

# Oracle® Solaris 11.4 のシステムの更新およびソフトウェアの追加

ORACLE®

Part No: E75002-01  
2018 年 8 月



## Part No: E75002-01

Copyright © 2007, 2018, Oracle and/or its affiliates. All rights reserved.

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

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

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

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

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

OracleおよびJavaはオラクル およびその関連会社の登録商標です。その他の社名、商品名等は各社の商標または登録商標である場合があります。

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

このソフトウェアまたはハードウェア、そしてドキュメントは、第三者のコンテンツ、製品、サービスへのアクセス、あるいはそれらに関する情報を提供することがあります。適用されるお客様とOracle Corporationとの間の契約に別段の定めがある場合を除いて、Oracle Corporationおよびその関連会社は、第三者のコンテンツ、製品、サービスに関して一切の責任を負わず、いかなる保証もいたしません。適用されるお客様とOracle Corporationとの間の契約に定めがある場合を除いて、Oracle Corporationおよびその関連会社は、第三者のコンテンツ、製品、サービスへのアクセスまたは使用によって損失、費用、あるいは損害が発生しても一切の責任を負いかねます。

### 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

---

このドキュメントの使用方法 .....	13
<b>1 Image Packaging System の概要 .....</b>	<b>15</b>
Image Packaging System .....	15
IPS の概念 .....	16
IPS パッケージ .....	16
障害管理リソース識別子 .....	18
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 .....	21
リポジトリの起点とミラー .....	22
イメージとブート環境 .....	22
パッケージのファセットとバリエーション .....	23
インストール権限 .....	24
<b>2 ソフトウェアパッケージに関する情報の取得 .....</b>	<b>25</b>
パッケージのインストール状態情報の表示 .....	25
インストール済みのパッケージ .....	26
インストール可能なパッケージ .....	26
最新のパッケージ .....	26
更新が利用可能なパッケージ .....	27
使用可能なすべてのパッケージ .....	27
パッケージの名前変更と廃止 .....	28
特定バージョンに凍結されているパッケージ .....	29
パッケージの説明またはライセンスの表示 .....	29
パッケージの説明、パッケージ化およびインストールされた時期、完全な FMRI の表示 .....	30
オープンソースのためのプロジェクトおよびソース位置の表示 .....	31
パッケージライセンスの表示 .....	31
パッケージマニフェストからの情報の表示 .....	32
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 .....	32
パス以外の情報の表示 .....	33

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 .....	36
ライセンス要件の表示 .....	37
パッケージの検索 .....	38
pkg search コマンドと pkg contents コマンドの比較 .....	38
検索クエリーの指定 .....	39
指定されたファイルを提供するパッケージの識別 .....	41
指定した SMF サービスを提供するパッケージの識別 .....	42
指定されたユーザーを提供するパッケージの識別 .....	42
指定された修正を提供するパッケージの識別 .....	43
分類またはカテゴリ別のパッケージの一覧表示 .....	43
依存パッケージの表示 .....	44
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 .....	44
<b>3 ソフトウェアパッケージのインストールおよび更新 .....</b>	<b>47</b>
操作のプレビュー .....	48
パッケージインストールのプレビュー .....	48
ファセットの変更のプレビュー .....	49
パッケージのインストールおよび更新 .....	50
一般的なインストールオプション .....	51
新しいパッケージのインストール .....	55
新しいブート環境へのパッケージのインストール .....	58
パッケージの拒否 .....	59
パッケージの更新 .....	60
パッケージのダウングレード .....	61
インストール済みパッケージの問題の修正 .....	61
pkg fix コマンドと pkg revert コマンドの比較 .....	61
パッケージの検証と検証エラーの修正 .....	62
ファイルシステムの内容の検証 .....	64
パッケージ解除されたファイルシステムの内容の識別 .....	65
ファイルの復元 .....	65
パッケージのアンインストール .....	67
イメージの再インストール .....	70
非大域ゾーンの操作 .....	72
大域ゾーンと非大域ゾーンの関係 .....	72
システムリポジトリおよびプロキシサービス .....	74
複数の非大域ゾーンの同時更新 .....	76

<b>4 Oracle Solaris イメージの更新またはアップグレード</b> .....	79
イメージの更新の概要 .....	79
▼ Oracle Solaris 11 システムを更新する方法 .....	80
イメージ更新のベストプラクティス .....	81
使用可能なバージョンのチェック .....	82
更新操作のプレビュー .....	84
新しいブート環境の指定 .....	85
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 .....	86
インストールするバージョンの指定 .....	86
更新前のバージョン制約の指定 .....	87
Oracle Solaris 制約パッケージの使用 .....	88
カスタム制約パッケージのインストール .....	89
Oracle Solaris 11.4 用の別のパッケージリポジトリの維持 .....	95
イメージのダウングレード .....	96
サポート更新の適用 .....	96
サポート更新へのアクセス .....	98
クリティカルパッチ更新パッケージ .....	99
SPARC システムのプラットフォームファームウェアの更新 .....	100
IDR カスタムソフトウェア更新のインストール .....	102
IDR のインストール .....	103
置換用 IDR とサポート更新のインストール .....	108
IDR の削除 .....	110
<b>5 インストールされるイメージの構成</b> .....	113
パブリッシャーの構成 .....	113
パブリッシャー情報の表示 .....	114
パッケージパブリッシャーの追加、変更、削除 .....	115
プロキシの指定 .....	121
オプションのコンポーネントのインストールの制御 .....	123
ファセットおよびバリエーションの値がパッケージのインストールに与える影響 .....	123
バリエーション値の表示と変更 .....	124
ファセット値の表示と変更 .....	126
指定したバージョンへのパッケージのロック .....	129
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 .....	130
デフォルトのアプリケーション実装の指定 .....	133
メディエーション内の参加者の識別 .....	133
現在利用できないメディエーション参加者の識別 .....	135

使用できない優先パスの設定: 結果と回復 .....	135
優先アプリケーションの変更 .....	137
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 .....	138
イメージとパブリッシャーのプロパティの構成 .....	140
ブート環境ポリシーイメージのプロパティ .....	141
パッケージの署名のプロパティ .....	143
追加のイメージのプロパティ .....	145
イメージのプロパティの設定 .....	147
イメージの作成 .....	148
操作履歴の表示 .....	149
<b>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 .....</b>	<b>153</b>
トラブルシューティングの初期手順 .....	153
pkg:/entire のインストール済みバージョンの確認 .....	155
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 .....	156
インストールの再試行 .....	158
パブリッシャーまたはリポジトリにアクセスできない .....	160
Oracle Enterprise Manager Ops Center のパブリッシャーの構成 .....	160
パッケージリポジトリにアクセスできない .....	160
SSL 証明書の問題 .....	161
場所が見つからない .....	164
サービスが使用不可 .....	165
使用可能な更新がない .....	166
パッケージをインストールできない .....	167
制約を満たすことができない .....	167
制約パッケージによって制約されたパッケージの更新 .....	168
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 .....	172
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 .....	174
必要なパッケージが見つからない .....	174
必要なパッケージが拒否された .....	175
パッケージが期待どおりに更新されない .....	176
同期リンクされたパッケージをインストールできない .....	177
子イメージで一時的な起点を使用できない .....	178
非大域ゾーンをインストールできない .....	179
イメージを修正できない .....	180
ファイルが回収された .....	180
格納されるイメージメタデータの最小化 .....	181



パッケージのインストールパフォーマンスの増大 .....	182
索引 .....	183



## 例目次

---

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 .....	33
例 2	アクションタイプの指定 .....	34
例 3	アクション属性の指定 .....	34
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 .....	35
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 .....	35
例 6	パッケージインストールの取り消し .....	68
例 7	アンインストールする複数のパッケージの指定 .....	69
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 .....	69
例 9	group 依存関係であるパッケージのアンインストール .....	70
例 10	SPARC T7 ファームウェア更新についての情報の表示 .....	101
例 11	IDR パッケージアーカイブに関する情報を取得する .....	103
例 12	この IDR で提供されている内容を表示する .....	103
例 13	IDR によって対処される問題を表示する .....	104
例 14	IDR をインストールできる Oracle Solaris リリースを判定する .....	104
例 15	IDR のリリースノートの場所を表示する .....	104
例 16	IDR の名前が変更されているかどうかを表示する .....	105
例 17	置換用 IDR をインストールする .....	108
例 18	IDR の問題を修正する SRU に更新する .....	109
例 19	IDR の問題を修正しない SRU に更新する .....	110
例 20	新しいパブリッシャーの指定 .....	116
例 21	パブリッシャー構成のインポート .....	116
例 22	パブリッシャーの鍵と証明書の指定 .....	119
例 23	パブリッシャーの鍵と証明書の取り消し .....	120
例 24	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 .....	139
例 25	イメージ変更に関する情報の取得 .....	151
例 26	履歴情報の削除 .....	152
例 27	Java Runtime Environment のロック解除と更新 .....	168

例 28	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 .....	172
------	--	-----

## このドキュメントの使用方法

---

- **概要** – Oracle Solaris 11.4 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=E75431-01> 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。



## 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
- ソフトウェ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 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、[16 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クネスなどの、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。『[Oracle Solaris 12 パッケージリポジトリのコピーと作成](#)』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。『[Oracle Solaris 12 でのImage Packaging System を使用したソフトウェアのパッケージ化と配布](#)』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、『[Oracle Solaris 12 システムの自動インストール](#)』または『[Oracle Solaris 12 システムの手動インストール](#)』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、『[Oracle Solaris 12 でのImage Packaging System を使用したソフトウェアのパッケージ化と配布](#)』の「[Package Content: Actions](#)」または [pkg\(7\)](#) 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、[130 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、[138 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、[36 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のパッケージ名およびバージョンの照合に対して次のルールが適用されます。

- パッケージ名。パッケージを一意に特定するパッケージ名のもっとも小さい部分を使用できます。glob(3C) 形式のワイルドカードとして 1 つ以上のパッケージ名と一致する \* および \* 文字を使用できます。
- パッケージバージョン。バージョンを指定した場合、バージョン文字列の右辺からコンポーネントを省略できます。文字 \* をワイルドカードとして使用して、パー

ジョン文字列のコンポーネント全体を照合できます。文字 \* はコンポーネントの一部の照合に使用できません。単一の \* 文字は複数のコンポーネントを照合できません。? をバージョン文字列内で使用できません。

次の例でパッケージの一致について説明します。

- 名前のどこかに `jre` を含むインストール済みパッケージを照合します。

```
$ pkg list '*jre*'
NAME (PUBLISHER)                VERSION                IFO
runtime/java/jre-8              1.8.0.181.12         i--
```

- 名前のどこかに `java` を含み、あらゆるコンポーネントバージョンで、ブランチバージョンが `11.4` のすべてのインストール済みパッケージを照合します。

```
$ pkg list '*java*@*11.4*'
NAME (PUBLISHER)                VERSION                IFO
library/javascript/jjv         1.0.2-11.4.0.0.1.10.0 i--
system/management/rad/client/rad-java 11.4-11.4.0.0.1.10.1 i--
```

- 次のコマンドと前のコマンドの 1 文字の違いに注意してください。

```
$ pkg list '*java*@*11.4'
pkg list: Illegal FMRI '*java*@*11.4': Bad Version: *11.4
```

- アスタリスク \* はバージョンコンポーネントの一部や複数のバージョンコンポーネントを照合できません。

```
$ pkg list '*java*@11.4-11.4.*.10.0'
pkg list: no packages matching the following patterns are installed:
  *java*@11.4-11.4.*.10.0
$ pkg list '*java*@*.10.0'
pkg list: no packages matching the following patterns are installed:
  *java*@*.10.0
```

すべての IPS パッケージ FMRI のスキームは `pkg` です。次に示す compliance セキュリティコンプライアンスフレームワークのパッケージ FMRI の例で、`solaris` はパブリッシャー、`security/compliance` はパッケージ名、`11.4-11.4.0.0.1.10.1:20180702T144054Z` はバージョンです。

```
pkg://solaris/security/compliance@11.4-11.4.0.0.1.10.1:20180702T144054Z
```

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

**security/compliance**

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 3 つの部分で構成され、タイムスタンプの形式は **dateTZ** です。

```
component_version-branch_version:time_stamp
```

コンポーネントバージョン番号: 11.4

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番号は **minor.update** です。FOSS コンポーネントなどのほかのコンポーネントは独自のバージョン番号を持っています。次の例で、**web/server/apache-24** パッケージのバージョン番号は 2.4.25 です。

```
pkg://solaris/web/server/apache-24@2.4.33-11.4.0.0.1.10.0:20180702T172601Z
```

コンポーネントバージョン番号は、**pkg info** 出力の「Version」と同じです。

ブランチバージョン: 11.4.0.0.1.10.1

ブランチバージョン (存在する場合) は、ハイフン (-) のあとに続ける必要があります。ブランチバージョン文字列は、**pkg info** の出力の「Branch」と同じです。

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。

```
minor.update.sru.order.platform.build.rev
```

マイナーリリース番号: 11

**uname -r** コマンドで出力される **major.minor** の **minor** 部分。

更新リリース番号: 4

この Oracle Solaris リリースの更新リリース番号。

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。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。

順序: 0

この値は内部で使用されます。

プラットフォーム: 1

この値は内部で使用されます。

リリースまたは SRU ビルド番号: 10

SRU のビルド番号、またはリリースの更新番号。

リビジョンまたはナイトリービルド番号: 1

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) 更新の場合、パッケージ FMRI の `component_version-branch_version` は単なる単一フィールドです。たとえば、idr1929 の FMRI は `pkg://solaris/idr1929@4:20160216T222617Z` で、バージョンの完全な `component_version-branch_version` 部分は 4 にすぎません。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、[102 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。

タイムスタンプ: 20180702T144054Z

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 (YYYYMMDDTHHMMSSZ) 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(7) 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り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、[141 ページの「ブート環境ポリシーイメージのプロパティ」](#)で説明するように、イメージポリシーに依存します。ま

た、51 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、『Oracle Solaris 12 ブート環境の作成と管理』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。

72 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、『Oracle Solaris ゾーンの紹介』を参照してください。

## パッケージのファセットとバリエーション

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エーションと呼びます。

ファセットとバリエーションは、イメージに設定される特殊プロパティです。ファセットとバリエーションは、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エーション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エーションの値の比較により、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エーションの値を表示または変更する方法など、ファセットとバリエーションの詳細については、123 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、『[Oracle Solaris 12 でのユーザーとプロセスのセキュリティー保護](#)』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。



