Adding and Updating Oracle® Solaris 11.1 Software Packages

Part No: E28984–02
February 2013

130304@25097

Contents

Preface

Adding and Updating Oracle Solaris 11.1 Software Packages describes the software installation functions of the Oracle Solaris Image Packaging System (IPS) feature. IPS commands enable you to list, search, install, update, and remove software packages for the Oracle Solaris 11 operating system. A single IPS command can update your image to a new operating system release. IPS commands enable you to restrict which packages can be installed or which versions of packages can be installed.

IPS commands also enable you to copy and create IPS package repositories, and create IPS packages. See "Related Documentation" on page 8 for information about those tools.

To use IPS, you must be running the Oracle Solaris 11 OS. To install the Oracle Solaris 11 OS, see *Installing Oracle Solaris 11.1 Systems*.

Who Should Use This Book

This book is for system administrators who install and manage software and manage system images.

How This Book Is Organized

- Chapter 1, "Introduction to the Image Packaging System," describes the Image Packaging System and components such as packages, publishers, and repositories.
- Chapter 2, "IPS Graphical User Interfaces," explains how to use Package Manager and Update Manager, including how to use Web Install.
- Chapter 3, "Getting Information About Software Packages," shows how to search for packages and display information about packages.
- Chapter 4, "Installing and Updating Software Packages," shows how to install, update, and uninstall packages.
- Chapter 5, "Configuring Installed Images," shows how to configure characteristics that apply to an entire image, such as configuring package publishers or restricting which packages can be installed.

Related Documentation

In addition to these books, see the Package Manager online help and the `pkg`(1M) and `beadm`(1M) man pages.

- *Copying and Creating Oracle Solaris 11.1 Package Repositories*
- *Creating and Administering Oracle Solaris 11.1 Boot Environments*
- *Installing Oracle Solaris 11.1 Systems*
- *Packaging and Delivering Software With the Image Packaging System in Oracle Solaris 11.1*

Access to Oracle Support

Oracle customers have access to electronic support through My Oracle Support. For information, visit `http://www.oracle.com/pls/topic/lookup?ctx=acc&id=info` or visit `http://www.oracle.com/pls/topic/lookup?ctx=acc&id=trs` if you are hearing impaired.

Typographic Conventions

The following table describes the typographic conventions that are used in this book.

TABLE P–1 Typographic Conventions

Typeface	Description	Example
`AaBbCc123`	The names of commands, files, and directories, and onscreen computer output	Edit your `.login` file. Use `ls -a` to list all files. `machine_name% you have mail.`

`AaBbCc123`	What you type, contrasted with onscreen computer output	`machine_name%` **`su`**
		`Password:`

aabbcc123	Placeholder: replace with a real name or value	The command to remove a file is `rm` *filename*.
AaBbCc123	Book titles, new terms, and terms to be emphasized	Read Chapter 6 in the *User's Guide*. A *cache* is a copy that is stored locally. Do *not* save the file. **Note:** Some emphasized items appear bold online.

Shell Prompts in Command Examples

The following table shows UNIX system prompts and superuser prompts for shells that are included in the Oracle Solaris OS. In command examples, the shell prompt indicates whether the command should be executed by a regular user or a user with privileges.

TABLE P–2 Shell Prompts

Shell	Prompt
Bash shell, Korn shell, and Bourne shell	`$`
Bash shell, Korn shell, and Bourne shell for superuser	`#`
C shell	`machine_name%`
C shell for superuser	`machine_name#`

◆ ◆ ◆ CHAPTER 1

Introduction to the Image Packaging System

The Oracle Solaris Image Packaging System (IPS) is a framework that enables you to list, search, install, update, and remove software packages for the Oracle Solaris 11 operating system. A single IPS command can update your image to a new operating system release.

Image Packaging System

Oracle Solaris 11 software is distributed in IPS packages. IPS packages are stored in IPS package repositories, which are populated by IPS publishers. IPS packages are installed into Oracle Solaris 11 images. A subset of the capabilities that are available through the IPS command-line interface is available through the Package Manager graphical user interface.

IPS tools provide the following capabilities. See "IPS Concepts" on page 12 for definitions of terms such as publisher and repository.

- List, search, install, restrict installation, update, and remove software packages.
- List, add, and remove package publishers. Change publisher attributes such as search priority and stickiness. Set publisher properties such as signature policy.
- Update an image to a new operating system release.
- Create copies of existing IPS package repositories. Create new package repositories.
- Create and publish packages.
- Create boot environments.

To use IPS, you must be running the Oracle Solaris 11 OS. To install the Oracle Solaris 11 OS, see *Installing Oracle Solaris 11.1 Systems*.

IPS Concepts

This section defines terms and concepts that are used in the remainder of this guide.

IPS Packages

An IPS *package* is defined by a text file called a *manifest*. A package manifest describes package *actions* in a defined format of key/value pairs and possibly a data payload. Package actions include files, directories, links, drivers, dependencies, groups, users, and license information. Package actions represent the installable objects of a package. Actions called `set` actions define package metadata such as classification, summary, and description.

You can search for packages by specifying package actions and action keys. See `pkg(5)` for descriptions of package actions.

Group and incorporation packages do not deliver content such as files, but they help you install sets of related packages.

An *incorporation* is a package that constrains the versions of a specified set of packages. For example, if a package in an installed incorporation is version 1.4.3, then no version less than 1.4.3 or greater than or equal to 1.4.4 can be installed. However, versions that merely extend the dotted sequence, such as 1.4.3.7, could be installed. Incorporations force the incorporated packages to upgrade synchronously. An incorporated package could be removed, but if the package is installed or updated, the version is constrained. See "Relaxing Version Constraints Specified by Incorporations" on page 63 for related information.

The package named `entire` is a special incorporation package that constrains the versions of other incorporation packages.

Caution – Do not remove the package named `entire`. The `entire` package constrains system package versions so that the resulting set of packages is a supportable image. Proper system update and correct package selection depend on this incorporation. Removing the `entire` package will result in an unsupported system.

A *group* package specifies the set of packages that constitute a feature or tool. Packages specified in a group package do not specify the package version. The group package is a content management tool, not a version management tool.

Group package manifests specify `group` dependencies. Group packages deliver the packages named in those `group` dependencies unless those packages are on the avoid list. See "Avoiding Installing Some Packages in a Group Package" on page 66 for information about the avoid list of an image.

The `group/feature/amp` package, for example, delivers the Apache web server, the MySQL database, and PHP. The `group/system/solaris-desktop` package delivers a set of packages suitable for a desktop system. The `group/system/solaris-large-server` package does not deliver desktop packages such as media tools and windowing themes. See "Listing All Installable Packages in a Group Package" on page 30 for an example of how to list of all the packages that are delivered by a group package.

Fault Management Resource Identifiers

Each package is represented by a Fault Management Resource Identifier (FMRI). The full FMRI for a package consists of the scheme, a publisher, the package name, and a version string in the following format. The scheme, publisher, and version string are optional. When using IPS commands, you can use the smallest portion of the package name that uniquely identifies the package.

FMRI format:

scheme`://`*publisher*`/`*package_name*`@`*version*`:`*date*`T`*time*`Z`

FMRI example:

```
pkg://solaris/driver/network/ethernet/bge@0.5.11,5.11-0.175.1.0.0.21.0:20120723T161616Z
```

Scheme	`pkg`
Publisher	`solaris` If the publisher is specified, then the publisher name must be preceded by `pkg://` or `//`.
Package name	`driver/network/ethernet/bge` Package names are hierarchical with an arbitrary number of components separated by forward slash (/) characters. In IPS commands, leading components of package names can be omitted if the package name that is used in the command uniquely identifies the package. If you specify the full package name but omit the publisher, the full package name can be preceded by `pkg:/` or `/` but not by `pkg://` or `//`. If you specify an abbreviated package name, do not use any other characters to the left of the package name.
Version	The package version has four parts: Component version: `0.5.11` For components tightly bound to the operating system, the component version usually includes the value of `uname -r` for that version of the operating system. For a component with its own development lifecycle, the component version is a dotted release number, such as 2.4.10.

Build version: `5.11`
: The build version must follow a comma (,). The build version specifies the version of the operating system on which the contents of the package were built.

Branch version: `0.175.1.0.0.21.0`
: The branch version must follow a hyphen (`-`). The branch version provides vendor-specific information.

  Oracle Solaris packages show the following information in the branch version portion of the version string of a package FMRI:

  Major release number: `0.175`
  : The major or marketing development release build number. In this example, `0.175` indicates Oracle Solaris 11.

  Update release number: `1`
  : The update release number for this Oracle Solaris release. The update value is 0 for the first customer shipment of an Oracle Solaris release, 1 for the first update of that release, 2 for the second update of that release, and so forth. In this example, `1` indicates Oracle Solaris 11.1.

  SRU number: `0`
  : The Support Repository Update (SRU) number for this update release. SRUs include only bug fixes; they do not include new features. The Oracle Support Repository is available only to systems under a support contract.

  Reserved: `0`
  : This field is not currently used for Oracle Solaris packages.

  SRU build number: `21`
  : The build number of the SRU, or the respin number for the major release.

  Nightly build number: `0`
  : The build number for the individual nightly builds.

Time stamp: `20110921T002716Z`
: The time stamp must follow a colon (:). The time stamp is the time the package was published in ISO-8601 basic format: *YYYYMMDD*`T`*HHMMSS*`Z`.

Publishers, Repositories, and Package Archives

A *publisher* identifies a person or organization that provides one or more packages. Publishers can distribute their packages using package repositories or package archives. Publishers can be configured into a preferred search order. When a package installation command is given and the package specification does not include the publisher name, the first publisher in the search order is searched for that package. If a match of the specified package FMRI pattern is not found, the second publisher in the search order is searched, and so forth until the package is found or all publishers have been searched.

A *repository* is a location where packages are published and from where packages are retrieved. The location is specified by a Universal Resource Identifier (URI). A *catalog* is the list of all the packages in a repository.

A *package archive* is a file that contains publisher information and one or more packages provided by that publisher.

Repository Origins and Mirrors

An *origin* is a package repository that contains both package *metadata* (such as catalogs, manifests, and search indexes) and package *content* (files). If multiple origins are configured for a given publisher in an image, the IPS client attempts to choose the best origin from which to retrieve package data.

A *mirror* is a package repository that contains only package content. IPS clients access the origin to obtain a publisher's catalog, even when the clients download package content from a mirror. If a mirror is configured for a publisher, the IPS client prefers the mirror for package content retrieval. If multiple mirrors are configured for a given publisher in an image, the IPS client attempts to choose the best mirror from which to retrieve package content. If all mirrors are unreachable, do not have the required content, or are slower, the IPS client retrieves the content from an origin.

Images and Boot Environments

An *image* is a location where IPS packages can be installed and where other IPS operations can be performed.

A *boot environment* (BE) is bootable instance of an image. You can maintain multiple BEs on your system, and each BE can have different software versions installed. When you boot your system, you have the option to boot into any of the BEs on the system. A new BE can be created automatically as a result of package operations. You can also explicitly create a new BE. Whether a new BE is created depends on image policy as described in "Boot Environment Policy Image Properties" on page 72.

Package Facets and Variants

Software can have components that are optional and components that are mutually exclusive. Examples of optional components include locales and documentation. Examples of mutually exclusive components include SPARC or x86 and debug or non-debug binaries. In IPS, optional components are called *facets* and mutually exclusive components are called *variants*.

Facets and variants are special properties set on the image and are tags set on actions within a package. Most variant tags can have various values. Facet tags set on an action can only have the value `true`. The values of facet and variant tags on an action compared with the values of facets and variants set in the image determine whether that package action can be installed. For example, if you set a particular locale facet to `false` in the image, any file actions that specify that facet will not be installed, and currently installed file actions that specify that facet are uninstalled.

The following algorithm describes how the facets and variants set on the image affect whether a particular action is installed.

- Actions with no facet or variant tags are always installed.
- Actions with facet tags are installed unless all of the facets or facet patterns matching the tags are set to `false` on the image. If any facet is set to `true` or is not explicitly set (`true` is the default), then the action is installed.
- Actions with variant tags are installed only if the values of all the variant tags are the same as the values set in the image.
- Actions with both facet and variant tags are installed if both the facets and the variants allow the action to be installed.

To view or modify the values of the facets and variants set on the image, see "Controlling Installation of Optional Components" on page 57.

Installation Privileges

The commands discussed in Chapter 3, "Getting Information About Software Packages," do not require any special privilege to use. Tasks such as installing and updating IPS packages, setting publishers, and modifying images require more privilege. Use one of the following methods to gain more privilege.

Rights profiles Use the `profiles` command to list the rights profiles that are assigned to you. If you have the Software Installation rights profile, you can use the `pfexec` command to install and update packages and manage boot environments.

```
$ pfexec pkg install editor/gnu-emacs
$ pfexec beadm activate solaris11_1-2
```

Roles	Use the `roles` command to list the roles that are assigned to you. If you have the `root` role, you can use the `su` command with the `root` password to assume the `root` role.
sudo command	Depending on the security policy at your site, you might be able to use the `sudo` command with your user password to execute a privileged command.

◆ ◆ ◆ CHAPTER 2

IPS Graphical User Interfaces

IPS includes two Graphical User Interface (GUI) tools.

- Package Manager provides most package and publisher operations and some boot environment (BE) operations. If you are new to the Oracle Solaris OS and IPS technologies, you can use Package Manager to quickly identify and install packages.
- Update Manager updates all packages in the image that have updates available.

Using Package Manager

Package Manager provides a subset of the tasks that can be performed from the command line:

- List, search, install, update, and remove packages
- Add and configure package sources
- Activate, rename, and remove BEs

Start Package Manager in one of the following ways:

Tool bar	Click the Package Manager icon in the tool bar. The Package Manager icon is a box with a circling arrow.
Desktop icon	Double-click the Package Manager icon on the desktop.
Menu bar	Select System>Administration>Package Manager.
Command line	`$ pfexec packagemanager &`

For complete Package Manager documentation, select Help>Contents from the Package Manager menu bar.

Package Manager Command Line Options

The following options are supported for the `packagemanager(1)` command.

