.11-0.175.2.0.0.26.0:20131028T145233Z
```

次のコマンドは、必要な desktop-incorporation パッケージについて見つからないパッケージが、必要ではないことを示しています。python-sexy-26 パッケージは desktop-incorporation パッケージによる incorporation 対象となっていますが、必要ではありません。

```
$ pkg search -Hlo pkg.name require:library/python-2/python-sexy-26
$ pkg search -Hlo pkg.name incorporate:library/python-2/python-sexy-26
consolidation/desktop/desktop-incorporation
```

incorporate 依存関係は、python-sexy-26 パッケージがインストールされる場合、指定されたバージョンでインストールされなければならないことを示しています。ただし、python-sexy-26 パッケージを必要とするパッケージが存在しないため、python-sexy-26 パッケージをインストールする必要はありません。したがって、この更新の失敗の 1 つの解決策は、python-sexy-26 パッケージをアンインストールすることです。このパッケージの別のバージョンが現在インストールされており、更新では desktop-incorporation 制約パッケージによって指定されたバージョンが見つかりませんでした。パッケージをアンインストールすると、更新処理では更新されたパッケージを見つける必要がなくなります。

python-sexy-26 パッケージが必要であってアンインストールすることを望まない場合、pkg:/library/python-2 /python-sexy-26@0.1.9-0.175.0.0.0.1.0 を提供するパッケージリポジトリを検索します。pkg set-publisher コマンドを使用して、そのリポジトリをパブリッシャーの起点の場所に追加するか、pkgrecv コマンドを使用して、そのパッケージを現在設定されているパブリッシャーの起点に追加します。

パッケージが期待どおりに更新されない

エラーメッセージ: pkg update: The installed package *package* is not permissible.

更新操作でワイルドカードを使用するとき、更新されるはずの一部のパッケージが更新されない場合にエラーメッセージがまったく表示されないことがあります。ワイルドカードを使用せずにパッケージ名を指定すると、エラーメッセージが表示される場合があります。

たとえば、新しいパッケージが構成済みのパブリッシャーから使用可能であることが確認されたにもかかわらず、次の操作がエラーを出さずに完了し、すべてのパッケージを更新しないことがあります。

```
$ pkg update '*'
```

詳細な情報を取得するには、ワイルドカードを使用する代わりに、更新後の `pkg:/entire` のバージョンを指定します。

```
$ pkg list -Hafv entire
pkg://solaris/entire@0.5.11,5.11-0.175.2.0.0.9.0:20130304T214506Z  ---
$ pkg update -nv pkg://solaris/entire@0.5.11,5.11-0.175.2.0.0.9.0:20130304T214506Z
Creating Plan (Solver setup): -
pkg update: The installed package compress/zip is not permissible.
  Reject:  pkg://solaris/compress/zip@3.0,5.11-0.175.2.0.0.7.0:20121119T070339Z
  Reason:  Excluded by proposed incorporation 'consolidation/userland/userland-
incorporation'
```

この例では、パッケージ `compress/zip` が `userland-incorporation` 制約パッケージの更新を妨げ、それが `pkg:/entire` 制約パッケージの更新を妨げています。次のコマンドは、`compress/zip` パッケージの詳細情報を表示します。

```
$ pkg list compress/zip
NAME (PUBLISHER)  VERSION                    IFO
compress/zip      3.0-5.11-0.175.2.0.0.7.0  if-
```

この出力は、`compress/zip` パッケージが凍結されていることを示します。パッケージは凍結されているため更新できません。

```
$ pkg unfreeze compress/zip
compress/zip was unfrozen.
```

`compress/zip` パッケージの凍結を解除した場合、元の `pkg update '*'` 操作によって、更新が使用可能なイメージ内のすべてのパッケージが更新されるはずです。`pkg freeze` および `pkg unfreeze` コマンドについては、113 ページの「指定したバージョンへのパッケージのロック」を参照してください。

156 ページの「同期リンクされたパッケージをインストールできない」に示す `pkg sync-linked` エラーも類似していますが、非大域ゾーン内のパッケージが更新を妨げていることを示しています。

同期リンクされたパッケージをインストールできない

エラーメッセージ: `pkg sync-linked: The installed package` *package* `is not permissible.`

親イメージとのバージョンの不一致という理由でパッケージの拒否が表示される場合があります。