## 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。1 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 '*java*8*'
NAME (PUBLISHER)                VERSION                IFO
consolidation/java-8/java-8-incorporation  1.8.0.172.11-0       i--
runtime/java/jre-8              1.8.0.172.11         i--
```

パブリッシャー名がパッケージ名のあとで丸括弧で囲まれて表示されている場合、そのパブリッシャーは、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。この例で示されている両方のパッケージは、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。[118 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 '*java*8*'
NAME (PUBLISHER)                VERSION                IFO
consolidation/java-8/java-8-incorporation  1.8.0.172.11-0       i--
developer/java/jdk-8            1.8.0.172.11         ---
library/java/java-demo-8       1.8.0.172.11         ---
runtime/java/jre-8              1.8.0.172.11         i--
web/java-servlet/tomcat-8       8.5.28-11.4.0.0.1.10.0 ---
web/java-servlet/tomcat-8/tomcat-admin      8.5.28-11.4.0.0.1.10.0 ---
web/java-servlet/tomcat-8/tomcat-examples  8.5.28-11.4.0.0.1.10.0 ---
```

この出力は、開発者キット、デモライブラリ、および `tomcat` パッケージが利用可能でこのイメージ内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 '*java*8*'
NAME (PUBLISHER)                VERSION                IFO
consolidation/java-8/java-8-incorporation  1.8.0.181.12-0       ---
developer/java/jdk-8            1.8.0.181.12         ---
```

library/java/java-demo-8	1.8.0.181.12	---
runtime/java/jre-8	1.8.0.181.12	---
web/java-servlet/tomcat-8	8.5.28-11.4.0.0.1.10.0	---
web/java-servlet/tomcat-8/tomcat-admin	8.5.28-11.4.0.0.1.10.0	---
web/java-servlet/tomcat-8/tomcat-examples	8.5.28-11.4.0.0.1.10.0	---

tomcat パッケージは `-a` オプションで一覧表示されたパッケージと同じバージョンです。これらの tomcat パッケージはインストールできます。このリストのほかのパッケージは、`-a` オプションで一覧表示された同じパッケージより新しいバージョンです。これらのパッケージは `-a` オプションによって一覧表示されていないため、これらのパッケージをこのイメージにインストールできないことがわかります。これらのパッケージをインストールできないのは、インストールできるバージョンが、[16 ページの「制約パッケージ」](#)に示すように制約されているためです。これらのパッケージをシステムで取得するには、システムに `pkg update` を実行する必要があります。これによって新しいイメージが作成されます。

最新バージョンのパッケージが、廃止されたパッケージ (O 列に「o」が表示されている) または名前が変更されたパッケージ (O 列に「r」が表示されている) である場合があります。詳細は、[28 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。

## 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。[26 ページの「最新のパッケージ」](#)に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。以下に示すインストール済みパッケージに使用可能な更新がありますが、これらの更新を現在のイメージにインストールすることはできません。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 '*java*8*'
NAME (PUBLISHER)          VERSION          INFO
consolidation/java-8/java-8-incorporation  1.8.0.172.11-0  i--
untime/java/jre-8        1.8.0.172.11    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があります。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 jre-8
NAME (PUBLISHER)          VERSION          IFO
runtime/java/jre-8       1.8.0.181.12   ---
runtime/java/jre-8       1.8.0.172.11   i--
runtime/java/jre-8       1.8.0.162.12   ---
...
```

## パッケージの名前変更と廃止

次の例で、`O` 列の「`o`」はパッケージが廃止されたことを示し、`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ます。

```
$ pkg list -n jdk storage/storage-nas
NAME (PUBLISHER)          VERSION          IFO
developer/java/jdk       1.7.0.999.99   --o
storage/storage-nas      0.1-0.173.0.0.0.1.0  --r
```

廃止されたパッケージも名前が変更されたパッケージもインストールできません。

廃止されたパッケージをインストールしようとする、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。

```
$ pkg install developer/java/jdk
No updates necessary for this image.
```

パッケージの名前が変更されていないとしても、インストールが可能な類似パッケージを見つけることができる場合があります。前の例では、`jdk-8` はインストール可能であることがわかります。

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、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 storage/storage-nas
Name: storage/storage-nas
State: Not installed (Renamed)
Renamed to: consolidation/osnet/osnet-incorporation
           pkg:/group/feature/storage-nas@0.5.11,5.11-0.173.0.0.0.0.0
Publisher: solaris
Version: 0.1
Branch: 0.173.0.0.0.1.0
Packaging Date: Fri Aug 26 16:26:38 2011
Size: 5.45 kB
FMRI: pkg://solaris/storage/storage-nas@0.1-0.173.0.0.0.1.0:20110826T162638Z
```

storage/storage-nas パッケージをインストールしようとする、group/feature/storage-nas 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)。

「Renamed to」に示すパッケージがすでにインストールされている場合、必要な更新が存在しないという報告がパッケージシステムから出されます。

## 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、[129 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 -v entire
FMRI                                                    IFO
pkg://solaris/entire@11.4-11.4.0.0.1.10.0:20180702T173343Z  i--
$ pkg freeze -c "Prevent update to SRU 1 until ready." entire@11.4-11.4.0
entire was frozen at 11.4-11.4.0
$ pkg list -v entire
FMRI                                                    IFO
pkg://solaris/entire@11.4-11.4.0.0.1.10.0:20180702T173343Z  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、パッケージが凍結されているかどうか、最後にインストールまたは更新された時期、および

オープンソースソフトウェアのソース位置を表示します。pkg info コマンドはライセンス情報を個別に表示できます。

## パッケージの説明、パッケージ化およびインストールされた時期、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@11.4-11.4.0.0.1.10.0:20180702T173343Z  i--
```

pkg info コマンドは、パッケージがパッケージ化された時期やパッケージのサイズなどのさらに詳しい情報を表示できます。この例で、pkg info の出力の「State」は、このパッケージが凍結されていることを示しています。

```
$ pkg info entire
Name: entire
Summary: Incorporation to lock all system packages to the same build
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.
Category: Meta Packages/Incorporations
State: Installed (Frozen)
Publisher: solaris
Version: 11.4 (Oracle Solaris 11.4.0.0.1.10.0)
Branch: 11.4.0.0.1.10.0
Packaging Date: Mon Jul 02 17:33:43 2018
Size: 2.53 kB
FMRI: pkg://solaris/entire@11.4-11.4.0.0.1.10.0:20180702T173343Z
```

情報が得られる場合、pkg info コマンドはパッケージが前回インストールまたは更新された時期を表示します。前回インストールまたは更新された時期が pkg info で表示されない場合、パッケージはこの BE が最初に作成されたときにインストールされた可能性があります。その場合、beadm list コマンドによって表示される BE 作成日を使用してください。また、[149 ページの「操作履歴の表示」](#)に記載されている pkg history コマンドを使用して、すべて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時期を見つけることもできます。

次の出力は、パッケージが最後にインストールされた時期を示しています。

```
$ pkg info mercurial-27
Name: developer/versioning/mercurial-27
Summary: The Mercurial Source Control Management System
Description: A fast, lightweight source control management system designed
              for efficient handling of very large distributed projects.
Category: Development/Source Code Management
```

```

State: Installed
Publisher: solaris
Version: 4.1.3
Branch: 11.4.0.0.1.10.0
Packaging Date: Mon Jul 02 16:32:13 2018
Last Install Time: Mon Aug 06 15:22:47 2018
Size: 9.90 MB
FMRI: pkg://solaris/developer/versioning/mercurial-27@4.1.3-11.4.0.0.1.10.0:
20180702T163213Z
Project URL: http://mercurial-scm.org/
Source URL: https://www.mercurial-scm.org/release/mercurial-4.1.3.tar.gz

```

28 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。`pkg info` 出力の「State」行も、パッケージが凍結されているかどうかを示しています。

## オープンソースのためのプロジェクトおよびソース位置の表示

FOSS パッケージの場合、プロジェクト URL およびソース URL が表示されます。

```

$ pkg info openssl
Name: library/security/openssl
Summary: OpenSSL - a Toolkit for Secure Sockets Layer (SSL v2/v3) and
Transport Layer (TLS v1) protocols and general purpose
cryptographic library
Description: OpenSSL is a full-featured toolkit implementing the Secure
Sockets Layer (SSL v2/v3) and Transport Layer Security (TLS v1)
protocols as well as a full-strength general purpose
cryptography library.
Category: System/Security
State: Installed
Publisher: solaris
Version: 1.0.2.15 (1.0.2o)
Branch: 11.4.0.0.1.10.0
Packaging Date: Mon Jul 02 17:04:44 2018
Size: 16.05 MB
FMRI: pkg://solaris/library/security/openssl@1.0.2.15-11.4.0.0.1.10.0:
20180702T170444Z
Project URL: https://www.openssl.org/
Source URL: https://www.openssl.org/source/openssl-1.0.2o.tar.gz

```

## 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。インストールされていない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 install-image/solaris-auto-install
Copyright (c) 1983, 2018, Oracle and/or its affiliates. All rights reserved.

```

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、[37 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。[38 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せん。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。バリエーションとファセットについては、[123 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(7)` 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。`-t` オプションを使用した場合、デフォルト出力はそのアクションのキー属性の値です。

`-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(7)` 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、[32 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-18c-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-18c-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-18c-preinstall
$ pkg contents -ro action.name oracle-rdbms-server-18c-preinstall
$ pkg contents -ro action.raw oracle-rdbms-server-18c-preinstall
```

### 例 2 アクションタイプの指定

次のコマンドは、oracle-rdbms-server-18c-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 oracle-rdbms-server-18c-preinstall
TYPE  FMRI
group  system/header
group  system/kernel/oracka
group  system/picl
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
require system/dtrace
require system/library/openmp
```

### 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*' oracle-rdbms-server-18c-preinstall
TYPE  FMRI
group  x11/library/libxi
group  x11/library/libxtst
require system/library/openmp
```

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-18c-preinstall
NAME          VALUE
pkg.summary   Prerequisite package for Oracle Database 18c
pkg.description Provides the set of Oracle Solaris packages required for installation and
               operation of Oracle Database 18c.
```

`set` アクションタイプを指定した場合、`name` および `value` は `set` アクションのデフォルト出力であるため、出力列を指定する必要はありません。

```
$ pkg contents -rt set -a name=pkg.summary -a name=pkg.description -s value oracle-rdbms-server-18c-preinstall
```

#### 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-35` 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、[133 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-35
PATH          TARGET
usr/bin/2to3  2to3-3.5
usr/bin/idle  idle3.5
usr/bin/pydoc pydoc3.5
usr/bin/python python3.5
usr/bin/python-config python3.5-config
usr/bin/pyenv pyenv-3.5
usr/lib/pkgconfig/python3.pc python-3.5.pc
usr/lib/sparcv9/pkgconfig/python3.pc python-3.5.pc
usr/share/man/man1/python3.1 python3.5.1
```

#### 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(7)` のマニュアルページに示す属性のほかに、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 2.7.14、Python 3.4.6、および Python 3.5.3 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、[133 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 python2.7 runtime/python-27
usr/bin/python python3.4 runtime/python-34
```

usr/bin/python python3.5 runtime/python-35

## 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。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せん。

---

```
$ pkg contents -rt depend solaris-minimal-server
TYPE      FMRI
group     network/ping
group     network/ssh
group     service/network/ssh
group     shell/tcsh
group     shell/zsh
group     system/network
group     system/rsyslog
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。`group` タイプの依存関係を持つパッケージのインストールを回避できます。[138 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。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、[17 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
network/ssh                   OpenSSH client and associated utilities
package/pkg                   Image Packaging System
release/name                  Solaris Naming Enabler
release/notices               Oracle Solaris notices
service/network/ssh           OpenSSH servers and SSH (Secure Shell) services
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
system/rsyslog                 reliable and extended syslogd
```

マニュアルページは、その対象となるソフトウェアがインストールされるときにデフォルトでインストールされます。ただし、`solaris-minimal-server` インストールグループは `man` コマンドをインストールしません。`solaris-minimal-server` グループをインストールする場合、次のいずれかを行うことが必要な場合があります。

- `doc.man` ファセットを `false` に設定して、マニュアルページのインストールを回避します。[126 ページの「ファセット値の表示と変更」](#) を参照してください。
- `text/doctools` パッケージをインストールして `man` コマンドをインストールします。

## 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
lic_OTN install-image/solaris-auto-install
lic_OTN release/notices
```

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、[31 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 install-image/solaris-auto-install
```

## 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。

表 1 pkg search と pkg contents の比較

pkg search	pkg contents
<ul style="list-style-type: none"> <li>■ 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、<code>-l</code> オプションを使用します。</li> <li>■ 検索するリポジトリの URI を指定する場合は、<code>-s</code> オプションを使用します。</li> <li>■ 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。</li> <li>■ 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。</li> <li>■ 結果の列を指定する場合は、<code>-o</code> オプションを使用します。</li> </ul>	<ul style="list-style-type: none"> <li>■ 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、<code>-r</code> オプションを使用します。</li> <li>■ 調べるリポジトリの URI を指定する場合は、<code>-g</code> オプションを使用します。</li> <li>■ アクションを指定する場合は、<code>-t</code> オプションを使用します。</li> <li>■ 属性および属性値を指定する場合は、<code>-a</code> オプションを使用します。</li> <li>■ 結果の列を指定する場合は、<code>-o</code> オプションを使用します。</li> <li>■ 結果をソートする場合は、<code>-s</code> オプションを使用します。</li> </ul>

**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
```

<i>package</i>	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。
<i>action</i>	<code>pkg(7)</code> 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。
<i>index</i>	通常は、 <code>pkg(7)</code> 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 <i>action</i> の属性を選択します。詳細は、次を参照してください。
<i>token</i>	<i>index</i> の値または <i>index</i> 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。

*index* はアクション属性の名前と必ずしも同じではありません。たとえば、`user` アクションの *index* は `name` ですが、その属性の名前は `username` です。

すべてのアクション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。

次の例は、アクションについて考えられるすべてのインデックスを見つける 1 つの方法を示したものです。

```
$ pkg search :user::
```

このコマンドは、`user` アクションの検索可能な属性が `name` のみであることを示します。

*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。

*index* は、*action.key* (アクションのキー属性) や *pkg.name* などの特殊な疑似属性名とすることもできます。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。

*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 *pci14e\** を指定できます。

```
driver alias=pci14e4,1000 alias=pci14e4,1200 alias=pci14e4,1203
```

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。

```
$ pkg search -Hpo pkg.name driver:alias:pci14e\*
driver/network/ethernet/bge
driver/network/ethernet/bnx
driver/network/ethernet/bnxe
```