TABLE 2–1 Package Manager Command Options

Option	Description
`--image-dir` or `-R` *dir*	Operate on the image rooted at *dir*. The default behavior is to operate on the current image. The following command operates on the image stored at `/aux0/example_root`: `# packagemanager -R /aux0/example_root`
`--update-all` or `-U`	Update all installed packages that have updates available. Specifying this option is the same as selecting the Updates option in the Package Manager GUI. See "Using Update Manager" on page 22 for more information about updating all packages.
`--info-install` or `-i` *file*`.p5i`	Specify a `.p5i` file to run Package Manager in Web Install mode. The specified file must have the extension `.p5i`. See "Using Web Install" on page 20 for more information.
`--help` or `-h`	Display command usage information.

Using Web Install

See the Package Manager Help for detailed information about the Web Install process.

Package Manager supports installing packages using a simple one-click Web Install process. The Web Install process uses a `.p5i` file. A `.p5i` file contains information to add publishers and add packages that can be installed from these publishers. The information in the `.p5i` file is read and used by the Web Install process.

Exporting Files Using Web Install

If you want other users to be able to install packages that you have installed on your system, you can export the installation instructions for those package files using the Web Install process. The Web Install process creates a `.p5i` file that consists of installation instructions for those packages and publishers to be installed.

To export the installation instructions for your selected packages and their publishers to a `.p5i` file, perform the following steps:

1. From the Package Manager Publisher drop-down menu, select the publisher from which you want to include the packages in the `.p5i` file.
2. In the Package Manager package list pane, select the package whose installation instructions you want to distribute.
3. Select File>Export Selections to display the Export Selections Confirmation window.
4. Click the OK button to confirm the selections. The Export Selections window is displayed.

5. A default name for the `.p5i` file is provided. You can change this file name, but do not change the `.p5i` extension.
6. A default location for the `.p5i` file is provided. You can change the location.
7. Click the Save button to save the file name and location.

Using Web Install to Add Publishers and Install Packages

The Web Install process enables you to install packages through a `.p5i` file. This file might be on your desktop or on a web site.

1. Use one of the following methods to start Package Manager in Web Install mode:
   - Select a `.p5i` file on your desktop.
   - Start Package Manager from the command line and specify a `.p5i` file:

     ```
     # packagemanager ./wifile.p5i
     ```
   - Go to a URL location that contains a link to a `.p5i` file.

     If the `.p5i` file is located on a web server that has registered this MIME type, just click the link to the `.p5i` file.

     If the `.p5i` file is located on a web server that has not registered this MIME type, save the `.p5i` file to your desktop and then select it.
2. The Install/Update window is displayed. The label at the top of the window is: "Package Manager Web Installer/The following will be added to your system." The publishers and packages to be installed are listed. Click the Proceed button to continue with the installation.
3. If the specified package publisher is not already configured on your system, the Add Publisher window is displayed. The name and URI of the publisher are already entered.

   If the publishers to be added are secure publishers, an SSL key and certificate are required. Browse to locate the SSL Key and SSL Certificate on your system.

   If the publisher is added successfully, the Adding Publisher Complete dialog displays. Click the OK button to continue with the installation.
4. If a `.p5i` file contains packages from a disabled publisher, Web Install opens an Enable Publisher dialog. Use this dialog to enable the publisher so that you can install the packages.

The Install/Update window now looks the same as when you select the Package Manager Install/Update option.

The application closes when all packages are installed.

Using Update Manager

Update Manager updates all installed packages to the newest version allowed by the constraints imposed on the system by installed package dependencies and publisher configuration. This function is the same as the following functions:

- In the Package Manager GUI, select the Updates button or the Package>Updates menu option.
- Use the `packagemanager` command.

  ```
  $ pfexec packagemanager --update-all
  ```
- Use the `pkg` command.

  ```
  $ pfexec pkg update
  ```

Start Update Manager in one of the following ways:

Status bar	When updates are available, you should see a notification in the status bar. Click where indicated in the notification. The Update Manager icon is a stack of three boxes.
Menu bar	Select System>Administration>Update Manager.
Command line	`# pm-updatemanager`
Automated	The Update Manager package, `package/pkg/update-manager`, delivers the `cron` job `/usr/lib/update-manager/update-refresh.sh`. When the SMF service `svc:/application/pkg/update` is online, this `cron` job checks periodically for updated packages available from configured publishers (the first two steps of the following process). If updated packages are available, you receive a notification in your desktop tool bar. Select the notification icon to open the Update Manager GUI.

The Updates window displays, and the update process starts:

1. The system refreshes all catalogs.
2. The system evaluates all installed packages to determine which packages have updates available.
   - If no packages have updates available, the message “No Updates Available” is displayed and processing stops.
   - If package updates are available, the packages to be updated are listed for your review. This is your last chance to click the Cancel button to abort the update.
3. Click the Proceed button to continue with the update. The system downloads and installs all package updates.

   The following packages are updated first if they have updates available. Then any other packages are updated.

```
package/pkg
package/pkg/package-manager
package/pkg/update-manager
```

By default, each package is updated from the publisher from which it was originally installed. If the original publisher is non-sticky, then a newer version of the package that is compatible with this image could be installed from another publisher. Use the Package Manager Manage Publishers window or the `pkg set-publisher` command to set a publisher as sticky or non-sticky.

A new BE might be created, depending on which packages are updated and depending on your image policy.

If an error occurs at any time during the update process, the Details panel expands and the details of the error are displayed. An error status indicator is shown next to the failed stage.

4. If the system created a new BE for the update, you can edit the default BE name. When you are satisfied with the BE name, click the Restart Now button to restart your system immediately. Click the Restart Later button to restart your system at a later time. You must restart to boot into the new BE. The new BE will be your default boot choice. Your current BE will be available as an alternate boot choice.

Update Manager Command Line Options

The following options are supported for the `pm-updatemanager(1)` command.

TABLE 2–2 Update Manager Command Options

Option	Description
`--image-dir` or `-R` *dir*	Operate on the image rooted at *dir*. The default behavior is to operate on the current image. The following command updates the image at `/aux0/example_root`: **`# pm-updatemanager -R /aux0/example_root`**
`--help` or `-h`	Display command usage information.

◆ ◆ ◆ CHAPTER 3

Getting Information About Software Packages

This chapter describes commands that give you the following kinds of information about packages:

- Whether the package is installed or can be updated
- The description, size, and version of the package
- Which packages are part of a group package
- Which packages are in a particular category
- Which package delivers a specified file

No special privileges are needed to run any of these commands.

Showing Package Install State Information

The `pkg list` command tells you whether a package is installed in the current image and whether an update is available. With no options or operands, this command lists all packages that are installed in the current image. To narrow your results, provide one or more package names. You can use wildcards in the package names. Quote the wildcards so that the argument is passed directly to `pkg` and the shell does not expand it. Package variants for an architecture or zone type that does not match this image are not listed.

```
/usr/bin/pkg list [-Hafnsuv] [-g path_or_uri ...] [--no-refresh]
    [pkg_fmri_pattern ...]
```

The `pkg list` command displays one line of information for each package.

```
$ pkg list '*toolkit'
NAME (PUBLISHER)                  VERSION                   IFO
isvtoolkit (isvpub)               1.0                       i--
system/dtrace/dtrace-toolkit      0.99-0.175.1.0.0.21.0     i--
```

The publisher name in parentheses indicates that the `isvpub` publisher is not the first publisher in the publisher search order in this image. The `dtrace-toolkit` package that is installed in this image is published by the publisher that is the first publisher in the search order.

The "i" in the I column indicates that these packages are installed in this image. To list packages that are installed and the newest versions of packages that are not installed but could be installed in this image, use the `-a` option.

```
$ pkg list -a '*toolkit'
NAME (PUBLISHER)                  VERSION                   IFO
image/nvidia/cg-toolkit           3.0.15-0.175.1.0.0.14.0   ---
isvtoolkit (isvpub)               1.0                       i--
system/dtrace/dtrace-toolkit      0.99-0.175.1.0.0.21.0     i--
```

This output indicates that the `image/nvidia/cg-toolkit` package can be installed in this image.

To list all matching packages, including packages that cannot be installed in this image, use the `-af` option. To list only the newest versions of these packages, specify `@latest`.

```
$ pkg list -af '*toolkit@latest'
NAME (PUBLISHER)                  VERSION                   IFO
developer/dtrace/toolkit          0.99-0.173.0.0.0.1.0      --r
image/nvidia/cg-toolkit           3.0.15-0.175.1.0.0.14.0   ---
isvtoolkit (isvpub)               1.0                       i--
system/dtrace/dtrace-toolkit      0.99-0.175.1.0.0.21.0     i--
```

This output indicates that the `developer/dtrace/toolkit` package cannot be installed in this image. The "r" in the O column indicates that this package has been renamed. The `developer/dtrace/toolkit` package has been renamed to `system/dtrace/dtrace-toolkit`, and `system/dtrace/dtrace-toolkit` is already installed.

In the following example, the `web/amp` package has been renamed to `group/feature/amp`. If you specify the command to install the `web/amp` package, the `group/feature/amp` package is installed automatically.

```
$ pkg list -a amp
NAME (PUBLISHER)                  VERSION                    IFO
group/feature/amp                 0.5.11-0.175.0.0.0.21.0    ---
web/amp                           0.5.11-0.174.0.0.0.0.0     --r
```

The `pkg list` command does not tell you the new name of a renamed package. In the previous examples, the pattern given as input to the `pkg list` command happened to match both the old and new names, and an inference could be drawn. In general, to display the new name of a renamed package, use the `pkg info` command as shown in "Displaying Package Descriptions or Licenses" on page 28.

The `-n` option lists the newest version of each known package. An "o" in the O column indicates that the package is obsolete. You cannot install a package that is obsolete.

```
$ pkg list -n '*mysql-5?'
NAME (PUBLISHER)                  VERSION                    IFO
database/mysql-50                 5.0.91-0.171               --o
database/mysql-51                 5.1.37-0.175.1.0.0.21.0    ---
```

This output indicates that the `database/mysql-50` package cannot be installed in this image. This package has not been renamed. If you specify the command to install the `mysql-50` package, the `mysql-51` package is not installed. No packages are installed in this case.

An "f" in the F column indicates the package is frozen. If a package is frozen, you can only install or update to packages that match the frozen version. See "Locking Packages to a Specified Version" on page 61 for information about freezing packages.

```
$ pkg list mercurial
NAME (PUBLISHER)                  VERSION                   IFO
developer/versioning/mercurial    2.2.1-0.175.1.0.0.21.0    if-
```

The `-s` option lists only the package name and summary.

```
$ pkg list -ns mysql-51 feature/amp
NAME (PUBLISHER)     SUMMARY
database/mysql-51    MySQL 5.1 Database Management System
group/feature/amp    AMP (Apache, MySQL, PHP) Deployment Kit for Oracle Solaris
```

The `-v` option lists the full package FMRI.

```
$ pkg list -nv mysql-51
FMRI                                                                      IFO
pkg://solaris/database/mysql-51@5.1.37,5.11-0.175.1.0.0.21.0:20120723T165236Z  ---
```

The `-u` option lists all installed packages that have newer versions available.

```
$ pkg list -u 'compress/*'
NAME (PUBLISHER)                  VERSION                   IFO
compress/bzip2                    1.0.6-0.175.1.0.0.19.0    i--
compress/gzip                     1.4-0.175.1.0.0.19.0      i--
compress/p7zip                    9.20.1-0.175.1.0.0.19.0   i--
compress/unzip                    6.0-0.175.1.0.0.19.0      i--
compress/zip                      3.0-0.175.1.0.0.19.0      i--
```

Note – The number of packages that have newer versions available in the package repository might be larger than the number of packages that could be updated in this image. Packages can only be updated to versions allowed by the constraints imposed on the image by installed package dependencies and publisher configuration. To determine what packages can be updated in this image, use `pkg update -nv`.

Use the `-g` option to specify the repository or package archive to use as the source of package data for the operation.

When you use the `--no-refresh` option, `pkg` does not attempt to contact the repositories for the image's publishers to retrieve the newest list of available packages.

Displaying Package Descriptions or Licenses

The `pkg info` command displays information about a package, including the name, installed state, version, packaging date, package size, and the full FMRI. With no options or operands, this command displays information about all packages that are installed in the current image. To narrow your results, provide one or more package names. You can use wildcards in the package names. Quote the wildcards so that the argument is passed directly to `pkg` and the shell does not expand it.

`/usr/bin/pkg info [-lr] [-g` *path_or_uri* `...] [--license] [`*pkg_fmri_pattern* `...]`

Both the `info` and `list` subcommands display the package name, publisher, and version information. The `pkg list` command shows whether an update exists for the package, whether an update can be installed in this image, and whether a package is obsolete or renamed. The `pkg info` command displays the package summary, description, category, and size, and can separately display the license information.

The `-r` option displays the newest available versions, retrieving information for any packages not currently installed from the repositories of the configured publishers.

```
$ pkg info -r group/feature/amp
          Name: group/feature/amp
       Summary: AMP (Apache, MySQL, PHP) Deployment Kit for Oracle Solaris
   Description: Provides a set of components for deployment of an AMP (Apache,
                MySQL, PHP) stack on Oracle Solaris
      Category: Meta Packages/Group Packages (org.opensolaris.category.2008)
                Web Services/Application and Web Servers (org.opensolaris.category.2008)
         State: Not installed
     Publisher: solaris
       Version: 0.5.11
 Build Release: 5.11
        Branch: 0.175.1.0.0.21.0
Packaging Date: July 23, 2012 06:20:57 PM
          Size: 5.46 kB
          FMRI: pkg://solaris/group/feature/amp@0.5.11,5.11-0.175.1.0.0.21.020120723T182057Z
```

Use the `pkg info` command to find the new name of a renamed package. The following example shows that the new name of the `developer/dtrace/toolkit` package is `system/dtrace/dtrace-toolkit`.

```
$ pkg info -r developer/dtrace/toolkit
          Name: developer/dtrace/toolkit
       Summary:
         State: Not installed (Renamed)
    Renamed to: pkg:/system/dtrace/dtrace-toolkit@0.99,5.11-0.173.0.0.0.0.0
                consolidation/osnet/osnet-incorporation
     Publisher: solaris
       Version: 0.99
 Build Release: 5.11
        Branch: 0.173.0.0.0.1.0
Packaging Date: August 26, 2011 02:55:51 PM
          Size: 5.45 kB
          FMRI: pkg://solaris/developer/dtrace/toolkit@0.99,5.11-0.173.0.0.0.1.0:20110826T145551Z
```

The --license option displays the license texts for the packages. This information can be quite lengthy. The information shown above (without the --license option) is not displayed.

```
$ pkg info --license x11/server/xorg
Copyright (c) 2012, Oracle and/or its affiliates. All rights reserved.
...
```

Use the -g option to specify the repository or package archive to use as the source of package data for the operation.