```
Linked progress: pkg: update failed (linked image exception(s)):

A 'sync-linked' operation failed for child 'zone:z1' with an unexpected
return value of 1 and generated the following output:

pkg sync-linked: The installed package package is not permissible.
  Reject:  package
  Reason:  Parent image has a incompatible newer version: package
```

次の理由により、この非互換性メッセージを受け取ることがあります。

- コアのオペレーティングシステムパッケージは、非大域ゾーンと大域ゾーンとで同じバージョンにする必要があります。これらのパッケージを非大域ゾーンで個別に更新することはできません。同様に、1 つの非大域ゾーン内でこれらのいずれかのパッケージを凍結すると、大域ゾーンとすべての非大域ゾーンで更新が失敗します。

  次のコマンドは、大域ゾーンと非大域ゾーンの間で同期を維持する必要があるパッケージを一覧表示します。

  ```
  $ pkg search -o pkg.name :depend:parent:
  ```

- ゾーンが構成されている BE を更新するとき、その BE をマウントし、`pkg -R` コマンドを使用して代替 BE (ABE) を更新するという操作は、ABE のパブリッシャー構成が、現在ブートされている BE のパブリッシャー構成と異なる場合は実行できません。ABE の非大域ゾーンでは、現在アクティブな BE からのパブリッシャー構成が使用されます。

158 ページの「非大域ゾーンをインストールできない」も参照してください。

子イメージで一時的な起点を使用できない

エラーメッセージ: `pkg install: The proposed operation on this parent image can not be performed because temporary origins were specified and this image has children.`

このメッセージで、子は非大域ゾーンです。親イメージは大域ゾーンです。`pkg install` コマンドまたは `pkg update` コマンドに `-g` オプションを使用して一時的な起点が指定されました。このイメージに非大域ゾーンがある場合、`-g` オプションは使用できません。

`-g` オプションを使用して一時的な起点を指定する代わりに、`pkg set-publisher` コマンドを使用して大域ゾーン内でそのパブリッシャーにその起点を追加してください。このパブリッシャーが大域ゾーン内で構成されている場合、非大域ゾーンはシステムリポジトリを介して起点にアクセスできます。

非大域ゾーンをインストールできない

エラーメッセージ:

- `The following pattern(s) did not match any allowable packages. Try using a different matching pattern, or refreshing publisher information`
- `Linked progress: pkg: update failed (linked image exception(s)): pkg sync-linked: No solution was found to satisfy constraints`

コアのオペレーティングシステムパッケージは、非大域ゾーンと大域ゾーンとで同じバージョンにする必要があります。このイメージ内の `solaris` パブリッシャーの起点が、大域ゾーンにインストールされたシステムパッケージと同じバージョンを含まないパッケージリポジトリに設定された場合、非大域ゾーンをインストールしようとすると、次のエラーになります。

```
$ zoneadm -z myzone install
The following ZFS file system(s) have been created:
    rpool/VARSHARE/zones/myzone
Progress being logged to /var/log/zones/zoneadm.20160206T181301Z.myzone.install
       Image: Preparing at /system/zones/myzone/root.

Install Log: /system/volatile/install.4606/install_log
AI Manifest: /tmp/manifest.xml.9daq.i
SC Profile: /usr/share/auto_install/sc_profiles/enable_sci.xml
  Zonename: myzone
Installation: Starting ...

            Creating IPS image
Startup linked: 1/1 done
            Installing packages from:
                solaris
                    origin:  http://pkg.oracle.com/solaris/release/
            Error occurred during execution of 'generated-transfer-4606-1' checkpoint.
            Failed Checkpoints:

            Checkpoint execution error:

                The following pattern(s) did not match any allowable packages. Try
                using a different matching pattern, or refreshing publisher information:

Installation: Failed.  See install log at /system/volatile/install.4606/install_log
ERROR: auto-install failed.
```

非大域ゾーンをインストールするには、`solaris` パブリッシャーの起点として設定するリポジトリが、非大域ゾーンのインストール場所となる大域ゾーン内にインストールされているものと同じシステムソフトウェアを最低限含んでいる必要があります。リポジトリには古いソフトウェアまたは新しいソフトウェアを含めることもできますが、大域ゾーンにインストールされているものと同じソフトウェアを含む必要があります。次のコマンドは、大域ゾーンにインストールされている `pkg:/entire` パッケージと同じバージョンが `/var/share/pkgrepos/solaris` リポジトリに含まれているため、このリポジトリはこの大域ゾーンの適切なパブリッシャーの起点であることを示しています。