*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="QLogic 57xxx 10/20GbE NIC Driver"
```

次の例では、パッケージ名を *\*driver\**、*index* を *pkg.summary*、*token* を *qlogic* に指定しています。*pkg.summary* 属性は *set* アクションにのみ適用されるため、*set* アクションが暗黙指定されます。*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 \*driver\*::pkg.summary:qlogic
SEARCH.MATCH                                PKG.NAME
QLogic 570x/571x Gigabit Ethernet Driver    driver/network/ethernet/bnx
QLogic 57xxx 10/20GbE NIC Driver            driver/network/ethernet/bnxe
QLogic ISP Fibre Channel device storage and NIC driver driver/fc/qlc
QLogic P3+ 10GbE NIC Driver                 driver/network/ethernet/qlcnic
```

次のコマンドは前のコマンドと同じ結果をもたらします。

```
$ pkg search -o search.match, pkg.name pkg.summary: 'qlogic * driver'
```

*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(7)* 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。

## 指定されたファイルを提供するパッケージの識別

次の例では、検索トークンはファイルシステムオブジェクトのフルパスです。

```
$ pkg search /lib/libinetutil.so.1
INDEX      ACTION VALUE      PACKAGE
path      file   lib/libinetutil.so.1  pkg://system/library@11.4-11.4.0.0.1.10.1
```

`pkg search` コマンドはパッケージアクションを検索します。前のコマンドは、`file` アクションの `path` 属性内に検索トークンを検出し、このファイルが `system/library` パッケージによって提供されたことを示します。

次の例は、必要な出力はパッケージの名前のみであることを指定しています。`-l` オプションを指定すると、インストール済みのパッケージのみが検索されるため、検索を速くすることができます。

```
$ pkg search -Hlo pkg.name '*libinetutil*'
system/library
```

必要な出力を指定しない場合、ファイルの基本名のみを検索トークンとして使用するか、上記のようにワイルドカードを使用することで、このファイルが6つの異なるディレクトリに提供され、3つのリンクのターゲットであることが表示されます。1つのパッケージについて多くの出力行を表示する代わりに各パッケージ名を1回だけ表示するための別の方法は、`-p` オプションを使用するか、検索トークンを角括弧で囲むことです。使用するシェルによっては角括弧をエスケープする必要があります。次のコマンドの出力は同じになります。

```
$ pkg search -p '*libinetutil*'
$ pkg search \<<'*libinetutil*'>>
PACKAGE                                     PUBLISHER
pkg://system/library@11.4-11.4.0.0.1.10.1  solaris
```

`pkg verify` コマンドの `-p` オプションにより、名前付きファイルを提供するパッケージも表示されます。この場合、`/` を基準としたファイルへのフルパスを指定する必要があります。

```
$ pkg verify -vp lib/libinetutil.so.1
PACKAGE                                     STATUS
pkg://solaris/system/library                OK
```

システム上でファイルを提供したパッケージを見つけることができない場合、そのファイルはパッケージ化されていない可能性があります。[65 ページの「パッケージ解除されたファイルシステムの内容の識別」](#) の `pkg verify` コマンドの `-p` および `--unpackaged` オプションを参照してください。

## 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/proxy/privoxy  svc:/network/http
web/proxy/squid    svc:/network/http
web/server/lighttpd-14  svc:/network/http
web/java-servlet/tomcat-8  svc:/network/http
web/server/apache-24  svc:/network/http
web/server/apache-24  svc:/network/http:apache24
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-8  svc:/network/http:tomcat8
```

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:apache24 value=svc:/network/http
```

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-24  svc:/network/http:apache24
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-8  svc:/network/http:tomcat8
```

## 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::openldap
ACTION.KEY PKG.NAME
```

```
openldap    system/network/ldap/openldap
```

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::openldap OR user::sshd
ACTION.KEY PKG.NAME
openldap   system/network/ldap/openldap
sshd       service/network/ssh
```

## 指定された修正を提供するパッケージの識別

次のコマンドは、特定のバグ修正を提供するパッケージを示しています。

```
$ pkg search -Ho pkg.shortfmri bugid
```

バグを修正するために変更されたすべてのパッケージが一覧表示されます。

特定の CVE 修正を提供した SRU を識別する方法について学習するには、[99 ページの「クリティカルパッチ更新パッケージ」](#)を参照してください。

## 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.name info.classification:'source code management'
pkg:/developer/versioning/sccs
pkg:/developer/versioning/git
pkg:/developer/versioning/mercurial-27
```

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
Name: developer/versioning/mercurial-27
Summary: The Mercurial Source Control Management System
Description: A fast, lightweight source control management system designed
             for efficient handling of very large distributed projects.
Category: Development/Source Code Management
State: Installed
Publisher: solaris
Version: 4.1.3
Branch: 11.4.0.0.1.10.0
Packaging Date: Mon Jul 02 16:32:13 2018
Last Install Time: Mon Aug 06 15:22:47 2018
Size: 9.90 MB
FMRI: pkg://solaris/developer/versioning/mercurial-27@4.1.3-11.4.0.0.1.10.0:
20180702T163213Z
Project URL: http://mercurial-scm.org/
```

Source URL: <https://www.mercurial-scm.org/release/mercurial-4.1.3.tar.gz>

検索可能なほかの分類については、『[Oracle Solaris 12 でのImage Packaging System を使用したソフトウェアのパッケージ化と配布](#)』の「Classification Values」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 -a '**versioning*'
$ pkg list '**storage*'
$ pkg list -n '**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。

次のコマンドは、system/network/ldap/openldap パッケージについて多くのパッケージが require 依存関係を持っていることを示します。

```
$ pkg search -Hlo pkg.name require:system/network/ldap/openldap

```

次の pkg contents コマンドは、検索結果の1つについて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 -t depend -a fmri='*openldap*' -o pkg.name,action.raw openscap
PKG.NAME          ACTION.RAW
security/compliance/openscap depend fmri=pkg:/system/network/ldap/openldap@2.4.45-11.4.0.0.1.9.0 type=require

```

次の例は、pkg:/driver/graphics/nvidiaR340 について exclude 依存関係を持っているパッケージを一覧表示します。

```
$ pkg contents -rt depend -a type=exclude nvidiaR340
TYPE    FMRI
exclude pkg:/driver/graphics/nvidia
exclude pkg:/driver/graphics/nvidiaR304
$ pkg search -o pkg.name,fmri 'depend:exclude:*nvidiaR340*'
PKG.NAME          FMRI
driver/graphics/nvidia    pkg:/driver/graphics/nvidiaR340
driver/graphics/nvidiaR304 pkg:/driver/graphics/nvidiaR340

```

## 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

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 Manifest 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せん。

---

```
$ 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
group   network/ssh
require package/pkg
require release/name
require release/notices
group   service/network/ssh
require shell/bash
require shell/ksh93
group   shell/tcsh
group   shell/zsh
require system/core-os
require system/library/platform
group   system/network
group   system/rsyslog
```

`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。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、17 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、36 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エーション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

## 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

[72 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、[24 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、このオプションによりイメージに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。

---

## パッケージインストールのプレビュー

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 oracle-rdbms-server-18c-preinstall
      Packages to install:      4
      Services to change:      1
      Estimated space available: 852.48 GB
      Estimated space to be consumed: 456.23 MB
      Create boot environment:  No
      Create backup boot environment:  No
      Rebuild boot archive:     No

Changed packages:
solaris
  group/prerequisite/oracle/oracle-rdbms-server-18c-preinstall
    None -> 11.4-11.4.0.0.1.10.0
  system/kernel/oracka
    None -> 11.4-11.4.0.0.1.10.1
  x11/diagnostic/x11-info-clients
    None -> 7.7-11.4.0.0.1.10.0
  x11/library/libdmx
    None -> 1.1.3-11.4.0.0.1.10.0

Services:
refresh_fmri:
  svc:/network/socket-config: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を指定することもできます。

「Changed packages」セクションには、コマンド行に一覧表示されたグループパッケージがインストールされ、そのほかの3つのパッケージ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。



oracle-rdbms-server-18c-preinstall パッケージには 3 つを超える依存関係があります。

```
$ pkg contents -rt depend oracle-rdbms-server-18c-preinstall
TYPE      FMRI
group     system/header
group     system/kernel/oracka
group     system/picl
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
require   system/dtrace
require   system/library/openmp
```

次のコマンドは、12 個のうち 10 個の依存関係がこのイメージにすでにインストールされていることを示しています。

```
$ pkg list `pkg contents -Hro fmri -t depend oracle-rdbms-server-18c-preinstall`
NAME (PUBLISHER)                                VERSION                                         IFO
compress/unzip                                  6.0.3-23-11.4.0.0.1.10.0                     i--
developer/assembler                             11.4-11.4.0.0.1.4.0                          i--
developer/build/make                            11.4-11.4.0.0.1.4.0                          i--
system/dtrace                                   11.4-11.4.0.0.1.10.1                        i--
system/header                                   11.4-11.4.0.0.1.10.1                        i--
system/library/openmp                           11.4-11.4.0.0.1.4.0                          i--
system/picl                                     11.4-11.4.0.0.1.10.1                        i--
x11/library/libxi                               1.7.9-11.4.0.0.1.10.0                       i--
x11/library/libxtst                             1.2.3-11.4.0.0.1.10.0                       i--
x11/session/xauth                               1.0.10-11.4.0.0.1.10.0                      i--
```

```
pkg list: no packages matching the following patterns are installed:
  system/kernel/oracka
  x11/diagnostic/x11-info-clients
```

インストールする必要がある oracle-rdbms-server-18c-preinstall の 2 つの直接の依存関係に加えて、x11/library/libdmx パッケージもインストールされるのは、次のコマンドが示すように、これが x11-info-clients パッケージの必須依存関係であるためです。

```
$ pkg search -o type, pkg.name :depend:group:x11/library/libdmx OR :depend:require:x11/library/libdmx
TYPE      PKG.NAME
require   x11/diagnostic/x11-info-clients
require   x11/server/xdmx
require   developer/opensolaris/userland
```

## ファセットの変更のプレビュー

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。

表 2 `pkg install` と `pkg update` の比較

<code>pkg install</code>	<code>pkg update</code>
<ul style="list-style-type: none"> <li>■ オペランドとして 1 つ以上のパッケージ名が必要です。</li> <li>■ 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。</li> <li>■ 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。</li> <li>■ 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。</li> </ul>	<ul style="list-style-type: none"> <li>■ 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。</li> <li>■ インストール済みパッケージを更新します。</li> <li>■ 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。</li> <li>■ 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。</li> <li>■ 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。</li> </ul>

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(7)` 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。62 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。

ファイルを提供するパッケージがインストールまたは更新されるときに特定の属性を持つファイルアクションがどのように処理されるかについては、『[Oracle Solaris 12](#) で

の [Image Packaging System](#) を使用したソフトウェアのパッケージ化と配布』の「[File Actions](#)」または [pkg\(7\)](#) のマニュアルページ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。メディアータの設定または設定解除、バリエーション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、[141 ページの「ブート環境ポリシーイメージのプロパティ」](#) 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。

---

**注記** - 操作中の BE が現在アクティブな BE ではない場合、新規またはバックアップ BE を作成するオプションは無視されます。操作中の BE が現在アクティブな BE ではない場合、変更は直接そのイメージに対して行われます。[pkg\(1\)](#) のマニュアルページの `-R` オプションおよび `PKG_IMAGE` 環境変数の説明を参照してください。

---

#### `--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\(8\)](#) のマニュアルページに記載されている `beadm` 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、[141 ページの「ブート環境ポリシーイメージのプロパティ」](#)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、[141 ページの「ブート環境ポリシーイメージのプロパティ」](#)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

[72 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となります。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、72 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、いずれか短い方が完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、[31 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、[37 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、[138 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、[18 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ッキネスと検索順序の設定については、[113 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、[25 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ています。

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

115 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イタス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、[141 ページの「ブート環境ポリシーイメージのプロパティ」](#) 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。

[48 ページの「パッケージインストールのプレビュー」](#) の例では、`oracle-rdbms-server-18c-preinstall` 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 --be-name rdbms oracle-rdbms-server-18c-preinstall
      Packages to install:  4
      Create boot environment: Yes
      Create backup boot environment: No
```

新しい BE は現在の BE のコピーとして開始したため、インストールするパッケージの数はまだ 4 つのみです。

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 11.4.0 exists and has been updated and activated.
On the next boot the Boot Environment be://rpool/rdbms 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 `oracle-rdbms-server-18c-preinstall` パッケージがインストールされていないため、`oracle-rdbms-server-18c-preinstall`

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。oracle-rdbms-server-18c-preinstall パッケージが新しい rdbms BE にインストールされます。

```
$ pkg list oracle-rdbms-server-18c-preinstall
pkg list: no packages matching the following patterns are installed:
  oracle-rdbms-server-18c-preinstall
```

システムに rdbms という名前の新しいアクティブな BE があることを確認するには、beadm list コマンドを使用します。「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 Name      Flags Mountpoint Space  Policy Created
-----
11.4.0      N      /           221.09M static 2018-07-24 15:18
rdbms       R      -           18.93G  static 2018-08-06 12:37
```

新しい BE に oracle-rdbms-server-18c-preinstall 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、oracle-rdbms-server-18c-preinstall 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 rdbms /mnt
$ beadm list
BE Name      Flags Mountpoint Space  Policy Created
-----
11.4.0      N      /           221.09M static 2018-07-24 15:18
rdbms       R      /mnt        18.93G  static 2018-08-06 12:37
$ pkg -R /mnt list oracle-rdbms-server-18c-preinstall
NAME (PUBLISHER)                                VERSION                                IFO
group/prerequisite/oracle/oracle-rdbms-server-18c-preinstall 11.4-11.4.0.0.1.10.0 i--
```

rdbms BE を必ずアンマウントしてください。

```
$ beadm unmount rdbms
```

現在の BE での作業を続行してイメージ構成を変更する場合、現在の BE はアクティブ化された BE ではないため次のようなリマインダが表示されます。

```
WARNING: The boot environment being modified is not the active one. Changes
made in the active BE will not be reflected on the next boot.
```

## 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、[138 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 `cvcs` 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/cvcs 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、[18 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、[113 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(7)` 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、[55 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。

どのパッケージのどのバージョンがインストールされ、どのファイルが保持されるかを確認するには、[48 ページの「パッケージインストールのプレビュー」](#)に記載されているように操作をプレビューしてください。

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、[18 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(7)` 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、[55 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。

依存関係によって、パッケージをダウングレードできない場合もあります。

## 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。

表 3 pkg fix と pkg revert の比較

pkg fix	pkg revert
<ul style="list-style-type: none"> <li>■ 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。</li> <li>■ 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。</li> <li>■ 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。</li> </ul>	<ul style="list-style-type: none"> <li>■ 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。</li> <li>■ 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。</li> </ul>

## 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、[143 ページの「署名付きパッケージのイメージプロパティ」](#)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。51 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
```

pkg verify の出力をプログラムで評価するには、`--parsable 0` オプションを指定します。`--parsable` オプションを使用する場合は、`-v` オプションを使用しないでください。