Showing Information from the Package Manifest

The pkg contents command displays the file system content of packages. With no options or operands, this command displays path information for all packages that are installed in the current image. Use command options to specify particular package content to display. To narrow your results, provide one or more package names. You can use wildcards in the package names. Quote the wildcards so that the argument is passed directly to pkg and the shell does not expand it.

```
/usr/bin/pkg contents [-Hmr] [-a attribute=pattern ...] [-g path_or_uri ...]
    [-o attribute ...] [-s sort_key] [-t action_name ...] [pkg_fmri_pattern ...]
```

Both the contents and search subcommands query the contents of packages. The pkg contents command displays actions and attributes of packages. The pkg search command lists the packages that match the query.

The following example shows the pkg contents default behavior. Use options to specify which actions and attributes to display.

```
$ pkg contents zip
PATH
usr
usr/bin
usr/bin/zip
usr/bin/zipcloak
usr/bin/zipnote
usr/bin/zipsplit
usr/share
usr/share/man
usr/share/man/man1
usr/share/man/man1/zip.1
usr/share/man/man1/zipcloak.1
usr/share/man/man1/zipnote.1
usr/share/man/man1/zipsplit.1
```

The -m option displays the entire package manifest.

The -r option displays the newest available versions, retrieving information for any packages not currently installed from the repositories of the configured publishers.

Use the -g option to specify the repository or package archive to use as the source of package data for the operation.

Listing Files Installed by a Package

Use the -t option to specify the type of actions to display. You can specify multiple types in a comma-separated list, or you can specify the -t option multiple times.

Use the -o option to specify the attributes to display in the output. You can specify multiple attributes in a comma-separated list, or you can specify the -o option multiple times. See the pkg(5) man page for a list of package actions and attributes. In this example, the pkg.size pseudo attribute shows the size of the file; the file action does not have a size attribute. See the pkg(1) man page for a list of pseudo attributes.

Use the -s option to sort actions by the specified action attribute. By default, output is sorted by path or by the first attribute specified by the -o option. The -s option can be specified multiple times.

```
$ pkg contents -t file -o owner,group,mode,pkg.size,path -s path zip
OWNER GROUP MODE PKG.SIZE PATH
root  bin   0555   228600 usr/bin/zip
root  bin   0555   107944 usr/bin/zipcloak
root  bin   0555   101856 usr/bin/zipnote
root  bin   0555   106252 usr/bin/zipsplit
root  bin   0444    86036 usr/share/man/man1/zip.1
root  bin   0444     2548 usr/share/man/man1/zipcloak.1
root  bin   0444     2239 usr/share/man/man1/zipnote.1
root  bin   0444     1680 usr/share/man/man1/zipsplit.1
```

If you view the package manifest, you see that the zip package has twelve file actions. The four that are not shown in the above output are files that cannot be installed in this image. This image is an x86 architecture. The files for the SPARC architecture are not shown. See "Controlling Installation of Optional Components" on page 57 for information about variants and facets.

Listing All Installable Packages in a Group Package

The Oracle Solaris 11 GUI installer installs the solaris-desktop group package. The text installer and the default AI manifest in an Automated Installer installation install the solaris-large-server group package. The default installation manifest for non-global zones installs the solaris-small-server group package. The solaris-small-server group package is also an alternative you can use to install a smaller set of packages on a server. You can use the following command to display the set of packages that is included in each group.

```
$ pkg contents -Hro fmri -t depend -a type=group solaris-large-server
archiver/gnu-tar
compress/bzip2
```

```
...
text/texinfo
web/wget
```

The `-t` option matches `depend` actions in the package. The `-a` option matches the `depend` actions that are type `group`. The `-o` option displays only the `fmri` attribute of the group `depend` action. Recall that group packages do not specify content such as files; group packages specify other packages that are part of the group. See "IPS Packages" on page 12 for more information about group packages.

To also show the summary description of each package, use the `pkg list -s` command:

```
$ pkg list -Has ‘pkg contents -Hro fmri -t depend -a type=group solaris-large-server‘
archiver/gnu-tar    GNU version of the tar archiving utility
compress/bzip2      high-quality block-sorting file compressor - utilities
compress/gzip       GNU Zip (gzip)
...
text/texinfo        Documentation system for on-line information and printed output
web/wget            wget - GTNU wget
```

Displaying License Requirements

This example displays all the incorporation packages that require you to accept the package license.

```
$ pkg contents -rt license -a must-accept=true \
-o must-display,license,pkg.name '*incorporation'
MUST-DISPLAY LICENSE                           PKG.NAME
true         usr/src/pkg.license_files/lic_OTN consolidation/osnet/osnet-incorporation
```

Searching for Packages

Use the `pkg search` command to search for packages whose data matches the specified pattern.

```
/usr/bin/pkg search [-HIaflpr] [-o attribute ...] [-s repo_uri] query
```

Like the `pkg contents` command, the `pkg search` command examines the contents of packages. While the `pkg contents` command returns the contents, the `pkg search` command returns the names of packages that match the query.

By default, *query* is interpreted as a series of terms to be matched exactly except for case. Use the `-I` option to specify a case-sensitive search. You can use `?` and `*` wildcards in query terms. You can use single or double quotation marks to search for phrases. Be sure to take your shell into account when you use wildcards or quotation marks.

You can specify more than one query term. By default, multiple terms are joined with AND. You can explicitly join two terms with OR.

Queries can be expressed in the following structured form:

pkg_name:*action_name*:*index*:*token*

Missing fields are implicitly wildcarded. The *pkg_name* and *token* fields can include explicit wildcards. The *action_name* and *index* values must match exactly. The value of *action_name* is the name of an action. The value of *index* is the name of an attribute of the action. See "Actions" in the pkg(5) man page for a list of package actions and attributes. Not all attributes are searchable. For example, `mode` is an attribute of the `file` action, but `mode` is not a valid value for *index*. Some values of *index* are values derived from other attributes. For example, *index* can be `basename`, which is the last component of the `path` attribute of a `file` or `dir` action. Examples of useful values for *index* include `basename` and `path` for `file` and `dir` actions, the dependency type (`require` or `group`, for example) for `depend` actions, and `driver_name` and `alias` for driver actions..

In general, the value of *token* is compared with the value of the attribute named by *index*. For example, in the following partial `driver` action, `alias` is an attribute name that could be specified for *index*, and `pci108e` could be specified for *token*.

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

The syntax of a `set` action is slightly different. The two attributes of a `set` action are `name` and `value`. In this case, the value of *index* is the value of a `name` attribute, and the value of *token* is compared with the value of the matching `value` attribute. For example, in the following partial `set` action, `pkg.summary` could be specified for *index*, and `Broadcom` could be specified for *token*.

```
set name=pkg.summary value="Broadcom 57xx 1GbE NIC Driver"
```

Some well defined values of `set` action `name` attributes include `pkg.fmri`, `info.classification`, `pkg.description`, and `pkg.summary`. See "Set Actions" in the pkg(5) man page.

By default, repositories associated with all publishers configured for this image are searched. Use the `-l` option to search only packages that are installed in this image. Use the `-s` option to specify the URI of the repository to search.

By default, matches are displayed only for currently installed or newer package versions. Use the `-f` option to display all matched versions.

By default, results are displayed for all matching actions, which can yield multiple lines of results for one package. Use the `-p` option to list each matching package only once.

Identifying Which Package Delivers a Specific File

The following examples show that the `libpower` library came from the `system/kernel/power` package.

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

Showing Which Packages Provide Which SMF Services

To show which packages provide a particular SMF service, search for the name of the service as the value of the `org.opensolaris.smf.fmri` attribute.

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

In this case, each attribute has two values: the service name with and without the instance name specified. The following example shows how this attribute is specified in the package manifest:

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

Listing Packages by Category

The following example identifies all packages that have "Source Code Management" in the value of their `info.classification` attribute.

```
$ pkg search 'info.classification:source code management'
INDEX               ACTION VALUE                              PACKAGE
info.classification set    Development/Source Code Management pkg:/developer/versioning/sccs@0.5.11-0.175
info.classification set    Development/Source Code Management pkg:/developer/xopen/xcu4@0.5.11-0.175.1.0.
info.classification set    Development/Source Code Management pkg:/developer/versioning/git@1.7.9.2-0.175
info.classification set    Development/Source Code Management pkg:/developer/versioning/mercurial-27@2.2.
info.classification set    Development/Source Code Management pkg:/library/python-2/subversion@1.7.5-0.17
info.classification set    Development/Source Code Management pkg:/developer/versioning/mercurial-26@2.2.
info.classification set    Development/Source Code Management pkg:/library/java/subversion@1.7.5-0.175.1.
info.classification set    Development/Source Code Management pkg:/developer/quilt@0.60-0.175.1.0.0.21.0
info.classification set    Development/Source Code Management pkg:/developer/versioning/cvs@1.12.13-0.175
info.classification set    Development/Source Code Management pkg:/developer/versioning/subversion@1.7.5-
info.classification set    Development/Source Code Management pkg:/developer/versioning/mercurial@2.2.1-0
info.classification set    Development/Source Code Management pkg:/library/perl-5/subversion@1.7.5-0.175.
```

This example shows a large amount of repeated information that obscures the information that was really wanted.

The following example uses the `-o` option to show only the names of the packages and uses the `-H` option to omit the column heading.

```
$ pkg search -Ho pkg.name 'info.classification:source code management'
developer/versioning/sccs
developer/xopen/xcu4
developer/versioning/git
developer/versioning/mercurial-27
library/python-2/subversion
developer/versioning/mercurial-26
library/java/subversion
developer/quilt
developer/versioning/cvs
developer/versioning/subversion
developer/versioning/mercurial
library/perl-5/subversion
```

Showing Dependent Packages

These examples show the packages that are dependencies of the specified package.

The following example shows packages that have a `require` dependency on the `system/kernel/power` package. If you used the `pkg contents` command to display `depend` actions of type `require` for the `i86pc` and `system/hal` packages, you would see that `system/kernel/power` is listed for both packages.

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

The following example shows that many packages have an `exclude` dependency on `pkg:/x11/server/xorg@1.12.99`.

```
$ pkg search -lo pkg.name,fmri 'depend:exclude:*xorg*'
PKG.NAME                                     FMRI
x11/server/xvnc                              pkg:/x11/server/xorg@1.12.99
x11/server/xorg                              pkg:/x11/server/xorg@1.12.99
x11/server/xorg/driver/xorg-video-mga        pkg:/x11/server/xorg@1.12.99
x11/server/xorg/driver/xorg-video-vesa       pkg:/x11/server/xorg@1.12.99
x11/server/xorg/driver/xorg-input-vmmouse    pkg:/x11/server/xorg@1.12.99
...
```

Listing All Packages in a Group Package

The Oracle Solaris 11 GUI installer installs the `solaris-desktop` group package. The text installer and the default AI manifest in an Automated Installer installation install the `solaris-large-server` group package. The default installation manifest for non-global zones installs the `solaris-small-server` group package. The `solaris-small-server` group package is also an alternative you can use to install a smaller set of packages on a server. You can use the following search form to display the set of packages that is included in each group.

```
$ pkg search -Hfo fmri '*/solaris-large-server:depend:group:*'
archiver/gnu-tar
compress/bzip2
...
text/texinfo
web/wget
```

In this example, `-o pkg.name` would return only the name of the package specified in the *pkg_name* field of the query:

```
group/system/solaris-desktop
```

The `-o fmri` option returns the FMRI of the packages that are specified in the `solaris-large-server` package as `group` type dependencies.

By default, search returns only packages that are installable in this image. In this example, search is not returning matching packages but rather is returning the value of an attribute of an action in a specified package. That attribute value happens to be a package name in this example. The number of results from this command is larger than the number of the results from the similar `pkg contents` command because these search results include the names of all packages that are named in `group depend` actions in the specified package, not just installable packages. For example, package variants might be included that are not installable in this image. Compare the output from this search to the output from the `pkg contents` command shown in "Listing All Installable Packages in a Group Package" on page 30.

Tip – In general, use the `pkg contents` command to show the contents of a specified package, and use the `pkg search` command to show packages that match a query. If you know which package delivers the content that you are interested in, use the `pkg contents` command.

CHAPTER 4

Installing and Updating Software Packages

Package installation and update are affected by image configuration such as constraining some packages to a particular version, configuring publisher search order, and setting package signing properties. Image configuration is discussed in Chapter 5, "Configuring Installed Images."

How to determine which packages are already installed, which packages are available to install, and which packages have updates available is covered in Chapter 3, "Getting Information About Software Packages."

This chapter shows how to perform the following tasks:

- Run a trial installation to see whether the installation would succeed and what would be installed
- Install, update, and uninstall packages
- Validate packages
- Fix problems with installed packages
- Restore an installed file to its original content
- Uninstall packages

"Working with Non-Global Zones" on page 48 discusses aspects of package operations that are unique to non-global zones.

Installing, updating, and uninstalling packages require increased privileges. See "Installation Privileges" on page 16 for more information.

Previewing an Operation

Many of the commands shown in this chapter and in Chapter 5, “Configuring Installed Images,” have an -n option that enables you to see what the command will do without making any changes.

Tip – Best practice is to use the -n option whenever it is available. Use the -n option with one or more verbose options (-nv, -nvv) and review the effects of the command before you execute the command without the -n option.