```
$ pkg list entire
NAME (PUBLISHER)      VERSION                    IFO
```

```
entire                0.5.11-0.175.2.0.0.26.0   i--
$ pkgrepo list -Hs /var/share/pkgrepos/solaris entire@0.5.11-0.175.2.0.0.26.0
solaris       entire       0.5.11-0.175.2.0.0.26.0:20131028T190148Z
$ pkg set-publisher -G '*' -M '*' -g /var/share/pkgrepos/solaris/ solaris
```

パブリッシャーが非大域ゾーン内で構成された場合、そのパブリッシャーが大域ゾーン内で構成されていなくても、そのパブリッシャーについてのすべての場所が大域ゾーンからアクセス可能である必要があります。

`svc:/application/pkg/system-repository:default` および `svc:/application/pkg/zones-proxyd:default` サービスが大域ゾーン内で `online` になっていることを確認します。`svc:/application/pkg/zones-proxy-client:default` サービスが非大域ゾーン内で `online` になっていることを確認します。

156 ページの「同期リンクされたパッケージをインストールできない」も参照してください。

イメージを修正できない

エラーメッセージ: `pkg: The image cannot be modified as it is currently in use by another package client`

次の例に示すように、エラーメッセージには、イメージがロックされているパッケージクライアントの名前と PID が含まれているはずです。

```
pkg: The image cannot be modified as it is currently in use by another package
client: pkg on cbusl0406l, pid 26604.
```

現在のパッケージ処理 (この例では処理 26604) が終了したあとで `pkg` コマンドを再試行してください。処理がすぐに終了しない場合、たとえば `ptree -a` を使用して、処理の調査を開始します。

このイメージについて現在動作中のパッケージ処理は、更新が使用可能かどうかを検査する更新マネージャーである場合があります。更新マネージャーの `cron` ジョブと `pkg/update` サービスの詳細については、166 ページの「更新マネージャーの使用」 を参照してください。

ファイルが回収された

次の情報メッセージの後に、改修されたファイルのパスと、ファイルの一時的な移動先が続きます。

```
The following unexpected or editable files and directories were
salvaged while executing the requested package operation; they
have been moved to the displayed location in the image:
```

これは情報メッセージのため無視してもかまいませんが、必要に応じて対応することもできます。

ディレクトリは、IPS で参照カウントされます。イメージ内にインストールされていないパッケージがあるディレクトリを明示的または暗黙的に参照している場合、そのディレクトリは削除されます。そのディレクトリにパッケージ解除されたファイルシステムオブジェクトが含まれている場合、それらの項目は `$IMAGE_META/lost+found` に移動されます。パッケージ解除されたファイルシステムオブジェクトは、IPS パッケージで配信されなかったファイルおよびディレクトリです。`IMAGE_META` の値は通常 `/var/pkg` です。`IMAGE_META` ディレクトリについての詳細は、`pkg`(5) のマニュアルページのファイルに関するセクションを参照してください。

これらのファイルが移動されないようにするには、パッケージ解除されたデータをパッケージ化されたディレクトリに格納しないようにします。たとえば、ユーザーデータは `/var/`*app-name* 内のディレクトリなどの別個のデータセットに格納します。

格納されるイメージメタデータの最小化

注記 - Oracle サポートまたは正式な Oracle ドキュメントによって明示的に指示された場合を除き、`/var/pkg` ディレクトリ内は一切変更しないでください。`/var/pkg` またはその内容を変更すると、サポートされていない使用不能なシステムになる可能性があります。

`/var/pkg` ディレクトリから手動で削除しないでください。`/var/pkg` ディレクトリはイメージのメタデータを保持しており、たとえば、イメージ状態とキャッシュカタログを管理するために使用されます。

`/var/pkg` ディレクトリは非常に大きくなる可能性があります。`flush-content-cache-on-success` イメージプロパティーの値が `true` に設定されていることを確認します。`flush-content-cache-on-success` プロパティーの値はデフォルトで `true` です。`flush-content-cache-on-success` の値が `true` の場合は、`pkg install` および `pkg update` の操作が正常に完了したとき、キャッシュ済みファイルが削除されます。`flush-content-cache-on-success` プロパティーが `false` に設定されている場合、次のコマンドを使用して、値を `true` にリセットできます。