## ファイルシステムの内容の検証

pkg verify コマンドは、インストール済みパッケージの検証に加えて、インストール済みのディレクトリ、ファイル、およびリンクを検証できます。

次の例では `-p` オプションを使用して、前のセクションでインストール済みのすべてのパッケージの完全な検証を実行して表示したものと同じ情報を表示します。

```
$ pkg verify -p var/cache/pkg/depot
PACKAGE                                     STATUS
pkg://nightly/package/pkg/depot             ERROR
  dir: var/cache/pkg/depot
        ERROR: Group: 'pkg5srv (97)' should be 'bin (2)'
```

次の例では、ファイル検証ステータスが OK であるため、`-v` オプションを追加しないかぎり出力は表示されません。

```
$ pkg verify -vp etc/sudoers
PACKAGE                                     STATUS
pkg://nightly/security/sudo                 OK
  file: etc/sudoers
        editable file has been changed
```



## パッケージ解除されたファイルシステムの内容の識別

次の例のメッセージは、ファイルがパッケージから提供されていないことを示しています。

```
$ pkg verify -p etc/resolv.conf
PACKAGE                               STATUS
etc/resolv.conf is not found in the image
```

ファイル `/etc/resolv.conf` はシステム上に存在しますが、パッケージ化されていません。ファイルには次のコメントが含まれており、ファイルは SMF データから生成されたことを示しています。

```
# _AUTOGENERATED_FROM_SMF_V1_
#
# WARNING: THIS FILE GENERATED FROM SMF DATA.
# DO NOT EDIT THIS FILE. EDITS WILL BE LOST.
# See resolv.conf(5) for details.
```

パッケージから提供されたものでないファイルシステムの内容をインストール済みパッケージ検証のレポートに含めるには、`--unpackaged` オプションを使用します。

```
$ pkg verify -v --unpackaged
```

パッケージから提供されたものでないファイルシステムの内容のみレポートするには、`--unpackaged-only` オプションを使用します。

```
$ pkg verify --unpackaged-only
```

## 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、『[Oracle Solaris 12 でのImage Packaging System を使用したソフトウェアのパッケージ化と配布](#)』の「File Actions」および『[Oracle Solaris 12 でのImage Packaging System を使用したソフトウェアのパッケージ化と配布](#)』の「Directory Actions」の `revert-tag` 属性の説明を参照してください。

次の例は、タグ名 `system:sysconfig-profile` でタグ付けされたディレクトリを表示します。sysconfig(8) のマニュアルページに記載されているように、`sysconfig unconfigure` コマンドに `--remove-profiles` オプションを指定して使用すると、パッケージ解除されたファイルはこれらのディレクトリから削除されます。

```

$ pkg contents -H -a revert-tag='system:sysconfig-profile*' '*'
etc/svc/profile/enterprise
etc/svc/profile/incoming
etc/svc/profile/node
etc/svc/profile/site
etc/svc/profile/sysconfig
etc/svc/profile/system

```

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

パッケージ化された状態に戻されます。詳細は、[archiveadm\(8\)](#) のマニュアルページを参照してください。

```
$ pkg contents -Ha 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せん。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: 852.20 GB
Estimated space to be consumed: 470.42 MB
      Create boot environment:      Yes
      Activate boot environment:     Yes
      Create backup boot environment: No
      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。

[59 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。

パッケージをアンインストールすると、[135 ページの「使用できない優先パスの設定: 結果と回復」](#)に記載されているように、メディアエーションされたリンクが削除される可能性があります。メディアエーションされたリンクが削除されるかどうかを確認するには、`-nv` または `-nvv` オプションを使用します。

ファイルを提供するパッケージがアンインストールされるときに特定の属性を持つファイルアクションがどのように処理されるかについては、『[Oracle Solaris 12 での Image Packaging System を使用したソフトウェアのパッケージ化と配布](#)』の「[File Actions](#)」または `pkg(7)` のマニュアルページを参照してください。

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、[149 ページの「操作履歴の表示」](#)を参照してください。

[48 ページの「パッケージインストールのプレビュー」](#)に示す例では、4つのパッケージがインストールされていました。ところが、次のコマンドは、コマンド行に名前を示すパッケージのみが `uninstall` によって削除されることを示しています。

```
$ pkg uninstall -v oracle-rdbms-server-18c-preinstall
      Packages to remove:      1
      Estimated space available: 852.20 GB
      Estimated space to be consumed: 455.25 MB
      Rebuild boot archive:      No

Changed packages:
solaris
  group/prerequisite/oracle/oracle-rdbms-server-18c-preinstall
    11.4-11.4.0.0.1.10.0 -> None
...
```

`pkg list` コマンドによれば、インストールされていたほかの3つのパッケージはまだインストールされていることを示しています。

```
$ pkg -R /mnt list oracle-rdbms-server-18c-preinstall system/kernel/oracka \
x11/diagnostic/x11-info-clients x11/library/libdmx
NAME (PUBLISHER)                                VERSION                                IFO
system/kernel/oracka                            11.4-11.4.0.0.1.10.1                  i--
x11/diagnostic/x11-info-clients                  7.7-11.4.0.0.1.10.0                  i--
x11/library/libdmx                               1.1.3-11.4.0.0.1.10.0                 i--
```

```
pkg list: no packages matching the following patterns are installed:
  oracle-rdbms-server-18c-preinstall
```

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、[149 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
2018-08-06T12:36:43  install             pkg                   Succeeded
$ pkg history -lt 2018-08-06T12:36:43
      Operation: install
...
      Command: /usr/bin/pkg install --be-name rdbms oracle-rdbms-server-18c-preinstall
...
```

```

      End State:
None -> pkg://solaris/x11/diagnostic/x11-info-clients@7.7,5.11-11.4.0.0.1.10.0:20180702T172622Z
None -> pkg://solaris/group/prerequisite/oracle/oracle-rdbms-server-18c-preinstall@11.4,5.11-11.4.0.0.1.10.0:20180702T172914Z
None -> pkg://solaris/system/kernel/oracka@11.4,5.11-11.4.0.0.1.10.1:20180702T144645Z
None -> pkg://solaris/x11/library/libdmx@1.1.3,5.11-11.4.0.0.1.10.0:20180702T172638Z

```

#### 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、`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 `git` 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 `git` 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 git
```

次のコマンドでは、`git` 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 git
```

#### 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すると、失敗します。

たとえば、`x11-info-clients` パッケージがまだインストールされている状態で `libdmx` パッケージをアンインストールしようすると、次のエラーメッセージを受け取ります。

```
$ pkg uninstall libdmx
pkg uninstall: Unable to remove 'x11/library/libdmx@1.1.3-11.4.0.0.1.10.0' due to the
following packages that depend on it:
  x11/diagnostic/x11-info-clients@7.7-11.4.0.0.1.10.0
```

libdmx をアンインストールするには、libdmx および x11-info-clients の両方をアンインストールする必要があります。単に x11-info-clients をアンインストールしても、依存パッケージ libdmx 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 x11-info-clients libdmx
```

**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、[138 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-56
...
$ pkg list mysql-56
pkg list: no packages matching the following patterns are installed:
  mysql-56
$ pkg avoid
  database/mysql-56 (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。

---

**注記** - 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 exact-install コマンドは、回避リストに記載されているパッケージをインストールしない制限を無視します。現在の BE の回避リストに登録されているパッケージが exact-install コマンドのオペランドとして指定されているか、依存関係チェーンに含まれている場合、パッケージは新しい BE にインストールされ、新しい BE は

空の回避リストを持つこととなります。回避リストについては、[138 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。現在の BE で凍結されているパッケージが `exact-install` コマンドのオペランドとして指定されているか、依存関係チェーンに含まれている場合、パッケージは `exact-install` コマンドによって指定されるバージョンで新しい BE にインストールされ、新しい BE は凍結されたパッケージを保持しません。凍結されたパッケージについては、[129 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。イメージバリエーションとファセットの設定、イメージプロパティ設定、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。[72 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、[55 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。

- [26 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。
- 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のバージョンで、インストールするパッケージの一覧に `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行い、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。
- 新しい BE が完全にテストされるまで、現在作業中の BE をバックアップとして常に保持してください。

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@11.4-11.4.0.0.1.10.0:20180702T173343Z
$ pkg exact-install --be-name 11.4.0min entire@11.4-11.4.0 solaris-minimal-server
```

---

注記 - 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より古いバージョンの pkg:/entire 制約パッケージを指定した場合、動作は保証されません。zpool バージョンの変更などのいくつかの変更によって、ダウングレードされた exact-install BE がブートに失敗したり予期しない動作を起こしたりする可能性があります。バージョンの差分が小さい方が、バージョンの差分が大きいよりも成功の可能性が高くなります。

---

## 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、[22 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、[72 ページの「大域ゾーンと非大域ゾーンの関係」](#) および [76 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。

[74 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。

ファセットとバリエーションを変更すると、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となります。

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、[52 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、[52 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、[76 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ア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、[177 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[179 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`、`file`、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。リポジトリバージョンについての詳細は、`pkgrepo(1)` のマニュアルページを参照してください。

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-proxy: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およびゾーンプロキシサービスの詳細は、`pkg.sysrepo(8)` 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、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、[121 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、『[Oracle Solaris ゾーンの作成と使用](#)』の「[Proxy Configuration on a System That Has Installed 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、[52 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 s12.0
Startup: Linked image publisher check ... Done
Startup: Refreshing catalog 'solaris' ... Done
Startup: Refreshing catalog 'isvpub' ... Done
Startup: Checking that pkg(7) 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.
```



## 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
  - Oracle Solaris 制約パッケージの使用
  - 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、[86 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エーション
  - 構成されたパッケージの署名プロパティ
  - 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。[22 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。[72 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、[24 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。

## ▼ Oracle Solaris 11 システムを更新する方法

システムのアップグレードは、pkg update によるワンステップのみで実行できます。

前提条件として、システムが必要なすべてのソフトウェアパッケージにアクセスできることを確認します。場合によってはフォローアップとして、pkg update を複数回行うことが必要なこともあります。

1. システムが必要なソフトウェアパッケージにアクセスできることを確認します。[82 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。



2. テスト更新を実行します。  
84 ページの「更新操作のプレビュー」を参照してください。
3. 実際の更新を実行します。  
--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。  
結果を指定します。更新するパッケージのバージョンの一部を指定します。

```
$ pkg update --be-name 11.4.0 entire@11.4-11.4.0
```

場合によっては、ソフトウェアのサブセットを更新しないとほかのソフトウェアを更新できないため、イメージのアップグレードに複数のステップが必要になることがあります。必要な最終状態にワンステップでアップグレードできない場合、できるかぎり更新し、新しい BE をリポートして、ふたたび更新してください。複数ステップのアップグレードが必要かどうかについては、リリースノートに記載されています。

## 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。37 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。31 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。82 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、パブリッシャーの特定の場所で 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、部分的なりポジトリを作成しないでください。『Oracle Solaris 12 パッケージリポジトリのコピーと作成』の説明に従って、完全なり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。84 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、85 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
```

更新で必要な内容が取得されていることを確認します。

```
$ pkg -R /mnt list -v entire
```

パッケージを検証します。

```
$ 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
```

62 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。
- 96 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

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。『[Oracle Solaris 12 パッケージリポジトリのコピーと作成](#)』の「[Best Practices for Creating and Using Local IPS Package Repositories](#)」を参照してください。更新されるシステムは、『[Oracle Solaris 12 パッケージリポジトリのコピーと作成](#)』の「[Minimal Required Repository](#)」に記載されているパッケージバージョンに少なくともアクセスできる必要があります。

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 11.3 から Oracle Solaris 11.4 に更新する場合は、インストールされている Oracle Solaris 11.3 パッケージと目的の Oracle Solaris 11.4 パッケージの両方へのアクセスが `solaris` パブリッシャーに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。

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。

システムにインストールされているバージョンを判別するには、次のコマンドを使用します。

```
$ pkg list entire
```

構成済みのパブリッシャーから利用できるバージョンを判別するには、次のコマンドを使用します。

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

このコマンドからの出力では、『Oracle Solaris 12 パッケージリポジトリのコピーと作成』の「Minimal Required Repository」に記載されているように、現在インストール済みのリリース、更新後のリリース、および必要な中間リリースが表示される必要があります。

`pkg:/entire` の現在インストール済みのバージョンが構成済みパブリッシャーからまだ利用可能であることを確認するには、次の例に示すように、有効化された特定のパブリッシャーの場所で特定のバージョンを確認してください。

```
$ pkgrepo -s /var/share/pkgrepos/solaris list entire@0.5.11-0.175.3.22.0.3.0
```

次のコマンドは、Oracle Solaris 11.3 SRU 22 がインストールされており、Oracle Solaris 11.3 SRU 25 および 28 が現在構成されている `solaris` パブリッシャーから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、18 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@0.5.11-0.175.3
NAME (PUBLISHER)          VERSION          IFO
entire                    0.5.11-0.175.3.28.0.4.0  ---
entire                    0.5.11-0.175.3.25.0.3.0  ---
entire                    0.5.11-0.175.3.22.0.3.0  i--
```

これらの中に必要なバージョンがない場合、または更新を完了するために必要な一部のリリースが欠落している場合は、`solaris` パブリッシャーの起点を別のパッケージリポジトリの場所に設定するか場所を追加する必要があります。システムに構成されているパブリッシャーを確認するには、次のコマンドを使用します。-n オプションは有効化されているパブリッシャーのみ表示することを指定します。