The following example shows information about a package installation that is not actually performed:

```
$ pfexec pkg install -nv group/feature/amp
           Packages to install: 5
     Estimated space available: 112.19 GB
Estimated space to be consumed: 374.19 MB
       Create boot environment: No
Create backup boot environment: No
            Services to change: 2
          Rebuild boot archive: No
Changed packages:
solaris
  database/mysql-51
    None -> 5.1.37,5.11-0.175.1.0.0.21.0:20120723T165236Z
  database/mysql-common
    None -> 0.5.11,5.11-0.175.1.0.0.21.0:20120723T165447Z
  group/feature/amp
    None -> 0.5.11,5.11-0.175.1.0.0.21.0:20120723T182057Z
  web/server/apache-22/module/apache-dtrace
    None -> 0.3.1,5.11-0.175.1.0.0.21.0:20120723T174611Z
  web/server/apache-22/module/apache-fcgid
    None -> 2.3.6,5.11-0.175.1.0.0.21.0:20120723T174613Z
Services:
  restart_fmri:
    svc:/system/manifest-import:default
    svc:/system/rbac:default
```

The following command produces a large amount of output since so many packages would be affected. Notice the amount of additional space that would be used is in gigabytes, not megabytes. This operation might require a large amount of time and cause a large amount of network traffic between this image and the package repository. Notice that a new BE would not be created by default, but a backup BE would be created. See “Boot Environment Policy Image Properties” on page 72 for information about when BEs are created.

```
$ pfexec pkg change-facet -nv 'facet.locale.*=true'
            Packages to update:       831
     Variants/Facets to change:         1
     Estimated space available: 112.19 GB
Estimated space to be consumed:   2.96 GB
```

```
       Create boot environment:        No
Create backup boot environment:       Yes
          Rebuild boot archive:        No
Changed variants/facets:
    facet facet.locale.*: True
Changed packages:
solaris
  ...
```

Installing and Updating Packages

The `pkg install` command installs packages that are not currently installed and updates packages that are already installed. The `pkg install` command requires one or more package names.

The `pkg update` command updates installed packages. If you specify a package that is not already installed to the `pkg update` command, the system does not install that package. The `pkg update` command takes zero or more names of packages that are already installed. Specifying no package names updates all packages that are installed in the image.

See the `preserve` and `overlay` attributes of the `file` action in the `pkg(1)` man page to understand how files with these attributes are handled during installation and update.

Boot Environment Options

A new BE or a backup BE might be created when you install, update, or uninstall a package. Note that setting or unsetting a mediator, changing a variant or facet, or reverting a file can also involve installing, updating, or uninstalling packages. Within the constraints of the image policy regarding BEs, you can control the creation of new and backup BEs using the options described below. See "Boot Environment Policy Image Properties" on page 72 for information about new BEs and backup BEs and how to set image policy regarding BEs.

Use the BE options to force a new BE or backup BE to be created or not created, to give the BE a custom name, and to specify that the new BE should not be activated.

`--no-be-activate`	If a BE is created, do not set it as the active BE on the next boot. Use the `beadm(1M)` command to show and change the active BE.
`--no-backup-be`	Do not create a backup BE.
`--require-backup-be`	Create a backup BE if a new BE will not be created. Without this option, a backup BE is created based on image policy. See "Boot Environment Policy Image Properties" on page 72 for an explanation of when backup BEs are created automatically.
`--backup-be-name` *name*	If a backup BE is created, name it *name* instead of a default name. Use of `--backup-be-name` implies `--require-backup-be`.

`--deny-new-be`	Do not create a new BE. The install, update, uninstall, or revert operation is not performed if a new BE is required.
`--require-new-be`	Create a new BE. Without this option, a BE is created based on image policy. See "Boot Environment Policy Image Properties" on page 72 for an explanation of when BEs are created automatically. This option cannot be combined with `--require-backup-be`.
`--be-name` *name*	If a BE is created, name it *name* instead of a default name. Use of `--be-name` implies `--require-new-be`.

Installing a New Package

By default, the newest version of a package that is compatible with the rest of the image is installed from the first publisher in the publisher search order that offers the package. To explicitly request the newest version, use `latest` for the version portion of the package FMRI.

If the package is already installed, the package is updated by installing the newest version of the package that is compatible with the rest of the image from the publisher that provided the currently installed version.

If the image has more than one publisher enabled, you can control which publisher provides a package by setting publisher stickiness and search order or by specifying the publisher in the package FMRI. You can also specify the version you want to install in the package FMRI. See "Fault Management Resource Identifiers" on page 13 for a description of a package FMRI. See "Configuring Publishers" on page 53 for information about setting publisher stickiness and search order.

```
/usr/bin/pkg install [-nvq] [-C n] [-g path_or_uri ...]
    [--accept] [--licenses] [--no-index] [--no-refresh] [--no-be-activate]
    [--no-backup-be | --require-backup-be] [--backup-be-name name]
    [--deny-new-be | --require-new-be] [--be-name name]
    [--reject pkg_fmri_pattern ...] pkg_fmri_pattern ...
```

If the *pkg_fmri_pattern* does not specify the publisher, the first publisher that provides a matching package is used as the installation source. If that publisher does not provide a version of the package that can be installed in this image, then the installation operation fails. Use the `pkg list -a` command to see which publishers provide a version of the package that can be installed in this image.

The following commands show that an installable version of the package `atool` is available from a configured publisher, but the publisher that is first in the search order has a version that is not installable in this image.

```
$ 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        ---
```

In this case, the following install command fails. The packaging system finds a match of the *pkg_fmri_pattern* "atool" from the publisher that is first in the search order, but that package cannot be installed.

```
$ pfexec pkg install atool
```

To install this package, make the *pkg_fmri_pattern* more specific, as shown in the following examples:

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

Use the `-nv` option to see what will be installed before you perform the actual installation.

Use the `-g` option to temporarily add the specified package repository or package archive to the list of sources in the image from which to retrieve package data. Repositories that require a client SSL certificate cannot be used with this option. This option cannot be used in images that have child images (non-global zones). If non-global zones are installed in this image, use the `pkg set-publisher` command to add this publisher and origin. This option can be specified multiple times.

When the `-g` option is specified, publishers that are enabled in the image are preferred when retrieving packages.

- If a package that matches the specified *pkg_fmri_pattern* is available from a publisher that is enabled in the image, and if that same publisher is not found in the location specified by the `-g` option, the packaging system attempts to install the package from the publisher that is enabled in the image. After `install` or `update`, any packages provided by publishers not configured in the image are added to the image configuration without an origin.
- If a package that matches the specified *pkg_fmri_pattern* is available from a publisher that is enabled in the image, and if that same publisher publishes the package in the location specified by the `-g` option, the packaging system attempts to install the package from the location specified by the `-g` option.

In the following example, `btool` is available from the `solaris` publisher configured in the image. The `btool` package is also available from the `devtool` publisher with repository origin `http://pkg.example1.com/`, but the `devtool` publisher is not configured in the image. The following command attempts to install the package from the `solaris` publisher because the publisher configured in the image is preferred to the `-g` source when the package is available from the configured publisher.

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

To install the package from the `devtool` publisher, specify the publisher name in the *pkg_fmri_pattern*.

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

In the following example, `isvpub` is a publisher configured in the image with an origin of `/export/isvrepo`. The `isvpub` publisher also publishes packages to a repository at `http://pkg.example2.com/`, but that origin is not specified for the publisher configured in the image. The following command attempts to install the package from the `http://pkg.example2.com/` location because the same publisher provides the package in both locations.

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

See also the description of publisher stickiness in "Adding, Modifying, or Removing Package Publishers" on page 54.

Use the `-C` option to install packages in *n* non-global zones concurrently with the global zone. See "Updating Multiple Non-Global Zones Concurrently" on page 51 for an example of using the `-C` option.

Use the `--accept` option to indicate that you agree to and accept the terms of the licenses of the packages that are updated or installed. If you do not provide this option, and any package licenses require acceptance, the installation operation fails. Use the `--licenses` option to display all of the licenses for the packages that are installed or updated as part of this operation.

When you specify the `--no-index` option, the search indices are not updated after the operation has completed successfully. Specifying this option might save some time if you are installing a large number of packages. When you are finished with all `install`, `update`, and `uninstall` operations, you can use `pkg refresh` to update the list of available packages and publisher metadata for each publisher specified. If no publishers are specified, the refresh is performed for all publishers.

When you specify the `-no-refresh` option, the repositories for the image's publishers are not contacted to retrieve the newest list of available packages and other metadata.

In the command output, note any messages that say a new boot environment has been created. If a new boot environment has been created and activated, that is the environment that is booted by default on next reboot if you do not specify the `--no-be-activate` option.

Installing a Package into a New Boot Environment

Tip – Explicitly specifying a new BE is the safest way to install or update. See "Boot Environment Policy Image Properties" on page 72 for information about when BEs are created.

The new BE is a clone of the current BE with the specified install, uninstall, or update changes applied. The current BE is not modified. The system is not automatically restarted. The new BE is the default boot selection the next time you restart the system. The current BE is still available to be booted.

If you specify the `--no-be-activate` option, the new BE is not the default boot selection the next time you reboot.

Use the `--be-name` option to force a new BE to be created or to give the new BE a meaningful name.

```
$ pfexec pkg install --be-name s11amp group/feature/amp
           Packages to install:   5
       Create boot environment: Yes
Create backup boot environment:  No

DOWNLOAD                                PKGS         FILES    XFER (MB)   SPEED
Completed                                5/5       271/271    52.3/52.3    0B/s

PHASE                                          ITEMS
Installing new actions                       410/410
Updating package state database                 Done
Updating image state                            Done
Creating fast lookup database                   Done
Reading search index                            Done
Updating search index                            5/5

A clone of s11_1 exists and has been updated and activated.
On the next boot the Boot Environment s11amp will be
mounted on '/'.  Reboot when ready to switch to this updated BE.

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

The `pkg list` command reports that the `group/feature/amp` package is not installed because the `group/feature/amp` package is not installed in the current BE. The `group/feature/amp` package is installed in the new `s11amp` BE.

Use the `beadm list` command to check that the system has a new active BE named `s11amp`. The "N" BE is currently booted. The "R" BE is the default on reboot. Use the `beadm activate` command to change which BE is the default on reboot.

```
$ beadm list
BE            Active Mountpoint Space   Policy Created
--            ------ ---------- -----   ------ -------
s11amp        R      -          20.75G  static 2012-08-06 15:36
solaris       -      -          44.81M  static 2010-11-07 17:45
solaris11_1   N      /          30.04M  static 2012-07-25 17:10
```

Check that the `group/feature/amp` package is installed in the new BE. The "i" in the I column indicates that the `group/feature/amp` package is installed.

```
$ pfexec beadm mount s11amp /mnt
$ pkg -R /mnt list group/feature/amp
NAME (PUBLISHER)       VERSION                    IFO
group/feature/amp      0.5.11-0.175.1.0.0.21.0    i--
```

Remember to unmount the `s11amp` BE.

```
$ beadm list
BE            Active Mountpoint Space   Policy Created
--            ------ ---------- -----   ------ -------
s11amp        R      /mnt       20.75G  static 2012-08-06 15:36
solaris       -      -          44.81M  static 2010-11-07 17:45
solaris11_1   N      /          30.05M  static 2012-07-25 17:10
$ pfexec beadm unmount s11amp
```

Rejecting a Package

Use the `--reject` option of the `pkg install` command to prevent packages with names that match the specified *pkg_fmri_pattern* from being installed. If matching packages are already installed, they are removed as part of this operation. Rejected packages that are the target of group dependencies are placed on the avoid list. See "Avoiding Installing Some Packages in a Group Package" on page 66 for information about the avoid list.

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

Updating a Package

You can use either the `install` or `update` subcommand to update an installed package to the newest version of the package that is compatible with the rest of the image from the publisher that provided the currently installed version. To avoid unintentionally installing a package that was not already installed, use the `pkg update` command to update packages.

If the image has more than one publisher enabled, you can control which publisher provides a package by setting publisher stickiness and search order or by specifying the publisher in the package FMRI. You can also specify the version you want to install in the package FMRI. See

"Fault Management Resource Identifiers" on page 13 for a description of a package FMRI. See "Configuring Publishers" on page 53 for information about setting publisher stickiness and search order.

```
/usr/bin/pkg update [-fnvq] [-C n] [-g path_or_uri ...]
    [--accept] [--licenses] [--no-index] [--no-refresh] [--no-be-activate]
    [--no-backup-be | --require-backup-be] [--backup-be-name name]
    [--deny-new-be | --require-new-be] [--be-name name]
    [--reject pkg_fmri_pattern ...] [pkg_fmri_pattern ...]
```

To explicitly request the newest version of a package, use `latest` for the version portion of *pkg_fmri_pattern*.

```
$ pfexec pkg update vim@latest
```

You can specify a package version older than the version that is currently installed to perform an in-place downgrade. Any preserved configuration files that are part of packages to be downgraded and that have been changed since the original version was installed are renamed with the extension `.update`. For more information about how the package system determines which files to preserve, and how these files are preserved during package upgrades, see "File Actions" in the `pkg(5)` man page.

Use the `-g` option to temporarily add the specified package repository or package archive to the list of sources in the image from which to retrieve package data. See "Installing a New Package" on page 40 for additional description and examples of the effects of the `-g` option.

Use the `-C` option to update packages in *n* non-global zones concurrently with the global zone. See "Updating Multiple Non-Global Zones Concurrently" on page 51 for an example.

In the command output, note any messages that say a new boot environment has been created. If a new boot environment has been created and activated, that is the environment that is booted by default on next reboot if you do not specify the `--no-be-activate` option.

Use the `--accept` option to indicate that you agree to and accept the terms of the licenses of the packages that are updated. If you do not provide this option, and any package licenses require acceptance, the update operation fails. Use the `--licenses` option to display all of the licenses for the packages that are updated as part of this operation.

When you specify the `--no-index` option, the search indices are not updated after the operation has completed successfully. Specifying this option might save some time if you are installing a large number of packages. When you are finished with all `install`, `update`, and `uninstall` operations, you can use `pkg refresh` to update the list of available packages and publisher metadata for each publisher specified. If no publishers are specified, the refresh is performed for all publishers.

See “Updating an Image” on page 68 for information about the special behavior of the `pkg update` command when no *pkg-fmri* is specified, or if the *pkg-fmri* specified is an asterisk character (`*`).

Fixing Package Problems

An example of a problem that could occur after a package is installed is that a file delivered by the package becomes corrupted. In the example shown in this section, the `/usr/share/auto_install/manifest/default.xml` file has been deleted.

Use the `pkg search` command to determine which package delivered the missing file:

```
$ pkg search -Hlo pkg.name /usr/share/auto_install/manifest/default.xml
system/install/auto-install/auto-install-common
```

Verifying Package Installation

Use the `pkg verify` command to validate the installation of packages in the current image.