```
$ pkg property flush-content-cache-on-success
PROPERTY                        VALUE
flush-content-cache-on-success False
$ pkg set-property flush-content-cache-on-success true
$ pkg property -H flush-content-cache-on-success
flush-content-cache-on-success True
```

内容キャッシュのフラッシュ (`flush-content-cache-on-success` を `true` に設定すること) によって、一部の `pkg` 操作が完了するまでの時間が長くなることがあります。

非大域ゾーンには別のキャッシュがあり、/usr/lib/pkg.sysrepo の -c オプションで設定できます。-s オプションで、このキャッシュの最大サイズを設定できます。pkg.sysrepo(1M) のマニュアルページを参照してください。

パッケージのインストールパフォーマンスの増大

次のステップは、パッケージのインストールおよび更新のパフォーマンスの増大に役立ちます。

- ZFS ストレージプールの容量が 90% より低いことを確認します。

```
$ zpool list
NAME  SIZE ALLOC FREE CAP DEDUP HEALTH ALTROOT
rpool 186G 75.2G 111G 40% 1.00x ONLINE -
```

- ローカルパッケージリポジトリを使用します。『Copying and Creating Package Repositories in Oracle Solaris 11.3』を参照してください。
- http_proxy が設定されている場合、プロキシのパフォーマンスを確認します。
- 多数の LDOM など、多数のシステムをインストールまたは更新する場合は、Apache Web サーバーを構成し、「Configuring a Simple Prefixed Proxy」 in 『Copying and Creating Package Repositories in Oracle Solaris 11.3』の説明に従って、pkg/threads プロパティーを設定します。

♦ ♦ ♦ 付 録 B

IPS のグラフィカルユーザーインタフェース

IPS には、2 つのグラフィカルユーザーインタフェース (GUI) ツールが含まれます。

- パッケージマネージャーでは、パッケージおよびパブリッシャーに対するほとんどの操作と、ブート環境 (BE) に対する一部の操作を実行できます。Oracle Solaris OS および IPS のテクノロジーに不慣れな場合、パッケージマネージャーを使用すると、パッケージをすばやく識別してインストールできます。
- 更新マネージャーは、利用可能な更新があるイメージ内のすべてのパッケージを更新します。

パッケージマネージャーの使用

パッケージマネージャーは、コマンド行から実行できるタスクのサブセットを提供します。

- パッケージの一覧表示、検索、インストール、更新、削除
- パッケージソースの追加と構成
- BE のアクティブ化、名前変更、削除

次のいずれかの方法でパッケージマネージャーを開始します。

ツールバー	ツールバーの「パッケージマネージャー」アイコンをクリックします。「パッケージマネージャー」アイコンは回転する矢印の付いたボックスです。
デスクトップアイコン	デスクトップの「パッケージマネージャー」アイコンをダブルクリックします。
メニューバー	「システム」->「システム管理」->「パッケージマネージャー」を選択します。
コマンド行	`$ packagemanager &`

パッケージマネージャーの完全なドキュメントについては、「パッケージマネージャー」メニューバーの「ヘルプ」->「内容」を選択してください。

パッケージマネージャーのコマンド行のオプション

packagemanager(1) コマンドでは、次のオプションがサポートされています。

表 1 パッケージマネージャーコマンドのオプション

オプション	説明
`--image-dir` または `-R` *dir*	*dir* をルートとするイメージを操作します。デフォルトの動作では、現在のイメージを操作します。 次のコマンドは、`/aux0/example_root` に格納されたイメージを操作します。 `$ packagemanager -R /aux0/example_root`
`--update-all` または `-U`	利用可能な更新があるすべてのインストール済みパッケージを更新します。このオプションを指定することは、パッケージマネージャーの GUI で「更新」オプションを選択することと同じです。すべてのパッケージの更新については、166 ページの「更新マネージャーの使用」を参照してください。
`--info-install` または `-i` *file*`.p5i`	パッケージマネージャーを Web インストールモードで実行するための `.p5i` ファイルを指定します。指定するファイルの拡張子が `.p5i` である必要があります。詳細は、164 ページの「Web インストールの使用」を参照してください。
`--help` または `-h`	コマンドの使用方法に関する情報を表示します。

Web インストールの使用

Web インストール処理の詳細は、パッケージマネージャーのヘルプを参照してください。

パッケージマネージャーでは、クリック 1 つで簡単に実行できる Web インストール処理を使用したパッケージのインストールがサポートされています。Web インストール処理では `.p5i` ファイルが使用されます。`.p5i` ファイルには、パブリッシャーと、それらのパブリッシャーからインストール可能なパッケージを追加するための情報が格納されます。`.p5i` ファイルの情報は、Web インストール処理によって読み取られて使用されます。

▼ Web インストールファイルを作成する方法

システムにインストールしたパッケージをほかのユーザーがインストールできるようにするために、Web インストール処理を使用して、それらのパッケージファイルのイ

ンストール命令をエクスポートすることができます。Web インストール処理が作成する .p5i ファイルは、インストールするパッケージとパブリッシャーのインストール命令で構成されます。

1. **パブリッシャーを選択します。**
   パッケージマネージャーの「パブリッシャー」ドロップダウンメニューから、.p5i ファイルに含めるパッケージのパブリッシャーを選択します。

2. **パッケージを選択します。**
   パッケージマネージャーのパッケージリストペインで、インストール命令を配布するパッケージを選択します。

3. **選択内容をエクスポートします。**
   「ファイル」->「選択項目をエクスポート」を選択して、「エクスポート選択項目の確認」ウィンドウを表示します。

4. **選択内容を確認します。**
   「OK」ボタンをクリックして選択項目を確認します。「選択項目をエクスポート」ウィンドウが表示されます。

5. **(オプション) ファイル名を変更します。**
   .p5i ファイルのデフォルトの名前が表示されます。このファイル名は変更できます。
   .p5i 拡張子を変更しないでください。

6. **(オプション) ファイルの場所を変更します。**
   .p5i ファイルのデフォルトの場所が表示されます。この場所は変更できます。

7. **Web インストールファイルを保存します。**
   「保存」ボタンをクリックして、ファイルの名前と場所を保存します。

▼ Web インストールを使用したパブリッシャーの追加とパッケージをインストールする方法

Web インストール処理を使用すると、.p5i ファイルからパッケージをインストールできます。デスクトップ上のファイルまたは Web サイト上のファイルを使用できます。

1. **次のいずれかの方法を使用して Web インストールモードでパッケージマネージャーを起動します。**

   - **デスクトップ上の .p5i ファイルを選択します。**

- **パッケージマネージャーをコマンド行から起動し、`.p5i` ファイルを指定します。**