```
$ pkg publisher -n
```

`solaris` などの単一パブリッシャーは複数のリポジトリの場所を持つことができます。パブリッシャーに複数のリポジトリの場所が構成されている場合、117 ページの「パブリッシャーの起点の有効化および無効化」に記載されているように、-F オプションを使用して、すべての場所が有効かどうかを確認します。更新を完了するために必要な一部のパッケージバージョンが別のパッケージリポジトリにある場合、116 ページの「パブリッシャーの起点の追加と変更」に記載されているように、`pkg set-publisher` コマンドに -g オプションを使用することによって、それらのパッケージリポジトリの場所をパブリッシャーに追加します。

すべてのパブリッシャーリポジトリの場所がサイトにローカルである場合、それらのリポジトリにコンテンツを追加することが必要な場合もあります。手順については、『[Oracle Solaris 12 パッケージリポジトリのコピーと作成](#)』を参照してください。

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ク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。115 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。

パブリッシャーリポジトリの場所の 1 つが Oracle Solaris サポートリポジトリまたはその他のセキュアリポジトリの場合、キーおよび証明書が有効であることを確認します。

リポジトリへのシステムのアクセスがファイアウォールまたはプロキシによって妨げられていないことを確認します。

詳細は、113 ページの「[パブリッシャーの構成](#)」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。130 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](#)」に記載されているように、`version-lock` ファセットをロック解除した場合、これらの新しいバージョンをインストールできる可能性があります。ほかのソフトウェアが古いバージョンに依存しているという理由で、古いバージョンをアンインストールできない場合があります。多くの場合、複数バージョンのソフトウェアをシステムに同時にインストールできます。そのような場合、デフォルトバージョンは、133 ページの「[デフォルトのアプリケーション実装の指定](#)」に記載されているようにメタデータ設定によって判別されます。特定のバージョンが必要な場合、スクリプトまたはその他のソフトウェア内で、メタエーションされたリンクではなくフルパスを指定します。

Oracle Solaris のこのリリースで利用可能な特定の FOSS バージョンは、『[Oracle Solaris 11.4 で使用可能なフリーウェア](#)』の「[Oracle Solaris 11.4 Bundled Software Updates](#)」を参照してください。

## 更新操作のプレビュー

`pkg update` コマンドに次のオプションを使用します。

- 実際には操作を行わずに操作の結果を表示するには、プレビューオプション (`-n`) を使用します。`-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。

- 更新、削除、および追加されるパッケージとそのバージョン、変更されるメディアエータ、再開されるサービス、変更される編集可能ファイル、新しい BE が作成されるかどうかについては、詳細オプション (-v) を使用します。更新に成功しない場合、詳細オプションは問題の診断に役立つ多くの情報を表示します。さらに多くの情報を取得するには、-vv を使用します。

必要な結果を指定します。更新に成功しない場合、更新するパッケージのバージョンについて指定する情報が多くなるほど問題の診断に役立つ情報が多く表示されます。

次のコマンドでは、これらのオプションの使用が示されています。

```
$ pkg update -nv entire@11.4-11.4.0.0.1.12.0
```

更新操作に成功しない場合、受け取るエラーメッセージに応じて次のいくつかのステップを検討してください。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](#)も参照してください。

- `pkg freeze` コマンドを使用して、更新の妨げになっている可能性がある凍結されたパッケージがシステムに存在するかどうかを確認します。
- `pkg facet` コマンドを使用して、いずれかのパッケージでバージョン制約がロック解除されているかどうかを確認します。必要な更新を完了するにはその制約を再ロックすることが必要な場合があります。バージョンロックを設定する行為によって、システム更新が可能になります。
- 更新を制約している IDR がインストールされているかどうかを確認します。
- パブリッシャーリポジトリの場所で `pkgrepo verify` を実行します。必要な依存関係が欠落しているか、間違った権限が設定されている可能性があります。
- 更新しようとしているシステムで `pkg verify` を実行します。パッケージ化された内容が、システム上でサポートされない方法で変更されている可能性があります。たとえば、IPS パッケージによって提供されたリンクをディレクトリまたはファイルになるようにだれかが変更した場合、システム更新の妨げとなる可能性があります。

非大域ゾーンにあるソフトウェアが、親の大域ゾーンにあるソフトウェアと同じリリースであることが必要な場合、非大域ゾーンにあるソフトウェアは大域ゾーンを更新したときに自動的に更新されます。非大域ゾーンにあるソフトウェアが、親の大域ゾーンにあるソフトウェアと同じリリースでなくてもよい場合、非大域ゾーンにあるソフトウェアは別途更新する必要があります。

## 新しいブート環境の指定

新しい BE を要求したか、新しい BE で更新を行う必要がある場合、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、[141 ページの「ブート環境ポリシーイメージのプロパティ」](#)を参照してください。--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ることをお勧めします。

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、[51 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。
- Oracle Solaris 制約パッケージを使用します。制約パッケージを使用することが、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。
- カスタムの制約パッケージを使用します。制約パッケージを使用することが、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 11.3 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@0.5.11,5.11-0.175.3.13 '*'
  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 -> 0.5.11,5.11-0.175.3.13.0.4.0:
20160929T175502Z
...
```

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` コマンドに名前を追加するか、`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、[88 ページの「Oracle Solaris 制約パッケージの使用」](#) および [89 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 11.3 バージョンへの更新は許可するが、Oracle Solaris 11.4 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。`0.5.11,5.11-0.175.3` を指定すると、`entire` パッケージを 11.4 ではなく、たとえば `0.5.11,5.11-0.175.3.16` に更新できることを示します。

```
$ pkg freeze -c "Keep this image at 11.3." entire@0.5.11,5.11-0.175.3
entire was frozen at 0.5.11-0.175.3
$ pkg freeze
NAME      VERSION      DATE              COMMENT
entire    0.5.11-0.175.3  15 Nov 2016 13:30:44 PST  Keep this image at 11.3.
$ pkg list entire
```

```
NAME (PUBLISHER)  VERSION      IFO
entire            0.5.11-0.175.3.13.0.4.0  if-
```

パッケージの凍結の詳細は、[129 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 11.3 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、[88 ページの「Oracle Solaris 制約パッケージの使用」](#) および [89 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、[16 ページの「制約パッケージ」](#)を参照してください。

Oracle Solaris 11.4 には、`pkg:/entire` を Oracle Solaris 11.4 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-11.4
NAME (PUBLISHER)      SUMMARY
release/constraint/solaris-11.4  Constraint Package for Oracle Solaris 11.4
```

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 11.4 SRU など任意の Oracle Solaris 11.4 リリース (`entire@11.4-11.4`) 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-11.4
set name=pkg.fmri value=pkg://solaris/release/constraint/solaris-11.4@0,5,11:20170724T163053Z
set name=pkg.summary value="Constraint Package for Oracle Solaris 11.4"
set name=variant.arch value=sparc value=i386
set name=pkg.depend.install-hold value=core-os
depend fmri=entire@11.4-11.4 type=incorporate
depend fmri=entire type=require
```

- `solaris-11.4` 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 11.4 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-11.4` 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-11.4` 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があります。

- 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に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 `upgradectl.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=upgradectl@1.0
set name=pkg.summary value="Package 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 upgradectl` 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

upgradectl 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

upgradectl 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、『[Oracle Solaris 12 パッケージリポジトリのコピーと作成](#)』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 プロパティの値です。

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

upgradectl パッケージを開発リポジトリに公開します。

```
$ pkgsend -s myrepo publish upgradectl.p5m
```

```
pkg://site/upgradectl@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          0 VERSION
site    upgradectl      1.0,5.11:20131104T072336Z
$ pkg list -vg myrepo
FMRI                                IFO
pkg://site/upgradectl@1.0,5.11:20131104T072336Z  ---
```

0 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 upgradectl
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 upgradectl
Name: upgradectl
Summary: Package 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/upgradectl@1.0,5.11:20131120T010105Z
$ pkg contents -Hro fmri -t depend -a type=incorporate upgradectl
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、『[Oracle Solaris 12 でのImage Packaging System を使用したソフトウェアのパッケージ化と配布](#)』の「[Creating and Publishing a Package](#)」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 upgradectl
      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/upgradectl@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、upgradectl.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=upgradectl@1.10
set name=pkg.summary value="Package 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 upgradectl.p5m
pkg://site/upgradectl@1.10,5.11:20131120T021902Z
PUBLISHED
$ pkgrepo -s myrepo list
PUBLISHER NAME                                0 VERSION
site      upgradectl                          1.10,5.11:20131120T021902Z
site      upgradectl                          1.0,5.11:20131120T010105Z
$ pkgrecv -s myrepo -d /var/share/pkgrepos/solaris upgradectl
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/upgradectl
NAME (PUBLISHER)          VERSION          IFO
upgradectl (site)        1.10            ---
upgradectl (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、`upgradectl` パッケージの `pkg.depend.install-hold` 設定により、`upgradectl` 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 `upgradectl` 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 upgradectl
NAME (PUBLISHER)                                VERSION      IFO
entire                                           0.5.11-0.175.1.0.0.24.2  i--
upgradectl (site)                              1.0         i--
$ beadm mount s11u1_10 /mnt
$ pkg -R /mnt list entire upgradectl
NAME (PUBLISHER)                                VERSION      IFO
entire                                           0.5.11-0.175.1.10.0.6.0  i--
upgradectl (site)                              1.10        i--
$ beadm unmount s11u1_10
```

## Oracle Solaris 11.4 用の別のパッケージリポジトリの維持

Oracle Solaris 11.3 以前のシステムを Oracle Solaris 11.4 にアップグレードさせないようにするための 1 つの方法は、Oracle Solaris 11.4 へのアクセスを提供しないことです。Oracle Solaris 11.3 と Oracle Solaris 11.4 の IPS パッケージリポジトリを別々に用意し、各システムでパブリッシャーを適切に構成します。

### Oracle Solaris 11.4 へのアップグレードの準備ができていない Oracle Solaris 11.3 システム

これらのシステムで Oracle URL にある IPS パッケージリポジトリが使用されている場合、『[Oracle Solaris 12 パッケージリポジトリのコピーと作成](#)』に記載されているように、ローカルの Oracle Solaris 11.3 パッケージリポジトリを作成します。

- 最新の公開リリースおよび後続のすべての SRU のためのリポジトリを Oracle Solaris 11.3 リポジトリアーカイブファイルから作成します。リポジトリには、リポジトリを使用するすべての Oracle Solaris 11.3 システム上にインストールされるすべてのパッケージを含める必要があります。Oracle Solaris 11.3 パッケージリポジトリを Oracle Solaris 11.4 パッケージリポジトリのサブセットから作成しないでください。
- SRU リポジトリアーカイブファイルから新しい Oracle Solaris 11.3 SRU を追加することによってリポジトリを保守します。Oracle URL をソースとして pkg/mirror サービスを使用してこのリポジトリを保守しないでください。
- Oracle Solaris 11.3 システムの solaris パブリッシャーをリセットして、この新しいローカルの Oracle Solaris 11.3 リポジトリの場所が唯一の起点であるようにします。

### Oracle Solaris 11.4 にアップグレード中の Oracle Solaris 11.3 システム

88 ページの「[Oracle Solaris 制約パッケージの使用](#)」に記載されているような制約パッケージなどの制約が適用されていない場合、Oracle URL で IPS パッケージリポジトリを使用しているシステムは Oracle Solaris 11.4 にアップグレードされます。

システムで使用中のローカルリポジトリが、Oracle Solaris 11.4 へのアップグレードの準備ができていないシステムによって共有されている場合、次のいずれかのオプションを使用します。

- solaris パブリッシャーの起点を Oracle Solaris リポジトリ URL に変更します。

- 『Oracle Solaris 12 パッケージリポジトリのコピーと作成』に記載されているように、新しいローカルの Oracle Solaris 11.4 パッケージリポジトリを作成します。これらのシステム上の solaris パブリッシャーをリセットし、この新しいローカルの Oracle Solaris 11.4 リポジトリ用の起点を、ローカルの Oracle Solaris 11.3 リポジトリ用の既存の起点に追加します。

## 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、必要な Oracle Solaris バージョンでカーネルゾーンまたは Oracle VM Server をインストールすることもできます。

## 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、[98 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。

Oracle Solaris ではこれらの修正を提供するための Support Repository Updates (SRU) が用意されています。SRU は 3 つごとにクリティカルパッチ更新 (CPU SRU) になります。CPU SRU のリリースのタイミングは、ほかの Oracle 製品のクリティカルパッチ更新のリリースと同じです。

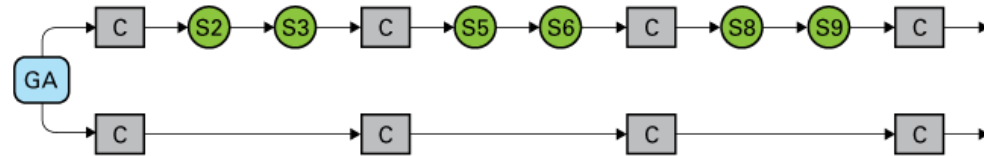
次の図は 2 つのシステムのアップグレード戦略を示しています。図で、GA は Oracle Solaris 11.3 や Oracle Solaris 11.4 などのリリース、S は SRU、C は CPU SRU です。

- 重要なセキュリティ修正でシステムを最新状態に維持するための最善の方法は、新しい SRU が利用可能になるたびに更新することです。
- 毎月リブートできない場合、Oracle クリティカルパッチ更新に合わせて少なくとも四半期ごとに更新します。

CPU SRU またはその他の SRU に更新すると、それ以前のすべての SRU で提供されたすべての修正および機能拡張が取得されます。



図 1 月次の SRU または四半期ごとの CPU システム更新



次の表では、SRU と CPU SRU の違いについて説明します。

表 4 SRU と CPU の比較

特徴	SRU	CPU SRU
更新リリースの頻度	月ごと	四半期ごと  CPU のリリーススケジュールについては、Oracle Technology Network で <a href="#">Critical Patch Updates</a> を参照してください。
更新内容	あらゆる種類の修正または機能拡張  新しい修正に加えて、各 SRU では、以前の Oracle Solaris リリースおよび現在のリリースについての以前のすべての SRU からのすべての修正および機能強化が提供されます。たとえば、Oracle Solaris 11.2 SRU 14 から Oracle Solaris 11.3 SRU 2 への更新では、Oracle Solaris 11.2 SRU 15 および Oracle Solaris 11.3 GA、SRU 1、および SRU 2 からのすべての修正がインストールされます。	新しい修正は、共通脆弱性 (CVE) の修正などの重大な修正に限定されます。  以前の Oracle Solaris リリースおよび現在のリリースについての以前のすべての SRU からのすべての修正および機能強化が提供されますが、この SRU に導入される新しい変更は重大な修正のみです。
再起動が必要かどうか	はい	はい
新しいシリーズの開始	Oracle Solaris リリースごと (たとえば、Oracle Solaris 11.2、Oracle Solaris 11.3、Oracle Solaris 11.4)	CPU SRU は SRU シリーズの一部です。3 回に 1 回の SRU は CPU SRU です。
シリーズの長さ	『Solaris Operating System End Of Life Matrix』(Doc ID <a href="#">1001343.1</a> ) に記載されている、関連する Oracle Solaris リリースと同じです。SRU は、後続のリリースが利用可能になるとリリース用に提供されなくなる場合があります。  ある Oracle Solaris リリースの特定の SRU から、別の Oracle Solaris リリースの特定の SRU に直接更新できます。	CPU SRU は SRU シリーズの一部です。3 回に 1 回の SRU は CPU SRU です。

特徴	SRU	CPU SRU
	「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 <a href="#">1391762.1</a> ) 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、<https://support.oracle.com/> で「Oracle Solaris 11.4 Support Repository Updates (SRU) Index」を検索します。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、[102 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。

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およびリポジトリに必要な最低限の内容については、『[Oracle Solaris 12 パッケージリポジトリのコピーと作成](#)』を参照してください。

[79 ページの「イメージの更新の概要」](#)に記載されているように、更新を実行します。最新の SRU リリースよりも古い SRU に更新するには、[86 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。

[表4](#)に記載されているように、各 SRU は以前リリースされた SRU によって提供されたすべての修正および機能拡張を含んでいますが、SRU は別の SRU を含んでおらず、SRU は 1 つのバージョンの `pkg:/entire` のみを含みます。システムを特定の SRU に更新するには、Oracle Solaris サポートリポジトリを使用するか、その SRU 用のリポジトリファイルの内容をローカルリポジトリに追加することによって、その SRU へのアクセスが必要です。

たとえば、SRU 28 リポジトリ内容をローカルリポジトリに追加していないが、SRU 29 リポジトリ内容を追加した場合、このリリースのすべての SRU で最初に提供されたすべての修正を SRU 29 を通じて利用できますが、システムを SRU 28 レベルに更新することはできません。クエリーによって、ローカルリポジトリに `entire@0.5.11-0.175.3.29` が含まれていても、`entire@0.5.11-0.175.3.28` は含まれていないことが表示されます。[82 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。

## 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は個々の月次 SRU で利用できます。このパッケージのほとんどの内容は、その SRU で提供される CVE 修正に関する情報です。

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

表 5 solaris-11-cpu パッケージのバージョン文字列コンポーネント

コンポーネント	説明
YYYY	この CPU パッケージに関連付けられた SRU がリリースされた年。
MM	この CPU パッケージに関連付けられた 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。

`solaris-11-cpu` パッケージはデフォルトでインストールされていません。このパッケージが必要な場合は、明示的にインストールする必要があります。このパッケージは、新しい SRU に更新する際は不要です。このパッケージをインストールする利点は次のとおりです。

- このシステムで修正された CVE を簡単に一覧表示できます。
- このシステムで実行している SRU を簡単に表示できます。
- このパッケージを特定の SRU バージョンに更新することによって、特定の SRU に簡単にアップグレードできます。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。
- これらの CVE を修正するために必要なすべてのパッケージが正しいバージョンでインストールされていることを確認できます。

次のコマンドは、このシステムに `solaris-11-cpu` パッケージがインストールされている場合に、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:
```