```
/usr/bin/pkg verify [-Hqv] [pkg_fmri_pattern ...]
```

If current signature policy for related publishers is not `ignore`, the signatures of each package are validated based on policy. See `signature-policy` in “Properties for Signing Packages” on page 74 for an explanation of how signature policies are applied.

Use the `-H` option to omit the headers from the verification output. Use the `-q` option to print nothing but return failure if any fatal errors are found. Use the `-v` option to include informational messages regarding packages.

```
$ pfexec pkg verify -v system/install/auto-install/auto-install-common
PACKAGE                                                           STATUS
pkg://solaris/system/install/auto-install/auto-install-common      ERROR
        file: usr/share/auto_install/manifest/default.xml
                Missing: regular file does not exist
```

Fixing Verification Errors

Use the `pkg fix` command to fix package installation errors reported by the `pkg verify` command.

```
/usr/bin/pkg fix [--accept] [--licenses] [pkg_fmri_pattern ...]
```

Verification of installed package content is based on a custom content analysis that might return different results than those of other programs.

Use the `--accept` option to indicate that you agree to and accept the terms of the licenses of the packages that are updated or installed. If you do not provide this option, and any package licenses require acceptance, the fix operation fails. Use the `--licenses` option to display all of the licenses for the packages that are updated as part of this operation.

```
$ pfexec pkg fix --accept system/install/auto-install/auto-install-common
Verifying: pkg://solaris/system/install/auto-install/auto-install-common      ERROR
        file: usr/share/auto_install/manifest/default.xml
                Missing: regular file does not exist
Created ZFS snapshot: 2012-08-06-23:32:03
Repairing: pkg://solaris/system/install/auto-install/auto-install-common
Creating Plan (Evaluating mediators):

DOWNLOAD                                  PKGS       FILES    XFER (MB)   SPEED
Completed                                  1/1         1/1      0.0/0.0    0B/s

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

Restoring a File

Use the `pkg revert` command to restore files to their as-delivered condition.

```
/usr/bin/pkg revert [-nv] [--no-be-activate]
    [--no-backup-be | --require-backup-be] [--backup-be-name name]
    [--deny-new-be | --require-new-be] [--be-name name]
    (--tagged tag-name ... | path-to-file ...)
```

Either all files tagged with a particular *tag-name*, or individual files can be reverted. File ownership and protections are also restored.

Caution – Reverting some editable files to their default values can make the system unbootable, or cause other malfunctions.

In the following example, a sample system configuration file that has been changed is restored to its original content.

```
$ pfexec pkg revert /usr/share/auto_install/sc_profiles/static_network.xml
            Packages to update:  1
       Create boot environment: No
Create backup boot environment: No

DOWNLOAD                                PKGS         FILES    XFER (MB)   SPEED
```

```
Completed                                 1/1           1/1      0.0/0.0    0B/s

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

Uninstalling Packages

Use the pkg uninstall command to remove installed packages.

```
/usr/bin/pkg uninstall [-nvq] [-C n] [--no-index] [--no-be-activate]
    [--no-backup-be | --require-backup-be] [--backup-be-name name]
    [--deny-new-be | --require-new-be] [--be-name name] pkg_fmri_pattern ...
```

If a package is the subject of a group dependency, uninstalling the package places it on the avoid list. See "Avoiding Installing Some Packages in a Group Package" on page 66 for information about the avoid list.

When you specify the `--no-index` option, the search indices are not updated after the operation has completed successfully. Specifying this option might save some time if you are installing a large number of packages. When you are finished with all `install`, `update`, and `uninstall` operations, you can use `pkg refresh` to update the list of available packages and publisher metadata for each publisher specified. If no publishers are specified, the refresh is performed for all publishers.

Use the `-C` option to uninstall packages in *n* non-global zones concurrently with the global zone. See "Updating Multiple Non-Global Zones Concurrently" on page 51 for an example of using the `-C` option.

Working with Non-Global Zones

You can use most IPS commands in a non-global zone the same way you use them in the global zone. Note that Oracle Solaris 10 branded zones are different from Oracle Solaris 11 non-global zones. IPS commands ignore Oracle Solaris 10 branded zones. In this book, "non-global zone" means Oracle Solaris 11 non-global zone.

An important difference between the global zone and non-global zones is the use of package publishers. In a non-global zone, the *system repository* provides access to the package repositories configured in the global zone. Publisher configuration changes made to the global zone are seen immediately by all non-global zones via the system repository.

Installing Packages in Non-Global Zones

Non-global zones can be affected by installing, updating, and uninstalling packages in the global zone. When you run the `pkg update` command with no arguments in the global zone, the global zone and each non-global zone is updated. See "Updating Multiple Non-Global Zones Concurrently" on page 51 for an example. When you specify package names with the install, update, or uninstall commands in the global zone, IPS checks each non-global zone and makes changes only if required to keep the non-global zone compatible with the global zone. Changing facets and variants in the global zone can also affect non-global zones.

Tip – Use the `-n` option to review what changes will be made in non-global zones as well as in the global zone.

When you run package commands while logged into a non-global zone, only that non-global zone is affected. You can install different packages and install different versions of the same package if the result is compatible with the global zone. You can avoid different packages, freeze packages at different versions, set mediators to select different default implementations, and set different facets in the non-global zone image.

Versions of packages installed in a non-global zone can be restricted by the versions installed in the global zone. Some packages cannot be updated or downgraded in a non-global zone because those packages must be the same version in the non-global zone as they are in the global zone. For example, the package named `entire` must be the same in each non-global zone as in the global zone. The `entire` package constrains system package versions so that the resulting set of packages is a supportable image.

In a non-global zone, the *system repository* provides access to the package repositories configured in the global zone. Publisher configuration changes made to the global zone are seen immediately by all non-global zones via the system repository. The system repository will proxy `http`, `https`, and `v4` file repositories and `.p5p` archive repositories.

The zones proxy is a service that enables `pkg` commands running inside a zone to communicate with the system repository, which is running in the global zone. The zones proxy has two parts. The following service runs in the global zone:

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

The following service runs in the non-global zone:

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

See the `pkg.sysrepo`(1M) man page for more information about the system repository and zones proxy services.

The following example shows publishers in a global zone:

```
global:~$ pkg publisher
PUBLISHER              TYPE     STATUS P LOCATION
solaris                origin   online F http://pkg.oracle.com/solaris/release/
solaris                origin   online F file:///export/repoSolaris11/
devtool  (disabled)    origin   online F http://pkg.example1.com/
isvpub                 origin   online F http://pkg.example2.com/
```

The following example shows how these same publishers appear when you are logged into a non-global zone:

```
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 F <system-repository>
```

The `T` in the `P` column means this origin has a proxy. Use either of the following commands to get more information:

```
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/omitted/    -
z1:~$ pkg publisher solaris
            Publisher: solaris
                Alias:
           Origin URI: http://localhost:1008/solaris/91b04f12f39930ae8e27f5636b7a342e8f460133/
              SSL Key: None
             SSL Cert: None
           Origin URI: http://pkg.oracle.com/solaris/release/
                Proxy: http://localhost:1008
              SSL Key: None
             SSL Cert: None
          Client UUID: c92e7a92-dce5-11e1-b7e5-8800209e4377
      Catalog Updated: August  2, 2012 05:10:48 PM
              Enabled: Yes
```

You cannot reconfigure the system repository from within a non-global zone. For example, you cannot change the origins or properties of publishers or the publisher search order of publishers whose location is `<system-repository>`.

If you cannot reach a publisher, you can set a proxy in the global zone by setting the `http_proxy` environment variable or by specifying the `--proxy` option to the `pkg set-publisher` command. See the `pkg`(1) man page and "Adding, Modifying, or Removing Package Publishers" on page 54 for information about the `--proxy` option. See "Proxy Configuration on a System That Has Installed Zones" in *Oracle Solaris 11.1 Administration: Oracle Solaris Zones, Oracle Solaris 10 Zones, and Resource Management* for instructions for setting the `http_proxy` and `https_proxy` environment variables. See the ENVIRONMENT section of the `curl`(1) man page for additional information about proxy environment variables.

To list packages from a specific publisher that is already configured in the global zone, the following format gives the same result in the both the global zone and non-global zones:

```
z1:~$ pkg list -a '//isvpub/*'
NAME (PUBLISHER)    VERSION    IFO
isvtool (isvpub)    1.0-0      ---
```

For repositories that are not configured in the global zone, but that are network or filesystem-accessible to the non-global zone, both of the following commands list the same packages if `file:///export/myrepo` is a repository location that is accessible to the non-global zone:

```
z1:~$ pkg list -af -g file:///export/myrepo
z1:~$ pkgrepo list -s file:///export/myrepo
```

Updating Multiple Non-Global Zones Concurrently

By default, when you use the `pkg update` command in the global zone, the packaging system updates the global zone and each non-global zone serially. To update multiple non-global zones concurrently, use the `-C` option or set the `PKG_CONCURRENCY` environment variable in the global zone. The `-C` *n* option and the `PKG_CONCURRENCY=`*n* environment variable specify to update at most *n* images in parallel for *n* greater than or equal to 1. The default value of n is 1. If *n* is 0 or a negative number, all non-global zones are updated in parallel with the global zone.

The `PKG_CONCURRENCY` environment variable is ignored if the `-C` option is specified. The `-C` option and the `PKG_CONCURRENCY` environment variable can be used with `pkg install`, `pkg uninstall`, `pkg change-variant`, and `pkg change-facet` as well as with `pkg update`.

Non-global zones do not need to be booted to be updated from the global zone. The non-global zones only need to be mounted.

In the following example, both non-global zones are updated at the same time as the global zone:

```
global:~$ pfexec pkg update -C 0 --be-name s11u1
 Startup: Linked image publisher check ... Done
 Startup: Refreshing catalog 'solaris' ... Done
 Startup: Refreshing catalog 'isvpub' ... Done
 Startup: Checking that pkg(5) is up to date ... Done
Planning: Solver setup ... Done
Planning: Running solver ... Done
Planning: Finding local manifests ... Done
Planning: Package planning ... Done
Planning: Merging actions ... Done
Planning: Checking for conflicting actions ... Done
Planning: Consolidating action changes ... Done
Planning: Evaluating mediators ... Done
Planning: Planning completed in 39.00 seconds
```

```
            Packages to remove:   2
           Packages to install:   1
            Packages to update: 640
       Create boot environment: Yes
Create backup boot environment:  No

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

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

CHAPTER 5

Configuring Installed Images

This chapter shows how to configure characteristics that apply to an entire image, such as configuring package publishers, restricting which packages can be installed, setting package signing policy, and configuring boot environment (BE) policy.

Configuring Publishers

To install and update software, you need to be able to contact a package repository.

Displaying Publisher Information

Use the `pkg publisher` command to display information about package publishers configured for this image. The publishers are listed in the order in which they are searched to find packages when the publisher is not specified in the package FMRI.

```
/usr/bin/pkg publisher [-HPn] [-F format] [publisher ...]
```

By default, the `solaris` publisher is configured on a newly installed Oracle Solaris 11 system. Use the `pkg publisher` command to check the origin of your publisher.

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

The TYPE column indicates whether the LOCATION value is an origin or a mirror. See "Repository Origins and Mirrors" on page 15 for descriptions.

Between the STATUS and LOCATION columns, the P column specifies whether the location is proxied. Values in this column are true (T) or false (F). File repositories are never proxied. HTTP repositories with the value F are not proxied, unless an $http_proxy environment variable is currently set (the pkg publisher output still shows F). HTTP repositories with the value T are proxied using the proxy specified with the --proxy option when the origin was added with pkg set-publisher. When you specify the -F tsv option to pkg publisher, the P column contains any proxy that is set for that mirror or origin. See "Installing Packages in Non-Global Zones" on page 49 for an example.

Specify publishers by name to display detailed configuration for those publishers.

```
$ pkg publisher solaris
       Publisher: solaris
           Alias:
      Origin URI: http://pkg.oracle.com/solaris/release/
         SSL Key: None
        SSL Cert: None
     Client UUID: e15e3228-eada-11df-80ab-8023183d954b
 Catalog Updated: July 25, 2012 11:40:03 PM
         Enabled: Yes
      Properties:
                  proxied-urls = []
```

Use the -P option to display only the first publisher in the publisher search order. Use the -n option to display only enabled publishers. The -H option omits headers in the output.

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

Adding, Modifying, or Removing Package Publishers

Use the pkg set-publisher command to perform the following operations:

- Configure a new publisher.
- Set publisher origins and mirrors.
- Enable or disable a publisher. A newly-added publisher is enabled by default. A disabled publisher is not used when populating the package list or in install, uninstall, or update package operations. The properties for a disabled publisher can still be set and viewed. If only one publisher is enabled, that publisher cannot be disabled.

- Set publisher stickiness. A newly-added publisher is sticky by default. If a publisher is non-sticky, then a package that was installed from this publisher could be updated from another publisher.
- Set publisher search order. A newly-added publisher is last in the search order by default. The publisher search order is used to find packages to install. The publisher search order is used to find packages to update if the publisher that the package was originally installed from is non-sticky.

  The first publisher that provides a matching package is used as the installation source. If that publisher does not provide a version of the package that can be installed in this image, then the installation operation fails. To install from a publisher further down the search order, provide more information in the package FMRI, such as the publisher name or the package version string.
- Specify SSL keys and certificates for a publisher.
- Set and unset a publisher property, and add and remove a publisher property value. See "Configuring Package Signature Properties" on page 75.