  ```
  $ packagemanager ./wifile.p5i
  ```

- **`.p5i` ファイルへのリンクを含む URL の場所にアクセスします。**
  - `.p5i` ファイルが置かれている Web サーバーでこの MIME タイプが登録済みの場合、`.p5i` ファイルへのリンクをクリックするだけです。
  - `.p5i` ファイルが置かれている Web サーバーでこの MIME タイプが未登録の場合、`.p5i` ファイルをデスクトップに保存してそのファイルを選択します。

「インストール/更新」ウィンドウが表示されます。ウィンドウ上部のラベルは「パッケージマネージャー Web インストーラ/次のものがシステムに追加されます:」です。インストールされるパブリッシャーとパッケージの一覧が表示されます。

2. **インストールを続けるには「続行」ボタンをクリックします。**
   指定されたパッケージパブリッシャーがシステムでまだ構成されていない場合、「パブリッシャーを追加」ウィンドウが表示されます。パブリッシャーの名前と URI はすでに入力されています。
   追加するパブリッシャーがセキュアなパブリッシャーである場合、SSL 鍵と証明書が必要です。システム上の「SSL キー」と「SSL 証明書」を参照して指定します。
   パブリッシャーが正常に追加されると、「パブリッシャーを追加しました」ダイアログが表示されます。

3. **インストールを続けるには「OK」ボタンをクリックします。**

4. **無効化されたパブリッシャーを有効にします。**
   無効なパブリッシャーからのパッケージが `.p5i` ファイルに含まれている場合、Web インストールでは「パブリッシャーの有効化」ダイアログが開きます。このダイアログを使用してパブリッシャーを有効にし、パッケージをインストールできるようにします。
   「インストール/更新」ウィンドウは、パッケージマネージャーの「インストール/更新」オプションを選択した場合と同じ表示になります。
   すべてのパッケージがインストールされると、アプリケーションは閉じます。

更新マネージャーの使用

更新マネージャーは、すべてのインストール済みパッケージを、システムの制約で許可された最新バージョンに更新します。この制約は、インストール済みパッケージの依存関係およびパブリッシャーの構成によってシステムに課せられるものです。この機能は、次の方法で実行される機能と同じ働きをします。

- パッケージマネージャー GUI で、「更新」ボタンを選択するか、「パッケージ」-「更新」メニューオプションを選択します。
- packagemanager コマンドを使用します。