このシステムに `solaris-11-cpu` パッケージがインストールされていない場合、インストールされている SRU のための `solaris-11-cpu` パッケージを識別し、そのパッケージをリモートからクエリー検索します。たとえば、このシステムで 2018 年 1 月にリリースされた Oracle Solaris 11.3 SRU 28 が実行されている場合、対応する `solaris-11-cpu` パッケージは `solaris-11-cpu@2018.1` です。

```
$ pkg contents -ro value -t set -a name=info.cve solaris-11-cpu@2018.1
```

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
```

利用可能な最新の SRU に更新するか、指定された SRU に更新してその SRU 用の新しい修正および機能拡張をインストールするには、`pkg update` コマンドを使用します。

```
$ pkg update --be-name Solaris-11.3-SRU30 solaris-11-cpu@2018.3 '**'
```

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 の詳細は、『[Oracle Solaris 12 セキュリティコンプライアンスガイド](#)』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/system/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。

利用可能な最新のファームウェア更新パッケージを識別するには、次のコマンドを使用します。

```
$ pkg list -n '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。古い更新をインストールする必要はありません。最新の更新のみインストールし、その後はそこから更新を継続できます。

`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。

#### 例 10 SPARC T7 ファームウェア更新についての情報の表示

`firmware/system/T7-4` パッケージは `T7-4/sysfw9-7` パッケージおよび `T7-4/hwprogrammables1-0-7` パッケージをインストールします。

```
$ pkg install -nv firmware/system/T7-4
  Packages to install:      3
  Estimated space available: 830.81 GB
  Estimated space to be consumed: 367.54 MB
  Create boot environment: No
  Create backup boot environment: No
  Rebuild boot archive:    No
```

Changed packages:

```
solaris
  firmware/system/T7-4
    None -> 1.1.0.4.0,5.11:20170726T000710Z
  firmware/system/T7-4/hwprogrammables1-0-7
    None -> 1.0.7,5.11:20160817T161305Z
  firmware/system/T7-4/sysfw9-7
    None -> 9.7.6.2,5.11:20170726T000706Z
```

Release Notes:

Sun System Firmware component revisions included with this release:

```
Sun System Firmware 9.7.6.b 2017/07/10 15:33
-----
ILOM 3.2.10.1.b Mon Jul 10 16:22:42 PDT 2017
Hostconfig 1.8.6 2017/06/07 09:34
POST 5.5.6 2017/06/07 09:22
FWP 1.3.6 2017/06/07 09:15
Hypervisor 1.17.6.a 2017/06/14 10:14
OpenBoot 4.40.6 2017/06/07 09:00
GM 1.8.6 2017/06/07 09:40
```

```
Checksum of Sun_System_Firmware-9_7_6_b-SPARC_T7-4.pkg : 2715457817  
(generated by the /usr/bin/cksum command)
```

pkg info コマンドは、firmware/system/T7-4/sysfw9-7 パッケージがパッチ p26407491\_976b を提供することを示します。

```
$ pkg info -r firmware/system/T7-4/sysfw9-7  
Name: firmware/system/T7-4/sysfw9-7  
Summary: T7-4 firmware  
Description: T7-4 firmware: MOS Patch p26407491_976b  
State: Not installed  
Publisher: solaris  
Version: 9.7.6.2 (9.7.6.b)  
Build Release: 5.11  
Branch: None  
Packaging Date: July 26, 2017 12:07:06 AM  
Size: 33.62 MB  
FMRI: pkg://solaris/firmware/system/T7-4/sysfw9-7@9.7.6.2,5.11:  
20170726T000706Z
```

/var/firmware/system/T7-4/sysfw9-7/p26407491\_976b/README.html の手順に従ってファームウェア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](#)』または使用している [SPARC サーバー管理者ガイド](#) 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利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、[96 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 11 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 16 「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                                0 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 12 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、18 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11「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 13** 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 14** 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12「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 15** IDR のリリースノートを表示する

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。149 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、105 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 16** IDR の名前が変更されているかどうかを表示する

例11「[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。『[Oracle Solaris 12 パッケージリポジトリのコピーと作成](#)』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17「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/svccadm` 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 17 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、[105 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16 「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 18 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、[86 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 19** 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15「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 11.4 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 idr1929 \
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。





## 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
- バリエーション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およびパブリッシャープロパティの構成
- 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、[24 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する必要があります。

Ops Center を使用している場合は、[160 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.example.com/
```

TYPE 列は、LOCATION 値が起点かミラーかを示します。詳細は、[22 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.example.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、[121 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: https://pkg.oracle.com/solaris/support/
Origin Status: Online
SSL Key: /var/pkg/ssl/458a3a3690dbdd688dc47aa2e66317b7ecc3ff12
SSL Cert: /var/pkg/ssl/01bffa3f4f7bc8cf164e8719fbf1a3a43a82f893
Cert. Effective Date: Thu Jun 22 17:09:43 2017
Cert. Expiration Date: Sun Jun 30 17:09:43 2019
```

```
Client UUID: f892c512-6c3f-11e5-befc-7f447de68cea
Catalog Updated: Mon Aug 06 19:36:55 2018
Enabled: Yes
```

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 https://pkg.oracle.com/solaris/support/
```

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 https://pkg.oracle.com/solaris/support/
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したりする
- 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せん。

## 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 20 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.example.com/release/ devtool
```

### 例 21 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。

```
$ pkg publisher
PUBLISHER          TYPE      STATUS P LOCATION
solaris             origin   online F http://pkg.oracle.com/solaris/release/
$ pkgrepo -s /var/share/pkgrepos/myrepo/ info
PUBLISHER  PACKAGES STATUS      UPDATED
site        7         online    2018-04-14T18:32:46.854674Z
solaris     6143      online    2018-04-15T00:45:52.227891Z
solarisstudio 112       online    2018-04-12T19:29:52.173011Z
```

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。

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

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
```

## パブリッシャーの起点の有効化および無効化

次の例では、新しい起点が使用不可な起点として追加されます。solaris パブリッシャーおよび元の起点はすでに有効化されています。

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

`/var/share/pkgrepos/solaris/` 起点は無効化されており、このことは STATUS 列が `disabled` と表示され、Origin Status 行が `Disabled` で、ENABLED 列が `false` であることによって示されています。

```
$ pkg publisher
PUBLISHER          TYPE      STATUS  P LOCATION
solaris             origin   disabled F file:///var/share/pkgrepos/solaris/
solaris             origin   online  F http://pkg.oracle.com/solaris/release/
$ pkg publisher solaris
  Publisher: solaris
  Alias:
  Origin URI: file:///var/share/pkgrepos/solaris/
  Origin Status: Disabled
  SSL Key: None
  SSL Cert: None
  Origin URI: http://pkg.oracle.com/solaris/release/
  Origin Status: Online
  SSL Key: None
  SSL Cert: None
  Client UUID: 97f0e8c2-217e-11e4-b444-13b4eae0609
  Catalog Updated: Wed Aug 02 05:10:59 2017
  Enabled: Yes
  Properties:
    signature-policy = verify
```

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

次の例では、無効化されている既存の起点が有効化されます。

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

次の2つのコマンドは同等です。

```
$ pkg set-publisher -g /var/share/pkgrepos/solaris solaris
$ pkg set-publisher --enable -g /var/share/pkgrepos/solaris solaris
```

次の2つのコマンドは同等の効果があります。

```
$ pkg set-publisher --disable solaris
$ pkg set-publisher --disable -g '*' solaris
```

## 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、[22 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://pkg2.example.com/ devtool
$ pkg publisher
PUBLISHER TYPE STATUS P LOCATION
devtool origin online F http://pkg.example.com/
devtool mirror online F http://pkg2.example.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の構成

パブリッシャープロパティ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` プロパティは、複数の値を取ることができます。

[144 ページの「パッケージの署名プロパティの構成」](#) 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 22**           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。[114 ページの「パブリッシャー情報の表示」](#) および [161 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 23                   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。プロキシを設定するための詳細およびさまざまな方法については、[121 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を設定または表示することはできません。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。

次の例で、`proxyURI` の値は次のようになり、`protocol` は `http` または `https` で、`:port` はオプションです。

```
protocol://host-name[:port]
```

`--proxy` オプションを使用して認証済みプロキシを設定することはできません。次の形式の `proxyURI` 値はサポートされていません。

```
protocol://user:password@host
```

```
$ 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。72 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。

## 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および受け入れられる環境変数名のリストについては、curl(1) 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。

次の例で、proxyURI の値は次のようになります。

```
[http://|https://](host-name|ipv4_address|[ipv6_address]):port
```

- スキームは http:// または https:// のいずれかとすることができます。スキームが指定されない場合、http:// とみなされます。
- プロキシにはホスト名またはドット付き数値 IP アドレスを含める必要があります。
  - 数値 IPv6 アドレスは、角括弧 [] で囲む必要があります。
- ポート番号を指定するには、ホスト名の末尾に :port を付けます。指定しない場合、デフォルトポートは 1080 です。

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。

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。

```
$ svccfg -s system-repository:default setprop config/http_proxy = 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。

## 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エーションと呼ばれます。

## ファセットおよびバリエーションの値がパッケージのインストールに与える影響

ファセットとバリエーションのどちらも次の 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
- 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エーションタグの値と、イメージに設定されたファセットおよびバリエーションプロパティの値の比較により、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。

ファセットとバリエーションのプロパティ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エーションタグを付けることができます。

ファセットとバリエーションのプロパティ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。一部のバリエーションプロパティ値は、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エーションのプロパティ値が、特定のアクションがインストールされるかどうか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エーション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エーションタグのあるアクションは、すべてのバリエーションタグの値がイメージに設定されている対応するバリエーションプロパティ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 つのバリエーション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エーションプロパティ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エーションタグの両方があるアクションは、ファセットとバリエーションの両方のプロパティ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。

## バリエーション値の表示と変更

ほとんどのバリエーション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エーションタグの値が、イメージ内の対応するバリエーションプロパティの値と一致している必要があります。`arch` および `zone` バリエーションプロパティ値は、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.*` バリエントの値と不明なバリエントの値は `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。不明なバリエントとは、パッケージマニフェスト内のアクションのバリエ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エントです。

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` バリエ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エントプロパティ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エ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エ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エ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
```

バリエントプロパティ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** - バリエントプロパティ値を変更すると、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、[141 ページの「ブート環境ポリシーイメージのプロパティ」](#)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、[123 ページの「ファセットおよびバリエーション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。168 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、[141 ページの「ブート環境ポリシーイメージのプロパティ」](#)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る、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せん。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。

```
$ pkg list openssl
NAME (PUBLISHER)                VERSION                IFO
library/security/openssl       1.0.2.15-11.4.0.0.1.10.0  i--
$ pkg freeze -c "Prevent update until testing complete." openssl
library/security/openssl was frozen at 1.0.2.15-11.4.0.0.1.10.0:20180702T170444Z
```

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。

```
$ pkg freeze
NAME                VERSION                DATE
COMMENT
library/security/openssl 1.0.2.15-11.4.0.0.1.10.0:20180702T170444Z 06 Aug 2018 15:56:16 PDT
Prevent
update until testing complete.
```

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 list openssl
NAME (PUBLISHER)                VERSION                IFO
library/security/openssl       1.0.2.15-11.4.0.0.1.10.0  if-
```

pkg info 出力の「State」フィールドは、このパッケージが凍結されているかどうかを示します。

```
$ pkg info openssl
Name: library/security/openssl
Summary: OpenSSL - a Toolkit for Secure Sockets Layer (SSL v2/v3) and
Transport Layer (TLS v1) protocols and general purpose
cryptographic library
Description: OpenSSL is a full-featured toolkit implementing the Secure
Sockets Layer (SSL v2/v3) and Transport Layer Security (TLS v1)
protocols as well as a full-strength general purpose
cryptography library.
Category: System/Security
State: Installed (Frozen)
Publisher: solaris
Version: 1.0.2.15 (1.0.20)
Branch: 11.4.0.0.1.10.0
Packaging Date: Mon Jul 02 17:04:44 2018
Size: 16.05 MB
FMRI: pkg://solaris/library/security/openssl@1.0.2.15-11.4.0.0.1.10.0:
20180702T170444Z
Project URL: https://www.openssl.org/
Source URL: https://www.openssl.org/source/openssl-1.0.20.tar.gz
```

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、凍結に関するメッセージが表示されます。