The `pkg set-publisher` command has two forms. In the following form, the name of the publisher is required:

```
/usr/bin/pkg set-publisher [-Ped] [-k ssl_key] [-c ssl_cert]
    [-g origin_to_add | --add-origin origin_to_add ...]
    [-G origin_to_remove | --remove-origin origin_to_remove ...]
    [-m mirror_to_add | --add-mirror mirror_to_add ...]
    [-M mirror_to_remove | --remove-mirror mirror_to_remove ...]
    [--enable] [--disable] [--no-refresh] [--reset-uuid]
    [--non-sticky] [--sticky] [--search-after publisher]
    [--search-before publisher] [--search-first]
    [--approve-ca-cert path_to_CA]
    [--revoke-ca-cert hash_of_CA_to_remove]
    [--unset-ca-cert hash_of_CA_to_remove]
    [--set-property name_of_property=value]
    [--add-property-value name_of_property=value_to_add]
    [--remove-property-value name_of_property=value_to_remove]
    [--unset-property name_of_property_to_delete]
    [--proxy proxy_to_use] publisher
```

In the following form, the name of the publisher is optional because you have specified the repository URI:

```
/usr/bin/pkg set-publisher -p repo_uri [-Ped]
    [-k ssl_key] [-c ssl_cert] [--non-sticky] [--sticky]
    [--search-after publisher] [--search-before publisher
    [--search-first] [--approve-ca-cert path_to_CA]
    [--revoke-ca-cert hash_of_CA_to_remove]
    [--unset-ca-cert hash_of_CA_to_remove]
    [--set-property name_of_property=value]
    [--add-property-value name_of_property=value_to_add]
    [--remove-property-value name_of_property=value_to_remove]
    [--unset-property name_of_property_to_delete]
    [--proxy proxy_to_use] [publisher]
```

The following command adds a new publisher named `devtool` with an origin URI specified with the `-g` option and sets this publisher to be first in the search order. Use the `-P` option or the `--search-first` option to set the specified publisher first in the search order.

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

The following command enables the `isvpub` publisher and sets it ahead of the `devtool` publisher in the search order.

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

Use the `-p` option to retrieve publisher configuration information from the specified repository URI. If a publisher is specified, then only the matching publisher is added or updated. If no publisher is specified, all publishers are added or updated as appropriate. The `-p` option cannot be combined with the `-g`, `--add-origin`, `-G`, `--remove-origin`, `-m`, `--add-mirror`, `-M`, `--remove-mirror`, `--disable`, `--enable`, `--no-refresh`, or `--reset-uuid` options.

To change the origin URI for a publisher, add the new URI and remove the old URI. Use the `-g` option to add a new origin URI. Use the `-G` option to remove the old origin URI.

```
$ pfexec pkg set-publisher -G '*' -g file:///export/isvrepo/ isvpub
```

The following commands show adding an origin to the `solaris` publisher. If multiple origins are configured for a given publisher in an image, the IPS client attempts to choose the best origin from which to retrieve package data.

```
$ pkg publisher
PUBLISHER                    TYPE     STATUS P LOCATION
solaris                      origin   online F file:///export/repoSolaris11/
$ pfexec pkg set-publisher -g http://pkg.oracle.com/solaris/release/ solaris
$ pkg publisher
PUBLISHER                    TYPE     STATUS P LOCATION
solaris                      origin   online F file:///export/repoSolaris11/
solaris                      origin   online F http://pkg.oracle.com/solaris/release/
```

Use the `-m` option to add a URI as a mirror for the specified publisher. See "Repository Origins and Mirrors" on page 15 for an explanation of the difference between an origin and a mirror. Use the `-M` option to remove a URI as a mirror for the specified publisher.

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

Use the `-k` option to specify the client SSL key. Use the `-c` option to specify the client SSL certificate. Use the `--approve-ca-cert` option to add the specified certificate as a CA certificate that is trusted. The hashes of the user approved CA certificates are listed in the output of the `pkg publisher` command for this publisher. See "Displaying Publisher Information" on page 53.

```
$ pfexec pkg set-publisher -k /root/creds/example.key -c /root/creds/example.cert \
--approve-ca-cert /tmp/example_file.pem isvpub
```

Use the `--revoked-ca-cert` option to treat the specified certificate as revoked. The hashes of the user revoked CA certificates are listed in the output of the `pkg publisher` command for this publisher.

Use the `--unset-ca-cert` option to remove the specified certificate from the list of approved and the list of revoked certificates.

When you specify the `-no-refresh` option, the repositories for the image's publishers are not contacted to retrieve the newest list of available packages and other metadata.

Use the `--reset-uuid` option to choose a new unique identifier that identifies this image to its publisher.

Use the `--proxy` option to specify a persistent web proxy URI from which to retrieve content for the specified origin (`-g`) or mirror (`-m`). The proxy value is stored as part of the publisher configuration. At run time, `$http_proxy` or related environment variables override this proxy setting. See the ENVIRONMENT section of the `curl`(1) man page for additional information about proxy environment variables.

Use the `pkg unset-publisher` command to remove a publisher.

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

Controlling Installation of Optional Components

Software can have components that are optional and components that are mutually exclusive. Examples of optional components include locales and documentation. Examples of mutually exclusive components include SPARC or x86 and debug or non-debug binaries. In IPS, optional components are called *facets* and mutually exclusive components are called *variants*.

Facets and variants are special properties set on the image and are tags set on actions within a package. Most variant tags can have various values. Facet tags set on an action can only have the value `true`. The values of facet and variant tags on an action compared with the values of facets and variants set in the image determine whether that package action can be installed. For example, if you set a particular locale facet to `false`, any files or other actions that specify that facet will not be installed, and currently installed files that specify that facet are uninstalled.

To view the current values of the facets and variants set on the image, use the `pkg facet` and `pkg variant` commands. To modify the values of the facets and variants set on the image, use the `pkg change-facet` and `pkg change-variant` commands. See the pkg(1) man page and the examples below.

Each facet and variant tag has a name and a value. A single action can have multiple facet and variant tags. An example of a component with multiple facet and variant tags is an architecture-specific header file that is used by developers, or a component that is only for a SPARC global zone.

An example of a variant tag is `variant.arch=sparc`. An example of a facet tag is `facet.devel=true`. Facets and variants are often referred to without the leading `facet.` and `variant.`.

Facets are boolean: They can be set only to `true` (enabled) or `false` (disabled). By default, all facets are considered to be set to `true` in the image. A facet tag on an action should only have the value `true`; other values have undefined behavior. A facet set on the image can be a full facet such as `doc.man` or a pattern such as `locale.*`. This flexibility is useful when you want to disable a portion of the facet namespace, and only enable individual facets within it. For example, you could disable all locales and then only enable one or two specific locales, as shown in the following example:

```
$ pfexec pkg change-facet 'locale.*=false'
[output about packages being updated]
$ pfexec pkg change-facet locale.en_US=true
[output about packages being updated]
```

Most variants can have any number of values. For example, the `arch` variant can be set to `i386`, `sparc`, `ppc`, `arm`, or whatever architectures the distribution supports. (Only `i386` and `sparc` are used in Oracle Solaris.) The exception are the `debug` variants. The `debug` variants can only be set to `true` or `false`; other values have undefined behavior. If a file action has both non-debug and debug versions, both versions must have the applicable `debug` variant explicitly set, as shown in the following example:

```
file group=sys mode=0644 overlay=allow owner=root \
  path=etc/motd pkg.csize=115 pkg.size=103 preserve=true \
  variant.debug.osnet=true

file group=sys mode=0644 overlay=allow owner=root \
  path=etc/motd pkg.csize=68 pkg.size=48 preserve=true \
  variant.debug.osnet=false
```

The variant value must be set on the image in order for a package using the variant to be installed. The `arch` and `zone` variants are set by the program that creates the image and installs its initial contents. The `debug.*` variants are `false` in the image by default.

The following algorithm describes how the facets and variants set on the image affect whether a particular action is installed.

- Actions with no facet or variant tags are always installed.
- Actions with facet tags are installed unless all of the facets or facet patterns matching the tags are set to `false` on the image. If any facet is set to `true` or is not explicitly set (`true` is the default), then the action is installed.

- Actions with variant tags are installed only if the values of all the variant tags are the same as the values set in the image.
- Actions with both facet and variant tags are installed if both the facets and the variants allow the action to be installed.

You can create your own facet and variant tags. The following tags are in common use in Oracle Solaris.

Variant Name	Possible Values
`variant.arch`	`sparc`, `i386`
`variant.opensolaris.zone`	`global`, `nonglobal`
`variant.debug.*`	`true`, `false`

The following list shows a small sample of the facet tags that are used in Oracle Solaris:

```
facet.devel            facet.doc
facet.doc.html         facet.doc.info
facet.doc.man          facet.doc.pdf
facet.locale.de        facet.locale.en_GB
facet.locale.en_US     facet.locale.fr
facet.locale.ja_JP     facet.locale.zh_CN
```

You can display the values of variants and facets that are set on the current image, and you can change variants and facets in the current image. Changing variants and facets might update a large number of packages and might require a new BE. Use `-nv` to review what changes will be made before you make any changes.

Showing and Changing Variant Values

Use the `pkg variant` command to display the values of variants that are set.

```
/usr/bin/pkg variant [-H] [variant_nsmr ...]
```

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

Use the `pkg change-variant` command to change the value of a variant.

```
/usr/bin/pkg change-variant [-nvq] [-C n] [-g path_or_uri ...]
    [--accept] [--licenses] [--no-be-activate]
    [--no-backup-be | --require-backup-be] [--backup-be-name name]
    [--deny-new-be | --require-new-be] [--be-name name]
    variant_name=value ...
```

The following command produces a large amount of output since so many packages would be affected. Notice that a new BE would not be created by default, but a backup BE would be created. See "Boot Environment Policy Image Properties" on page 72 for information about when BEs are created.

Use the `-C` option to change variants in *n* non-global zones concurrently with the global zone. See "Updating Multiple Non-Global Zones Concurrently" on page 51 for an example of using the `-C` option.

Use the `-n` option to see what would change if you performed the operation without `-n`, but make no actual changes.

```
$ pfexec pkg change-variant -nv --accept 'variant.debug.*=true'
            Packages to update:       851
     Variants/Facets to change:         3
     Estimated space available:  49.88 GB
Estimated space to be consumed: 270.57 MB
       Create boot environment:        No
Create backup boot environment:       Yes
          Rebuild boot archive:        No

Changed variants/facets:
  variant variant.debug.*: true
    facet facet.locale.en_US: None
    facet facet.locale.*: None
Changed packages:
solaris
  ...
```

Showing and Changing Facet Values

Use the `pkg facet` command to display the values of facets that are set.

```
/usr/bin/pkg facet [-H] [facet_name ...]
```

```
$ pkg facet
FACETS             VALUE
facet.locale.en_US True
facet.locale.en    True
facet.locale.*     False
$ pkg facet -H 'facet.locale.*'
facet.locale.* False
```

Use the `pkg change-facet` command to change the value of a facet.

```
/usr/bin/pkg change-facet [-nvq] [-C n] [-g path_or_uri ...]
    [--accept] [--licenses] [--no-be-activate]
    [--no-backup-be | --require-backup-be] [--backup-be-name name]
    [--deny-new-be | --require-new-be] [--be-name name]
    facet_name=[True|False|None] ...
```

Use the `-C` option to change facets in *n* non-global zones concurrently with the global zone. See "Updating Multiple Non-Global Zones Concurrently" on page 51 for an example of using the `-C` option.

Use the `-n` option to see what would change if you performed the operation without `-n`, but make no actual changes.

If the facet value is set to `None`, the facet specification is removed from the current image.

The following command produces a large amount of output since so many packages would be affected. This operation might require a large amount of time and cause a large amount of network traffic between this image and the package repository. Notice that a new BE would not be created by default, but a backup BE would be created. See "Boot Environment Policy Image Properties" on page 72 for information about when BEs are created.

```
$ pfexec pkg change-facet -nv 'facet.locale.*=true'
            Packages to update:      851
     Variants/Facets to change:        1
     Estimated space available: 49.88 GB
Estimated space to be consumed:  3.13 GB
       Create boot environment:       No
Create backup boot environment:      Yes
          Rebuild boot archive:       No

Changed variants/facets:
    facet facet.locale.*: True
Changed packages:
solaris
  ...
```

Locking Packages to a Specified Version

Use the `pkg freeze` command to constrain a package version. One example of a time to freeze a package is when you do not want the package in a non-global zone to be updated when the global zone is updated.

```
/usr/bin/pkg freeze [-n] [-c reason] [pkg_fmri_pattern] ...
```

If no version is provided in *pkg_fmri_pattern*, the named package must be installed and is constrained to the version installed on the system. If a version is provided in *pkg_fmri_pattern*, then this constraint, or freeze, acts as if an `incorporate` dependency were installed where the `fmri` attribute had the value of the provided package version.

When a package that is frozen is installed or updated, it must end up at a version that matches the version at which it was frozen. For example, if a package was frozen at 1.2, then it could be updated to 1.2.1, 1.2.9, 1.2.0.0.1, and so on. That package could not end up at 1.3, or 1.1.

A publisher specified in the *pkg_fmri_pattern* is used to find matching packages. However, publisher information is not recorded as part of the freeze. A package is frozen with respect to its version only, not its publisher.

Freezing a package that is already frozen replaces the frozen version with the newly specified version.

If no packages are specified, information about currently frozen packages is displayed: package names, versions, when the package was frozen, and any associated reasons.

Freezing a package does not prevent removal of the package. No warning is displayed if the package is removed.

Use the `-c` option to record the reason the package is being frozen. The reason is shown if a freeze prevents an installation or update from succeeding.

Use the `-n` option to perform a trial run of the operation, displaying the list of packages that would be frozen without freezing any packages.