```
$ packagemanager --update-all
```

- pkg コマンドを使用します。

```
$ pkg update
```

次のいずれかの方法で更新マネージャーを開始します。

ステータスバー　更新が利用可能な場合、ステータスバーに通知が表示されます。通知で指示されている場所をクリックします。更新マネージャーアイコンは、3 つの箱が積み重なったものです。

メニューバー　「システム」->「システム管理」->「更新マネージャー」を選択します。

コマンド行　**$ pm-updatemanager**

自動　更新マネージャーパッケージ package/pkg/update-manager は、cron ジョブ /usr/lib/update-manager/update-refresh.sh を生成します。

```
30 0,9,12,18,21 * * * /usr/lib/update-manager/update-refresh.sh
```

SMF サービス svc:/application/pkg/update がオンラインのとき、この cron ジョブは、構成されたパブリッシャーから利用可能な更新パッケージを定期的に確認します (次に示すプロセスの最初の 2 手順)。更新パッケージが利用可能な場合、デスクトップのツールバーに通知が表示されます。通知アイコンを選択すると、更新マネージャー GUI が開きます。

「更新」ウィンドウが表示され、更新処理が始まります。

1. システムはすべてのカタログをリフレッシュします。
2. システムはすべてのインストール済みパッケージを評価し、パッケージの更新が利用可能かどうかを調べます。
   - 更新が利用可能なパッケージがない場合、「有効な更新はありません」というメッセージが表示され、処理は停止します。
   - パッケージの更新が利用可能な場合、更新されるパッケージの一覧が確認のために表示されます。これは、「キャンセル」ボタンをクリックして更新を中止する最後の機会です。

   更新を続けるには「続行」ボタンをクリックします。
3. システムはすべてのパッケージの更新をダウンロードし、インストールします。

次のパッケージの更新が利用可能な場合、これらのパッケージが最初に更新されます。その後、ほかのパッケージが更新されます。

```
package/pkg
package/pkg/package-manager
package/pkg/update-manager
```

デフォルトでは、各パッケージは、最初にインストールされたときの取得元であるパブリッシャーから更新されます。元のパブリッシャーが非固定の場合、パッケージの新しいバージョンがこのイメージと互換性があるものであれば、別のパブリッシャーからのパッケージをインストールできます。パブリッシャーを固定または非固定として設定するには、パッケージマネージャーの「パブリッシャーの管理」ウィンドウまたは `pkg set-publisher` コマンドを使用します。

更新処理中にエラーが発生した場合、「詳細」パネルが展開され、エラーの詳細が表示されます。問題が発生した段階の隣に、エラーステータスのインジケータが表示されます。

4. 更新されるパッケージとイメージポリシーに応じて、新しい BE が作成されることがあります。

   更新に伴ってシステムが新しい BE を作成した場合、デフォルトの BE の名前を編集できます。

   再ブートして新しい BE にブートする必要があります。新しい BE はブート時のデフォルトの選択肢になります。現在の BE は、代替のブート選択肢になります。

   - システムをすぐに再起動するには「今すぐ再起動」ボタンをクリックします。
   - システムをあとから再起動するには「あとで再起動」ボタンをクリックします。

更新マネージャーのコマンド行のオプション

`pm-updatemanager(1)` コマンドでは、次のオプションがサポートされています。

表 2 更新マネージャーコマンドのオプション

オプション	説明
`--image-dir` または `-R` *dir*	*dir* をルートとするイメージを操作します。デフォルトの動作では、現在のイメージを操作します。 次のコマンドは、`/aux0/example_root` にあるイメージを更新します。 `$ pm-updatemanager -R /aux0/example_root`
`--help` または `-h`	コマンドの使用方法に関する情報を表示します。

索引

U

V

W

Z