```
$ pkg list -n openssl
NAME (PUBLISHER)                                VERSION                                         IFO
library/security/openssl                        1.0.2.15-11.4.0.0.1.12.0                    ---
$ pkg update openssl
No updates available for this image.
$ pkg update openssl@1.0.2.15-11.4.0.0.1.12.0
Creating Plan (Solver setup): /
pkg update: No matching version of library/security/openssl can be installed:
  Reject: pkg://solaris/library/security/openssl@1.0.2.15-11.4.0.0.1.12.0
  Reason: This version is excluded by a freeze on library/security/openssl at
version 1.0.2.15-11.4.0.0.1.10.0:20180702T170444Z. The reason for the freeze is:
Prevent update until testing complete.
```

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、[16 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。[96 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、[141 ページの「ブート環境ポリシーイメージのプロパティ」](#)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。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。

あるバージョンをときどき使用し、それ以外は別のバージョンを使用する必要がある場合、メディエーションされたリンクを変更することではなくフルパスを指定することを検討してください。

## 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が、`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。一覧表示される内容は、今までにインストールした内容によって異なります。

```
$ pkg mediator
MEDIATOR   VER. SRC.  VERSION IMPL. SRC. IMPLEMENTATION
apache     system    2.4      system
apr-1-config system    1.5      system
apu-1-config system    1.5      system
csh        vendor    vendor    suncsh
```

```
gcc          system    5      system
java         system    1.8    system
openssl     vendor    vendor  default
perl        vendor    5.22   vendor
python      local     2.7    vendor
ruby        vendor    2.1    vendor
sendmail    vendor    vendor  sendmail
which       vendor    vendor  gnu
```

-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 1.8    system
java      system 1.7    system
```

次の出力は、java メディエーションを提供するすべてのパッケージを示しています。

```
$ pkg search -lo pkg.name,mediator-version link:mediator:java
PKG.NAME          MEDIATOR-VERSION
runtime/java/jre-7 1.7
runtime/java/jre-8 1.8
```

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 1.8.0\_121 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-7       Java Platform Standard Edition Runtime Environment (1.7.0_131-b12)
runtime/java/jre-8       Java Platform Standard Edition Runtime Environment (1.8.0_121-b13)
$ java -version
java version "1.8.0_121"
Java(TM) SE Runtime Environment (build 1.8.0_121-b13)
Java HotSpot(TM) 64-Bit Server VM (build 25.121-b13, mixed mode)
```

次の出力は、メディエーションされたリンク (PATH) およびリンク先 (TARGET) を示しています。

```
$ pkg contents -o mediator-version,path,target -t link -a mediator=java jre-8 jre-7
MEDIATOR-VERSION PATH          TARGET
1.7              usr/java      jdk/jdk1.7.0_131
1.8              usr/java      jdk/jdk1.8.0_121
1.7              usr/jdk/jdk1.7.0_131 instances/jdk1.7.0
1.8              usr/jdk/jdk1.8.0_121 instances/jdk1.8.0
1.7              usr/jdk/latest jdk1.7.0_131
1.8              usr/jdk/latest jdk1.8.0_121
$ ls -l /usr/java
lrwxrwxrwx  1 root    root          16 Mar 23 16:11 /usr/java -> jdk/jdk1.8.0_121/
```

jre-8 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-7 パッケージでは、リンクのターゲットは jdk1.6.0 です。jre-8 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 1.8 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。

## 現在利用できないメディエーション参加者の識別

次の出力は、`openssl` メディエーション内で現在利用できるすべての参加者を示しています。

```
$ pkg mediator -a openssl
MEDIATOR  VER. SRC. VERSION IMPL. SRC. IMPLEMENTATION
openssl   vendor          vendor  default
```

次の出力は、`openssl-fips-140` パッケージまたは `apache-ssl-fips-140` パッケージをインストールしたあとで、`openssl` の `fips-140` 実装が利用可能になることを示しています (`apache-ssl-fips-140` パッケージが `openssl-fips-140` パッケージをインストールします)。

```
$ pkg search -o pkg.name,mediator-implementation link:mediator:openssl
PKG.NAME                                MEDIATOR-IMPLEMENTATION
web/server/apache-24/module/apache-ssl  default
web/server/apache-24/module/apache-ssl-fips-140  fips-140
library/security/openssl                 default
library/security/openssl/openssl-fips-140       fips-140
```

## 使用できない優先パスの設定: 結果と回復

次の出力は、`perl` コマンドについてはこのイメージで 1 つの実装のみが利用できることを示しています。

```
$ pkg mediator -a perl
MEDIATOR  VER. SRC. VERSION IMPL. SRC. IMPLEMENTATION
perl      vendor  5.22  vendor
perl      system  5.12  system
```

利用できる実装は 1 つだけですが、パスは依然としてメディエーションされたリンクです。

```
$ which perl
/usr/bin/perl
$ ls -l /usr/bin/perl
lrwxrwxrwx  1 root    root          22 Mar 23 16:06 /usr/bin/perl -> ../perl5/5.12/bin/perl
```

新しいバージョンが存在することがわかっているため、次のコマンドを入力して新しいバージョンを指定するリンクを更新します。次のコマンドは、指定されたバージョンがこのイメージで現在利用できないバージョンであるため正しくありません。

```
$ pkg set-mediator -vV 5.22 perl
```

```

        Packages to change:      2
        Mediators to change:     1
        Estimated space available: 867.06 GB
        Estimated space to be consumed: 241.78 MB
        Create boot environment:  No
        Create backup boot environment:  Yes
        Rebuild boot archive:     No

Changed mediators:
mediator perl:
    version: 5.12 (vendor default) -> 5.22 (local default)
...

```

pkg mediator コマンドは、更新された設定を表示します。

```

$ pkg mediator perl
MEDIATOR  VER. SRC. VERSION IMPL. SRC. IMPLEMENTATION
perl      local  5.22  system

```

ところが、pkg mediator -a コマンドでは、現在の設定が利用できないことが示されています。

```

$ pkg mediator -a perl
MEDIATOR  VER. SRC. VERSION IMPL. SRC. IMPLEMENTATION
perl      vendor 5.12  vendor

```

バージョン指定が正しくないため、メディエーションされたリンクが削除されています。

```

$ which perl
no perl in /usr/sbin /usr/bin
$ ls -l /usr/bin/perl
/usr/bin/perl: No such file or directory

```

この問題は、コマンドへのフルパス (/usr/per15/5.12/bin/perl) を指定することによって回避できます。ターゲットパスは影響を受けません。メディエーションされたリンクのみが削除されます。

この問題は、次のいずれかのアクションによって修正されます。

- 優先バージョンを利用可能な元のバージョンに戻す。

```
$ pkg set-mediator -V 5.12 perl
```

- pkg unset-mediator コマンドを使用して、システムが利用可能な実装から新しい実装を選択できるようにします。

```
$ pkg unset-mediator perl
```

- 必要な更新済みバージョンを提供するパッケージをインストールする。

次のコマンドの出力は、perl のメディエーションされたリンクを提供したパッケージを表示します。

```
$ pkg search -o pkg.name,mediator-version link:mediator:perl
```

runtime/perl-522 パッケージおよび terminal/cssh-522 パッケージによって、バージョン 5.22 の perl のメディエーションされたリンクが提供されま



す。terminal/cssh-522 パッケージによって runtime/perl-522 パッケージがインストールされます。問題を修正するために必要なことは、terminal/cssh-522 をインストールすることだけです。

```
$ pkg install terminal/cssh-522
```

問題を修正または回避できない場合、pkg set-mediator コマンドを使用したときに作成されたバックアップ BE にリポートしてください。

## 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。pkg mediator -a によって表示されないメディエーション参加者を指定しないでください。



**注意** - タイプミスをしたか、現在利用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[135 ページの「使用できない優先パスの設定: 結果と回復」](#)に示すように削除されます。

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。

次の出力では、java メディエーションの現在選択されている優先バージョンがバージョン 1.8 であること、およびバージョン 1.7 もこのイメージで利用可能であることが示されています。

```
$ pkg mediator java
MEDIATOR VER. SRC. VERSION IMPL. SRC. IMPLEMENTATION
java      system  1.8    system
$ pkg mediator -a java
MEDIATOR  VER. SRC. VERSION IMPL. SRC. IMPLEMENTATION
java      system  1.8    system
java      system  1.7    system
```

次のコマンドは、バージョン 1.7 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 1.7 が呼び出されることを意味します。ユーザーが JRE version 1.8 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 set-mediator -v -V 1.7 java
    Packages to change:      3
    Mediators to change:    1
    Estimated space available: 867.06 GB
    Estimated space to be consumed: 238.26 MB
    Create boot environment: No
    Create backup boot environment: Yes
    Rebuild boot archive:   No
...
$ pkg mediator java
MEDIATOR VER. SRC. VERSION IMPL. SRC. IMPLEMENTATION
java     local  1.7     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。

## 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 24 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              5.12-5.12.0.0.0.110.0 ---
$ pkg list -a `pkg contents -o fmri -Hrt depend -a type=group group/feature/amp`
NAME (PUBLISHER)                VERSION                IFO
database/mysql-56              5.6.29-5.12.0.0.0.110.0 ---
web/php-56                     5.6.22-5.12.0.0.0.110.0 ---
web/server/apache-24           2.4.23-5.12.0.0.0.110.0 i--
web/server/apache-24/module/apache-dbd 2.4.23-5.12.0.0.0.110.0 ---
web/server/apache-24/module/apache-fcgid 2.3.9-5.12.0.0.0.110.0 ---
web/server/apache-24/module/apache-ldap 2.4.23-5.12.0.0.0.110.0 ---
web/server/apache-24/module/apache-ssl 2.4.23-5.12.0.0.0.110.0 i--
```

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 mysql-56
$ pkg avoid
  database/mysql-56
```

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-56              5.6.29-5.12.0.0.0.110.0 ---
web/php-56                     5.6.22-5.12.0.0.0.110.0 i--
web/server/apache-24           2.4.23-5.12.0.0.0.110.0 i--
web/server/apache-24/module/apache-dbd 2.4.23-5.12.0.0.0.110.0 i--
web/server/apache-24/module/apache-fcgid 2.3.9-5.12.0.0.0.110.0 i--
```

```
web/server/apache-24/module/apache-ldap      2.4.23-5.12.0.0.0.110.0  i--
web/server/apache-24/module/apache-ssl      2.4.23-5.12.0.0.0.110.0  i--
```

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
  database/mysql-56 (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 mysql-56
pkg unavoid: The following packages are a target of group dependencies; use install to
unavoid these:
  database/mysql-56
$ pkg install mysql-56
$ 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 mysql-56
pkg avoid: The following packages are already installed in this image; use uninstall to
avoid these:
  database/mysql-56
$ pkg uninstall mysql-56
$ pkg avoid
  database/mysql-56 (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-56
$ pkg unavail mysql-56
$ pkg avoid
$
```

## イメージとパブリッシャーのプロパティの構成

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と、これらのプロパティの設定方法について説明します。イメージプロパティの説明については、`pkg(1)`のマニュアルページのイメージプロパティに関するセクションも参照してください。

イメージが最後に変更された時間を知る必要があるデーモンまたはほかのプログラムは、ファイル `/var/pkg/modified` のタイムスタンプを参照できます。 `/var/pkg/`

`modified` のタイムスタンプは、イメージを変更する操作が発生するたびに更新されます。その時点の変更に関する情報を表示するには、`pkg history` コマンドを使用します。詳細は、[149 ページの「操作履歴の表示」](#)を参照してください。

## ブート環境ポリシーイメージのプロパティ

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、バリエーション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、および更新したり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、『[Oracle Solaris 12 ブート環境の作成と管理](#)』を参照してください。

次のイメージプロパティの設定方法については、[147 ページの「イメージのプロパティの設定」](#)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せん。

##### 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

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とパブリッシャープロパティを、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

次のイメージプロパティを、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

次のパブリッシャープロパティを、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

このプロパティの機能は、このプロパティ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の機能と同じです。

#### signature-required-names

このプロパティの機能は、このプロパティ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の機能と同じです。

## パッケージの署名プロパティの構成

パッケージの署名プロパティを構成するには、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

### ca-path

SSL 操作の 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 `CPath` 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、発行時よりも後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の実行方法については、`pkg.depotd(8)` のマニュアルページ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 を供給しないパッケージクライアント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(8)` のマニュアルページ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の設定

イメージプロパティ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の値の表示

イメージのプロパティを表示するには、`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があります。[118 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の値の設定

イメージプロパティの値を設定したり、プロパティ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の値のリセット

指定したプロパティ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、 <code>-F</code> または <code>--full</code> オプションを使用します。   |
| 部分   | 部分イメージは、所定の <code>dir</code> 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、 <code>-P</code> または <code>--partial</code> オプションを指定します。<br><br>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、 <code>-z</code> または <code>--zone</code> オプションを指定して、適切なバリエーションを設定します。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、 <a href="#">22 ページの「イメージとブート環境」</a> 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、 <code>-U</code> または <code>--user</code> 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。

バリエーション値、ファセット値、およびイメージプロパティ値を設定するには、`--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。

前回行なったイメージの変更または構成変更の時期をすばやく確認するには、`/var/pkg/modified` ファイルのタイムスタンプを確認します。`/var/pkg/modified` ファイルは、イメージを変更する操作が発生するたびにタイムスタンプが更新される空のファイルです。変更された内容の詳細を調べるには、[例25「イメージ変更に関する情報の取得」](#)に示すように `pkg history` コマンドに `-t` オプションを指定して使用してください。

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。

-1

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にある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 25 イメージ変更に関する情報の取得

次のコマンドは /var/pkg/modified ファイルを使用して、このイメージが前回変更された時期を表示します。

```
$ ls -l /var/pkg/modified
-rw-r--r-- 1 root root 0 Jul 22 10:54 /var/pkg/modified
```

これ以上の詳細な情報が不要な場合もあります。知る必要があることは、前回特定の操作を実行してからイメージが変更されたかどうかということだけの場合もあります。

次の例は、pkg history コマンドを使用して詳細な情報を取得する方法を示しています。

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 26**                   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

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。`pkg:/entire` パッケージの更新が必要な場合もあります。

次のコマンドは、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を表示します。

```
$ pkg list -v entire
```

- パッケージパブリッシャーの起点を確認します。パブリッシャーの起点を追加または削除することが必要な場合もあります。

次のコマンドは、このイメージに構成されている有効化されたパブリッシャーを一覧表示します。