In the following example, the package is frozen at the current installed version. The "f" in the package listing indicates that the package is frozen.

```
$ pfexec pkg freeze -c "Downgrade to avoid bug" library/security/openssl
library/security/openssl was frozen at 1.0.0.10-0.175.1.0.0.18.0:20120611T201116Z
$ pkg freeze
NAME                     VERSION                                    DATE                     COMMENT
library/security/openssl 1.0.0.10-0.175.1.0.0.19.0:20120625T171753Z 29 Jul 2012 17:45:44 PDT Downgrade to
avoid bug
$ pkg list library/security/openssl
NAME (PUBLISHER)                    VERSION                    IFO
library/security/openssl            1.0.0.10-0.175.1.0.0.18.0  if-
```

When you try to install a different version of the frozen package, you see a message about the freeze.

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

A freeze is never lifted automatically by the packaging system. To relax a constraint, use the `pkg unfreeze` command.

`/usr/bin/pkg unfreeze [-n] [`*pkg_name_pattern*`] ...`

Remove the constraints that freezing imposes from the specified packages. Any versions provided are ignored.

Use the `-n` option to perform a trial run of the unfreeze, displaying the list of packages that would be unfrozen without unfreezing any packages.

Relaxing Version Constraints Specified by Incorporations

Every package that is part of the Oracle Solaris 11 OS has a dependency on an incorporation package. Incorporation packages constrain the versions of the their incorporated packages to help keep the system in a supportable state across updates. Some incorporated packages might be safe to downgrade or upgrade at a version different from the version specified by the incorporation. Such incorporated packages have a `version-lock.`*pkg_name* facet attribute specified in the incorporation package. The default value of the `version-lock.`*pkg_name* facet is `true`. To relax the version constraint on a package, set the value of its `version-lock.`*pkg_name* facet to `false`.

In the following example, you want to downgrade to an earlier version of package. The `pkg update` command downgrades as well as upgrades packages.

```
$ pkg list -af library/security/openssl
NAME (PUBLISHER)                    VERSION                     IFO
library/security/openssl            1.0.0.10-0.175.1.0.0.19.0  i--
library/security/openssl            1.0.0.10-0.175.1.0.0.18.0  ---
$ pfexec pkg update library/security/openssl@1.0.0.10-0.175.1.0.0.18.0
Creating Plan (Solver setup): |
pkg update: No matching version of library/security/openssl can be installed:
  Reject: pkg://solaris/library/security/openssl@1.0.0.10,5.11-0.175.1.0.0.18.0:20120611T201116Z
  Reason:  This version is excluded by installed incorporation
  pkg://solaris/consolidation/userland/userland-incorporation@0.5.11,5.11-0.175.1.0.0.19.0:20120625T163952Z
```

To relax the version constraint on this package, set its `version-lock` facet to `false`. Then try the downgrade again. Notice that a new BE is not created, but a backup BE is created. See "Boot Environment Policy Image Properties" on page 72 for information about when BEs are created.

```
$ pfexec 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
$ pfexec pkg update library/security/openssl@1.0.0.10-0.175.1.0.0.18.0
            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            1.0.0.10-0.175.1.0.0.18.0  i--
```

To prevent this package from being downgraded or upgraded, freeze the package at the current version. The "f" in the package listing indicates that the package is frozen.

```
$ pfexec pkg freeze -c "Downgrade to avoid bug" library/security/openssl
library/security/openssl was frozen at 1.0.0.10-0.175.1.0.0.18.0:20120611T201116Z
$ pkg list library/security/openssl
NAME (PUBLISHER)                    VERSION                    IFO
library/security/openssl            1.0.0.10-0.175.1.0.0.18.0  if-
```

To re-enable downgrade or upgrade, use the `pkg unfreeze` command to remove the version freeze. If the package is installed at a version lower than the version specified in the incorporation package, setting the `version-lock` facet for this package to `true` installs the version specified in the incorporation package.

If other installed packages have `require` dependency relationships with the package that you want to downgrade or upgrade, you might need to also relax version constraints on those related packages. In the following example, version constraints have been lifted on the `hexedit` package, but installation is rejected because of version constraints on the `system/library` package.

```
$ pfexec pkg install editor/hexedit@1.2.12-0.175.1.0.0.21.0
Creating Plan (Solver setup): -
pkg install: No matching version of editor/hexedit can be installed:
  Reject: pkg://solaris/editor/hexedit@1.2.12,5.11-0.175.1.0.0.21.0:20120723T170720Z
  Reason:  All versions matching ’require’ dependency
  pkg:/system/library@0.5.11,5.11-0.175.1.0.0.20.0 are rejected
    Reject: pkg://solaris/system/library@0.5.11,5.11-0.175.1.0.0.20.0:20120709T163421Z

pkg://solaris/system/library@0.5.11,5.11-0.175.1.0.0.21.0:20120723T163000Z
    Reason:  This version is excluded by installed incorporation
    pkg://solaris/consolidation/osnet/osnet-incorporation@0.5.11,5.11-0.175.1.0.0.19.0:20120625T152525Z
```

In addition to individual component packages, you can also relax version constraints on incorporations. In this case, setting the `version-lock` facet to `false` enables you to unlock the incorporation from the rest of the system, while the packages it incorporates continue to be synchronized.

Specifying a Default Application Implementation

You might want to provide multiple versions of an application or tool in the same image. If the different versions of the application are delivered as part of the same mediation, then you can easily reset the version that is the default or preferred version. A *mediation* is a set of links to different implementations of an application, where each of the links has the same mediator name and the same link path but different target link paths.

Use the `pkg mediator` command to display all mediators in the image or display the current selected version of the specified mediator.

```
/usr/bin/pkg mediator [-aH] [-F format] [mediator ...]
```

Use the `pkg set-mediator` command to reset the version of a specified mediator that is the default or preferred version.

```
usr/bin/pkg set-mediator [-nv] [-I implementation]
    [-V version] [--no-be-activate]
    [--no-backup-be | --require-backup-be]
    [--backup-be-name name]
    [--deny-new-be | --require-new-be] [--be-name name]
    mediator ...
```

In the following example, two different versions of the Java Runtime Environment are installed.

```
$ pkg list 'runtime/java*'
NAME (PUBLISHER)     VERSION                     IFO
runtime/java         1.6.0.33-0.175.1.0.0.18.1  i--
runtime/java/jre-6   1.6.0.33-0.175.1.0.0.18.1  i--
runtime/java/jre-7   1.7.0.5-0.175.1.0.0.18.0   i--
```

The following command shows that a `java` mediation is defined, and version 1.7 is the default version.

```
$ pkg mediator
MEDIATOR VER. SRC. VERSION IMPL. SRC. IMPLEMENTATION
java     system    1.7     system
php      system    5.2     system
python   vendor    2.6     vendor
```

The following command shows all `java` mediations that can be set. Both `jre-6` and `jre-7` define a symbolic link from `/usr/bin/java`. In the `jre-6` package, the target of the `/usr/bin/java` link is `jdk1.6`. In the `jre-7` package, the target of the `/usr/bin/java` link is `jdk1.7`. The previous command showed that version 1.7 is currently the target of the `/usr/bin/java` link. This mediation does not specify the preferred implementation, and the package system selected the implementation with the higher version as the preferred implementation.

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

```
java     system    1.7     system
java     system    1.6     system
```

The following command shows setting version 1.6 as the preferred implementation. This means that invoking `/usr/bin/java` will invoke JRE version 1.6. JRE version 1.7 is still available on the system when users specify the full path to that version. Compare the output of the two `pkg mediator` commands.

```
$ pkg mediator java
MEDIATOR VER. SRC. VERSION IMPL. SRC. IMPLEMENTATION
java     system    1.7     system
$ pfexec pkg set-mediator -V 1.6 java
            Packages to update:  3
           Mediators to change:  1
       Create boot environment: No
Create backup boot environment: No

PHASE                                         ITEMS
Removing old actions                            2/2
Updating modified actions                       3/3
Updating image state                           Done
Creating fast lookup database                  Done
Reading search index                           Done
Updating search index                           3/3
$ pkg mediator java
MEDIATOR VER. SRC. VERSION IMPL. SRC. IMPLEMENTATION
java     local     1.6     system
```

Avoiding Installing Some Packages in a Group Package

Use the `pkg avoid` command to avoid installing specified packages if they are the target of a `group` dependency. You can always explicitly install a package that is compatible with the image, even if that package is on the avoid list. Installing a package that is on the avoid list removes that package from the avoid list. The `pkg avoid` command enables you to avoid installing specified packages that are part of a group package when you install that group package.

```
/usr/bin/pkg avoid [pkg_fmri_pattern ...]
```

With no arguments, the `pkg avoid` command displays each avoided package along with any packages that have a group dependency on that package.

With *pkg_fmri_pattern* specified, the `pkg avoid` command places the package names that currently match the specified patterns on the avoid list. Only packages that are not currently installed can be avoided. If a package is currently the target of a group dependency, uninstalling the package places it on the avoid list.

If a package is on the avoid list, installing it removes it from that list. Packages that are on the avoid list are installed if needed to satisfy a `require` dependency. If that dependency is removed, the package is uninstalled.

Use the `pkg unavoid` command to remove the specified packages from the avoid list.

```
/usr/bin/pkg unavoid [pkg_fmri_pattern ...]
```

Packages on the avoid list that match an installed package's group dependency cannot be removed from the avoid list using this subcommand. To remove a package from the avoid list that matches a group dependency, install the package.

The following command output shows that the `group/feature/amp` group package is not installed, and none of the packages that are part of that group package are installed. Some of these packages could have been installed explicitly or as `require` dependencies of other packages.

```
$ pkg list -a group/feature/amp
NAME (PUBLISHER)     VERSION                     IFO
group/feature/amp    0.5.11-0.175.1.0.0.21.0     ---
$ pkg list -a ‘pkg contents -o fmri -H -rt depend -a type=group group/feature/amp‘
NAME (PUBLISHER)                             VERSION                  IFO
database/mysql-51                            5.1.37-0.175.1.0.0.21.0  ---
web/php-52                                   5.2.17-0.175.1.0.0.21.0  ---
web/php-52/extension/php-apc                 3.0.19-0.175.1.0.0.21.0  ---
web/php-52/extension/php-mysql               5.2.17-0.175.1.0.0.21.0  ---
web/server/apache-22                         2.2.22-0.175.1.0.0.21.0  ---
web/server/apache-22/module/apache-dtrace    0.3.1-0.175.1.0.0.21.0   ---
web/server/apache-22/module/apache-fcgid     2.3.6-0.175.1.0.0.21.0   ---
web/server/apache-22/module/apache-php5      5.2.17-0.175.1.0.0.18    --r
```

The following command places one of the packages that belongs to this group package on the avoid list. The group package is not noted on the avoid list because the group package is not installed.

```
$ pfexec pkg avoid web/server/apache-22/module/apache-fcgid
$ pkg avoid
    web/server/apache-22/module/apache-fcgid
```

The following commands show that the avoided package is not installed when the group package is installed. After the group package is installed, the group package is noted on the avoid list.

```
$ pfexec pkg install group/feature/amp
...
$ pkg list -a ‘pkg contents -o fmri -H -rt depend -a type=group group/feature/amp‘
NAME (PUBLISHER)                             VERSION                  IFO
database/mysql-51                            5.1.37-0.175.1.0.0.21.0  i--
web/php-52                                   5.2.17-0.175.1.0.0.21.0  i--
web/php-52/extension/php-apc                 3.0.19-0.175.1.0.0.21.0  i--
web/php-52/extension/php-mysql               5.2.17-0.175.1.0.0.21.0  i--
web/server/apache-22                         2.2.22-0.175.1.0.0.21.0  i--
web/server/apache-22/module/apache-dtrace    0.3.1-0.175.1.0.0.21.0   i--
web/server/apache-22/module/apache-fcgid     2.3.6-0.175.1.0.0.21.0   ---
web/server/apache-22/module/apache-php5      5.2.17-0.175.1.0.0.18    i-r
$ pkg avoid
    web/server/apache-22/module/apache-fcgid (group dependency of ’group/feature/amp’)
```

The pkg unavoid command does not remove a package from the avoid list if that package is part of an installed group package. To remove such a package from the avoid list, install the package.

```
$ pfexec pkg unavoid web/server/apache-22/module/apache-fcgid
pkg unavoid: The following packages are a target of group dependencies; use install to unavoid these:
    web/server/apache-22/module/apache-fcgid
$ pfexec pkg install web/server/apache-22/module/apache-fcgid
$ pkg avoid
$
```

You cannot place a package on the avoid list if that package is already installed. The package is placed on the avoid list if you uninstall the package.

```
$ pfexec pkg avoid web/server/apache-22/module/apache-fcgid
pkg avoid: The following packages are already installed in this image; use uninstall to avoid these:
    web/server/apache-22/module/apache-fcgid
$ pfexec pkg uninstall web/server/apache-22/module/apache-fcgid
...
$ pkg avoid
    web/server/apache-22/module/apache-fcgid (group dependency of ’group/feature/amp’)
```

Uninstalling a package that is part of a group package automatically places that package on the avoid list.

```
$ pfexec pkg uninstall database/mysql-51
$ pkg avoid
    database/mysql-51 (group dependency of ’group/feature/amp’)
    web/server/apache-22/module/apache-fcgid (group dependency of ’group/feature/amp’)
```

If the group package is uninstalled, the avoided packages remain on the avoid list, but the avoid list no longer notes their association with the group package.

```
$ pfexec pkg uninstall group/feature/amp
$ pkg avoid
    database/mysql-51
    web/server/apache-22/module/apache-fcgid
$ pfexec pkg unavoid database/mysql-51 web/server/apache-22/module/apache-fcgid
$ pkg avoid
$
```

Updating an Image

Use the pkg update command with no *pkg-fmri* specified, or with an asterisk character (*) as the *pkg-fmri*, to update all installed packages that have updates available to the newest version allowed by the constraints imposed on the system by installed package dependencies and publisher configuration. If non-global zones are mounted in the current image, these zones are also updated. See “Updating Multiple Non-Global Zones Concurrently” on page 51.

```
/usr/bin/pkg update [-fnvq] [-C n] [-g path_or_uri ...]
    [--accept] [--licenses] [--no-index] [--no-refresh] [--no-be-activate]
```

```
[--no-backup-be | --require-backup-be] [--backup-be-name name]
[--deny-new-be | --require-new-be] [--be-name name]
[--reject pkg_fmri_pattern ...] [pkg_fmri_pattern ...]
```

Image Update Best Practices

Before you use the `pkg update` command, check the versions that are available from your configured publisher origin, and use the `-nv` options to display the list of packages that will be updated without actually performing the update.