```
$ pkg publisher -n
```

キャッシュされたコピーでなく、構成済みのパブリッシャーの場所の現在の内容を表示するには、`--no-network-cache` グローバルオプションを使用します。詳細は、`pkg(1)` のマニュアルページを参照してください。

特定のパブリッシャーの場所が現在有効化されているかどうかについては、117 ページの「パブリッシャーの起点の有効化および無効化」を参照してください。

- いずれかの `version-lock` ファセットが `false` に設定されているかどうか確認します。詳細は、130 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 facet`

- いずれかのパッケージが特定のバージョンで凍結されているかどうかを確認します。詳細は、129 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 freeze`

- IDR がインストールされているかどうかを確認します。IDR を拒否することが必要な場合もあります。置換用 IDR のインストールおよび IDR の削除については、102 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。

次のコマンドは、インストール済みのすべての IDR を一覧表示します。

`$ pkg list '*idr*'`

- 別のパブリッシャーからインストールされたパッケージが更新またはインストールを妨げていないか確認します。

次のコマンドは、`internalpublisher` パブリッシャーからインストールされたすべてのパッケージを一覧表示します。

`$ pkg list -v '*internalpublisher*'`

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。次のコマンドは、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表示します。82 ページの「使用可能なバージョンのチェック」も参照してください。

`$ pkg list -af name-of-package-to-install`

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。84 ページの「更新操作のプレビュー」も参照してください。
- 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 11.3 から Oracle Solaris 11.4 に更新する場合は、インストールされている Oracle Solaris 11.3 パッケージと目的の Oracle Solaris 11.4 パッケージの両方へのアクセスが `solaris` パブリッシャーに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。『Oracle Solaris 12 パッケージリポジトリのコピーと作成』の「Best Practices for Creating and Using Local IPS Package Repositories」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、18 ページの「障害管理リソース識別子」および『Oracle Solaris 12 での Image Packaging System を使用したソフトウェアのパッケージ化と配布』の「Package Version」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、16 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、167 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、168 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。

サイトで外部ロケーション用のプロキシが必要な場合、`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、[121 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、[117 ページの「パブリッシャーの起点の有効化および無効化」](#)に示すようにアクセスできない起点を `pkg set-publisher -dg` で無効化するか、`pkg set-publisher -d` 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、指定されたり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、指定されたり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。『[Oracle Solaris 12 パッケージリポジトリのコピーと作成](#)』の「[Best Practices for Creating and Using Local IPS Package Repositories](#)」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。

- パッケージがバリエーション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、[167 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

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。 [149 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、[164 ページの「場所が見つからない」](#) 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に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、[121 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。
- 自己署名証明書を使用中の場合、『[Oracle Solaris 12 パッケージリポジトリのコピーと作成](#)』の「[Creating a Self-Signed Server Certificate Authority](#)」に示すように 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、[160 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。

## 場所が見つからない

エラーメッセージ: http protocol error: code: 404 reason: Not Found

160 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、『Oracle Solaris 12 パッケージリポジトリのコピーと作成』の第 5 章、「Running the Package Depot Server Behind a Web Server」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(8)` 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、72 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、[121 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、`port` プロパティ、およびその他のプロパティ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、[167 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、[16 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時 (たとえば更新される時)、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。

[179 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実行します。`-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。[130 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 27 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@0.5.11,5.11-0.175.2.0.0.9.0:20130304T213946Z
  Reason: All versions matching 'incorporate' dependency pkg:/runtime/java/jre-7@1.7.0.17,5.11-0.175.2.0.0.9.0 are rejected
  Reject: pkg://solaris/runtime/java/jre-7@1.7.0.17,5.11-0.175.2.0.0.9.0:20130304T214022Z
  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、[130 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があります。

```
$ 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、[16 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 28** 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://` の後に示されません。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が、`pkg freeze` および `pkg unfreeze` コマンドについては、[129 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。

[177 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。

[179 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。

[177 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が。

```
pkg: The image cannot be modified as it is currently in use by another package
client: pkg on cbusl04061, 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。

## ファイルが回収された

次の情報メッセージの後に、回収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(7) 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場合があります。

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(8) 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。『[Oracle Solaris 12 パッケージリポジトリのコピーと作成](#)』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 Oracle VM Server 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、『[Oracle Solaris 12 パッケージリポジトリのコピーと作成](#)』の「[Configuring a Simple Prefixed Proxy](#)」の説明に従って、`pkg/threads` プロパティを設定します。

# 索引

---

## あ

アクセス権, 24  
アプリケーションバージョンメディエーション,  
133  
依存関係  
    group, 17  
    incorporate, 16  
イメージ, 22  
    アップグレード, 79  
    オプションのコンポーネント, 123  
    再インストール, 70  
    作成, 148  
    相互に排他的なコンポーネント, 123  
    プロパティ, 140  
        BE 作成ポリシー, 141  
        キャッシュのフラッシュ, 181  
        署名付きパッケージの使用, 143  
        追加と削除, 144  
        バリエーション, 123  
        ファセット, 123  
    ポリシー, 140  
インストールの制約, 16, 167  
親イメージ, 22

## か

カーネルゾーン, 22  
回避リスト, 138  
    削除, 55, 70  
    追加, 59, 70  
起点リポジトリ, 22  
キャッシュ  
    フラッシュ, 181  
共通脆弱性 (CVE) 参照 CVE  
クリティカルパッチ更新 (CPU) 参照 CPU

グループパッケージ, 17, 36, 44, 138  
検索インデックス, 25, 55  
権利プロファイル, 24  
子イメージ, 22  
更新の制約, 167  
    entire 制約パッケージ, 16

## さ

サーフェス, 16, 167  
サポートリポジトリ, 96  
サポートリポジトリ更新 (SRU) 参照 SRU  
システムリポジトリ, 74  
障害管理リソース識別子 (FMRI), 18  
承認, 24  
制約, 16, 167  
    緩和, 130  
制約パッケージ, 16, 130, 132, 155, 167, 168  
    カスタム, 89  
セキュリティ  
    権利, 24  
    パッケージ署名の検証, 143  
    パッケージの検証, 61  
ゾーン, 52, 72  
    Oracle Solaris 10 ゾーン, 22  
    カーネルゾーン, 22  
    大域ゾーン, 22  
    非大域ゾーン, 22  
ゾーンプロキシサービス  
    zones-proxy-client, 74  
    zones-proxyd, 74  
ソフトウェアリポジトリ, 15

## た

大域ゾーン, 22, 52, 72  
特権, 24

## は

バージョン制約, 16, 167  
    緩和, 130  
バージョン文字列コンポーネント, 18  
パッケージ, 16  
    CPU, 99  
    FMRI, 18, 56  
    IDR, 102  
    pkg:/entire, 155, 167  
    SMF サービスアクチュエータ, 53  
    version-lock. ファセット, 130  
    アンインストール, 67  
    依存関係, 44  
    一覧表示, 25  
    イメージのクリーンアップ, 70  
    インストール, 55  
    インストール可能, 56  
    インストール可能な, 26  
    インストールされたファイルの修正, 65  
    インストールされていないパッケージの無視, 60  
    インストールされていないパッケージを無視, 69  
    インストール済みの, 26  
    インストール済みパッケージの検証 参照 pkg verify コマンド  
    インストール済みパッケージの修正, 61  
    インストールソース, 57  
    インストール中の拒否, 59, 174  
    オプションのコンポーネント, 23, 123  
    親の依存関係, 72  
    回避リスト, 138  
        削除, 55, 70  
        追加, 59, 70  
    カスタム制約パッケージ, 89  
    完全な FMRI, 30  
    クリティカルパッチ更新, 99  
    グループ, 17, 36, 44, 138  
    検索, 38  
    公開, 16  
    更新が利用可能な, 27  
    更新またはアップグレード, 60  
    最新の, 26  
    削除, 67, 67  
    作成, 16  
    識別子, 18  
    使用可能なすべての, 27  
    署名プロパティ, 143  
    すべて更新, 79  
    制約, 16, 167  
    説明, 29  
    相互に排他的なコンポーネント, 23, 123  
    ダウングレード, 61  
    提供されたサービス, 42  
    提供されたバグ修正, 43  
    提供されたファイル, 41  
    提供されたユーザー, 42  
    凍結, 29, 71, 129, 158, 177  
    名前, 18  
    名前の変更, 28  
    バージョンが制約された, 16, 129, 158, 167, 177  
    廃止, 28  
    パブリッシャー, 21, 56  
    バリエーション, 23, 123  
    ファイルシステムの内容, 32  
    ファセット, 23, 123  
    ライセンス, 31, 37, 54  
    リポジトリ, 21  
    ロック解除, 132, 168  
    パッケージアーカイブ, 21, 57, 102  
    パッケージ解除されたファイルシステムオブジェクト, 180  
    パッケージ解除されたファイルまたは編集可能ファイルの回収, 180  
    パッケージに関する情報の更新, 25, 55  
    パッケージのアンインストール, 67  
    パッケージの削除, 67, 67  
    パッケージパブリッシャー 参照 パブリッシャー  
    パッケージリポジトリ 参照 リポジトリ  
    パッケージをバージョン制約からロック解除, 132, 168  
    パフォーマンス, 182  
    パブリッシャー, 21, 113  
        起点, 114  
        検索順序, 26, 118, 158



構成, 115  
 プロキシ, 121  
 プロパティ, 140  
   固定, 118, 158  
   署名付きパッケージの使用, 144  
   追加と削除, 144  
 バリエーション, 23, 123  
 非大域ゾーン, 22, 52, 72  
   更新, 76  
   システムリポジトリサービス, 180  
   パッケージのインストール, 72  
   パッケージパブリッシャー, 74, 161, 180  
 ファームウェアの更新, 100  
 ファイアウォール, 160, 163  
 ファイル  
   編集可能な preserve または overlay, 50  
 ファセット, 23, 123, 130  
 ブート環境 (BE), 22  
 プロキシ  
   SMF サービスプロパティ, 122  
   環境変数, 122  
   パッケージパブリッシャー, 121, 156, 156, 163  
 プロキシサービス, 74  
 プロパティ, 140  
   BE 作成ポリシー, 141  
   キャッシュのフラッシュ, 181  
   署名付きパッケージの使用, 143  
   設定, 147  
   追加と削除, 144  
   表示, 147  
 編集可能ファイル, 180

## ま

ミラーリポジトリ, 22  
 メディエーション, 133  
   リンク, 35  
 メディエーター, 133

## や

役割, 24

## ら

リポジトリ, 15, 21, 57  
   起点, 22  
   内容の更新, 25  
   ミラー, 22  
 リンクされたイメージ, 22, 52, 76  
   pkg uninstall, 69  
   pkg update, 60

## B

BE, 22  
   pkg コマンドオプション, 51  
   アクティブ化, 51  
   イメージポリシープロパティ設定, 141  
   名前の指定, 51  
   パッケージのインストール中の作成, 58  
   必要, 51  
 beadm コマンド, 59

## C

ca-certificates サービス, 163  
 ca-certificates パッケージ, 163  
 CPU, 96  
   パッケージ, 99  
 critical-patch-update/solaris-11-cpu パッケージ, 99  
 crypto/ca-certificates パッケージ, 163  
 CVE, 96, 99

## E

Enterprise Manager Ops Center, 160  
 entire 制約パッケージ, 17

## F

flush-content-cache-on-success イメージプロパティ, 181  
 FMRI, 18

**G**

group 依存関係, 17

**I**

IDR, 21, 96, 102

インストール, 105

更新, 108

置換, 108

IDR (Interim Diagnostic or Relief) 更新 参照 IDR

IMAGE\_META ディレクトリ, 180

incorporate 依存関係, 16

incorporation 参照 制約パッケージ

**L**

lost+found ディレクトリ, 180

**O**

Ops Center, 160

Oracle Solaris 10 ゾーン, 22

**P**

pkg:/entire パッケージ, 155, 167

pkg.sysrepo コマンド, 74

pkg コマンド

プレビュー, 48

履歴の表示, 149

pkg avoid コマンド, 138

pkg change-facet コマンド, 126, 130

pkg change-variant コマンド, 124

pkg contents コマンド, 32

-t オプション, 33

pkg search との比較, 38

pkg exact-install コマンド, 70

pkg facet コマンド, 126, 158

pkg fix コマンド, 62

pkg revert との比較, 61

pkg freeze コマンド, 129, 158, 177

pkg history コマンド, 149

pkg image-create コマンド, 148

pkg info コマンド, 29

pkg list との比較, 29

pkg install コマンド, 56

--be-name オプション, 58

--reject オプション, 59, 174

pkg update との比較, 50

pkg list コマンド, 25, 56

pkg info との比較, 29

pkg mediator コマンド, 133

pkg publisher コマンド, 114

pkg purge-history コマンド, 149

pkg refresh コマンド, 25, 55

pkg revert コマンド, 65

--tagged オプション, 66

参照 revert-tag 属性

pkg fix との比較, 61

pkg search コマンド, 38

OR キーワードの使用, 42

pkg contents との比較, 38

クエリー, 39

pkg set-mediator コマンド, 137

pkg set-publisher コマンド, 115

proxy オプション, 156

鍵および証明書のオプション, 162

プロキシオプション, 121, 156, 163

プロパティーの追加と削除, 144

pkg unavoid コマンド, 138

pkg unfreeze コマンド, 129, 177

pkg uninstall コマンド, 67

pkg unset-mediator コマンド, 135

pkg update コマンド

pkg install との比較, 50

イメージのアップグレード, 79

パッケージの更新またはアップグレード, 60

パッケージのダウングレード, 61

pkg variant コマンド, 124

pkg verify コマンド, 61

-p オプション, 64, 65

--parsable オプション, 62

--unpackaged-only オプション, 65

--unpackaged オプション, 65

インストール済みパッケージ, 62

ファイルシステムの内容, 64

pkg コマンドのプレビュー, 48

pkgrepo コマンド, 90

**R**

revert-tag 属性  
  system:clone, 66  
  system:dev-init, 66  
  system:sysconfig-profile, 66

**S**

SMF サービス, 53  
SMF プロキシサービス, 74  
solaris-11-cpu クリティカルパッチ更新パッケージ, 99  
solaris-kz ブランドゾーン, 22  
solaris-minimal-server グループのインストールパッケージ, 36, 44  
solaris10 ブランドゾーン, 22  
solaris ブランドゾーン, 22  
SRU, 20, 96, 102  
SSL 証明書, 161  
  自己署名, 163  
support/critical-patch-update/solaris-11-cpu パッケージ, 99  
system:clone, 66  
system:dev-init, 66  
system:sysconfig-profile, 66

**U**

URI (Universal Resource Identifier), 21

**V**

/var/pkg/modified ファイル, 149  
/var/pkg ディレクトリ, 180, 181

**Z**

ZFS ストレージプール容量, 182