If you want to update your operating system release, check the available versions of the `entire` incorporation package. The following command shows that Oracle Solaris 11 11/11 SRU 10 is installed, Oracle Solaris 11 11/11 SRUs 11, 12, and 13 are available, and Oracle Solaris 11.1 is available from the currently configured `solaris` publisher. For information about fields in the FMRI, see "Fault Management Resource Identifiers" on page 13.

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

If none of these versions is what you want, then you need to set your `solaris` publisher origin to a different package repository location.

By default, each package is updated from the publisher that provided the currently installed version. You can control the publisher that provides packages by specifying publisher stickiness and search order. See "Adding, Modifying, or Removing Package Publishers" on page 54.

The following command shows which packages, if any, would actually be installed by an update. Because the `-v` option is specified, this command shows the full FMRIs, including versions, of all 627 packages that would be updated, the three packages that would be removed, and the one new package that would be installed. This example omits most of that output and only shows the `entire` package. Because the `-n` option is specified, no update is actually done. Review this output before you perform an update without the `-n` option.

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

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

The preceding example shows that the `entire` incorporation package for Oracle Solaris 11.1 would be installed. All installed packages that belong to the `entire` incorporation would be updated accordingly. Because no package FMRI was specified, any installed packages that do not belong to the `entire` incorporation would also be updated. All installed packages would be updated to the newest version allowed by the constraints imposed on the system by installed package dependencies and publisher configuration. Installed packages can be removed and new packages can be installed when updated installed packages specify new dependencies.

The preceding example shows that a new BE would be created for this update if you reran this command without the `-n` option. If you run this command without the `-n` option, you see the following message at the end of the update output:

```
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.
```

The current BE is not modified. All changes are made in the new BE.

Explicitly specifying a new BE is the safest way to install or update. See "Boot Environment Policy Image Properties" on page 72 for information about when BEs are created. You might want to use the `--be-name` option to give the new BE a meaningful name. The new BE is activated, so this new environment is booted by default the next time you boot the system. If that is not what you want, use the `--no-be-activate` option with the `pkg update` command. Then when you are ready to use the new environment, use the `beadm activate` command to activate the new BE.

Specifying the Version to Install

If you do not want to update to the newest version allowed, you can specify the package name on the `pkg update` command, including a portion of the version string. The following example shows how to specify the version of the `entire` incorporation to update to Oracle Solaris 11 11/11 SRU 13, even though a newer version would be allowed. Be sure to use the `-nv` options again and check the output again.

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

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

Some installed packages might not belong to the `entire` incorporation. Those packages might have been installed separately and will not be updated by updating just the `entire` incorporation. You can add those packages to the same `pkg update` command.

Specifying a Version Constraint Prior to Updating

If you want to allow updates to any Oracle Solaris 11 11/11 version but not allow update to Oracle Solaris 11.1, you can freeze the `entire` incorporation as shown in the following command. Specifying `0.175.0` means the `entire` package can be updated to `0.175.0.13`, for example, but not to `0.175.1`.

```
$ pfexec pkg freeze -c "Keep this image at 11 11/11." entire@0.5.11,5.11-0.175.0
entire was frozen at 0.5.11,5.11-0.175.0
$ pkg freeze
NAME    VERSION             DATE                     COMMENT
entire  0.5.11,5.11-0.175.0 30 Jan 2013 15:50:01 PST Keep this image at 11 11/11.
$ pkg list entire
NAME (PUBLISHER)  VERSION                       IFO
entire            0.5.11,5.11-0.175.0.10.0.5.0  if-
```

For more information about package freezing, see “Locking Packages to a Specified Version” on page 61.

Constraining the Available Packages

Another way to control what versions can be installed or updated is to provide your own local IPS package repository and control that repository content. For example, your repository could contain all the support updates for Oracle Solaris 11 11/11 but not contain Oracle Solaris 11.1 packages.

If you create your own repository, be sure to update your `solaris` publisher origin.

For more information, see *Copying and Creating Oracle Solaris 11.1 Package Repositories*.

Downgrading an Image

To downgrade your operating system release, boot into a BE older than the version you want to downgrade to, and upgrade from there. For example, if you updated from Oracle Solaris 11 11/11 SRU 10 to Oracle Solaris 11 11/11 SRU 13 and then realized you need an SRU 12 image, reboot to your SRU 10 BE and update to SRU 12 from there.

More Update Command Options

When you specify the `-f` option when updating all installed packages, the client up-to-date check is not executed.

Use the `-g` option to temporarily add the specified package repository or package archive to the list of sources in the image from which to retrieve package data. See "Installing a New Package" on page 40 for additional description and examples of the effects of the `-g` option.

Use the `-C` option to update *n* non-global zones concurrently with the global zone. See "Updating Multiple Non-Global Zones Concurrently" on page 51 for an example.

Use the `--accept` option to indicate that you agree to and accept the terms of the licenses of the packages that are updated. If you do not provide this option, and any package licenses require acceptance, the update operation fails. Use the `--licenses` option to display all of the licenses for the packages that are updated as part of this operation.

When you specify the `-no-refresh` option, the repositories for the image's publishers are not contacted to retrieve the newest list of available packages and other metadata.

When you specify the `--no-index` option, the search indices are not updated after the operation has completed successfully. Specifying this option might save some time if you are installing a large number of packages. When the update operation finishes, you can use `pkg refresh` to update the list of available packages and publisher metadata for each publisher specified. If no publishers are specified, the refresh is performed for all publishers. If a new BE was created, do this publisher refresh in the new BE.

Configuring Image and Publisher Properties

To implement image policies, set image properties. This section describes image and publisher properties and how to set these properties. See also "Image Properties" in the `pkg(1)` man page for descriptions of image properties.

Boot Environment Policy Image Properties

An image is a location where IPS packages can be installed and where other IPS operations can be performed.

A boot environment (BE) is bootable instance of an image. You can maintain multiple BEs on your system, and each BE can have different software versions installed. When you boot your system, you have the option to boot into any of the BEs on the system. A new BE can be created automatically as a result of package operations. You can also explicitly create a new BE. Whether a new BE is created depends on image policy, as discussed in this section

By default, a new BE is automatically created when you perform one of the following operations:

- Update particular key system packages such as some drivers and other kernel components. This can happen when you install, uninstall, update, change variant, or change facet.

  Often a new BE is created when you execute the `pkg update` command to update all packages that have updates available.
- Specify any of the following options: `--be-name`, `--require-new-be`, `--backup-be-name`, `--require-backup-be`.
- Set the `be-policy` image policy to `always-new`. Under this policy, all package operations are performed in a new BE set as active on the next boot.

When a new BE is created, the system performs the following steps:

1. Creates a clone of the current BE.

   The clone BE includes everything hierarchically under the main root dataset of the original BE. Shared file systems are not under the root dataset and are not cloned. Instead, the new BE accesses the original shared file systems.
2. Updates the packages in the clone BE, but does not update any packages in the current BE.

   If non-global zones are configured in the current BE, these existing zones are configured in the new BE.
3. Sets the new BE as the default boot choice the next time the system is booted, unless `--no-be-activate` is specified. The current BE remains as an alternate boot choice.

If a new BE is required but not enough space is available to create a new BE, you might be able to delete existing unneeded BEs. For more information about BEs, see *Creating and Administering Oracle Solaris 11.1 Boot Environments*.

See "Setting Image Properties" on page 77 for instructions to set the image properties described below.

`be-policy`
Specifies when a BE is created during packaging operations. The following values are allowed:

`default`	Apply the default BE creation policy: `create-backup`.
`always-new`	Require a reboot for all package operations by performing them in a new BE set as active on the next boot. A backup BE is not created unless explicitly requested.

This policy is the safest, but is more strict than most sites need since no packages can be added without a reboot.

`create-backup`
For package operations that require a reboot, this policy creates a new BE set as active on the next boot. If packages are modified or content that could affect the kernel is installed and the operation affects the live BE, a backup BE is created but not set as active. A backup BE can also be explicitly requested.

This policy is potentially risky only if newly installed software causes system instability, which is possible, but relatively rare.

`when-required`
For package operations that require a reboot, this policy creates a new BE set as active on the next boot. A backup BE is not created unless explicitly requested.

This policy carries the greatest risk since if a packaging change to the live BE makes further changes impossible, a recent fallback BE might not exist.

Properties for Signing Packages

If you are installing signed packages, set the image properties and publisher properties described in this section to verify package signatures.

Image Properties for Signed Packages

Configure the following image properties to use signed packages.

`signature-policy`
The value of this property determines what checks will be performed on manifests when installing, updating, modifying, or verifying packages in the image. The final policy applied to a package depends on the combination of image policy and publisher policy. The combination will be at least as strict as the stricter of the two policies taken individually. By default, the package client does not check whether certificates have been revoked. To enable those checks, which might require the client to contact external web sites, set the `check-certificate-revocation` image property to `true`. The following values are allowed:

`ignore`	Ignore signatures for all manifests.
`verify`	Verify that all manifests with signatures are validly signed, but do not require all installed packages to be signed. This is the default value.
`require-signatures`	Require that all newly installed packages have at least one valid signature. The `pkg fix` and `pkg verify` commands also warn if an installed package does not have a valid signature.

`require-names`	Follow the same requirements as `require-signatures` but also require that the strings listed in the `signature-required-names` image property appear as a common name of the certificates used to verify the chains of trust of the signatures.

`signature-required-names`
The value of this property is a list of names that must be seen as common names of certificates while validating the signatures of a package.

Publisher Properties for Signed Packages

Configure the following publisher properties to use signed packages from a particular publisher.

`signature-policy`
The function of this property is identical to the function of the `signature-policy` image property except that this property only applies to packages from the specified publisher.

`signature-required-names`
The function of this property is identical to the function of the `signature-required-names` image property except that this property only applies to packages from the specified publisher.

Configuring Package Signature Properties

Use the `set-property`, `add-property-value`, `remove-property-value`, and `unset-property` subcommands to configure package signature properties for this image.

Use the `--set-property`, `--add-property-value`, `--remove-property-value`, and `--unset-property` options of the `set-publisher` subcommand to specify signature policy and required names for a particular publisher.

The following example configures this image to require all packages to be signed. This example also requires the string "oracle.com" to be seen as a common name for one of the certificates in the chain of trust.

```
$ pfexec pkg set-property signature-policy require-names oracle.com
```

The following example configures this image to require all signed packages to be verified.

```
$ pfexec pkg set-property signature-policy verify
```

The following example configures this image to require that all packages installed from the publisher `example.com` must be signed.

```
$ pfexec pkg set-publisher --set-property signature-policy=require-signatures example.com
```

The following example adds a required signature name. This example adds the string `trustedname` to the image's list of common names that must be seen in a signature's chain of trust to be considered valid.

```
$ pfexec pkg add-property-value signature-require-names trustedname
```

The following example removes a required signature name. This example removes the string `trustedname` from the image's list of common names that must be seen in a signature's chain of trust to be considered valid.

```
$ pfexec pkg remove-property-value signature-require-names trustedname
```

The following example adds a required signature name for a specified publisher. This example adds the string `trustedname` to the `example.com` publisher's list of common names that must be seen in a signature's chain of trust to be considered valid.

```
$ pfexec pkg set-publisher --add-property-value \
signature-require-names=trustedname example.com
```

Additional Image Properties

`ca-path`
: Specifies a path name that points to a directory where CA certificates are kept for SSL operations. The format of this directory is specific to the underlying SSL implementation. To use an alternate location for trusted CA certificates, change this value to point to a different directory. See the `CApath` portions of `SSL_CTX_load_verify_locations`(3openssl) for requirements for the CA directory.

  The default value is `/etc/openssl/certs`.

`check-certificate-revocation`
: If this is set to `True`, the package client attempts to contact any CRL distribution points in the certificates used for signature verification to determine whether the certificate has been revoked since being issued.

  The default value is `False`.

`flush-content-cache-on-success`
: If this is set to `True`, the package client removes the files in its content-cache when install or update operations complete. For update operations, the content is removed only from the source BE. When a packaging operation next occurs in the destination BE, the package client flushes its content cache if this option has not been changed.

  This property can be used to keep the content-cache small on systems with limited disk space. This property can cause operations to take longer to complete.

  The default value is `True`.

`mirror-discovery`
: This property tells the client to discover link-local content mirrors using mDNS and DNS-SD. If this property is set to `True`, the client attempts to download package content from mirrors it dynamically discovers. To run a mirror that advertises its content via mDNS, see `pkg.depotd`(1M).

  The default value is `False`.

`send-uuid`
: Send the image's Universally Unique Identifier (UUID) when performing network operations. Although users can disable this option, some network repositories might refuse to talk to clients that do not supply a UUID.

  The default value is `True`.

`trust-anchor-directory`
: The value of this property is the path name of the directory that contains the trust anchors for the image. This path is relative to the image.

  The default value is `ignore`.

`use-system-repo`
: This property indicates whether the image should use the system repository as a source for image and publisher configuration and as a proxy for communicating with the publishers provided. See `pkg.sysrepo`(1M) for information about system repositories.

  The default value is `ignore`.

Setting Image Properties

Use the `set-property`, `add-property-value`, `remove-property-value`, and `unset-property` subcommands to configure properties for this image.

```
/usr/bin/pkg property [-H] [propname ...]
/usr/bin/pkg set-property propname propvalue
/usr/bin/pkg add-property-value propname propvalue
/usr/bin/pkg remove-property-value propname propvalue
/usr/bin/pkg unset-property propname ...
```

Displaying the Values of Image Properties

Use the `pkg property` command to view the properties of an image.

```
$ 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
```

The `preferred-authority` and `publisher-search-order` properties can be set using `pkg set-publisher` command options. See "Adding, Modifying, or Removing Package Publishers" on page 54.

Setting the Value of an Image Property

Use the `pkg set-property` command to set the value of an image property or add and set a property.

The following example sets the value of the `mirror-discovery` property.

```
$ pfexec pkg set-property mirror-discovery True
$ pkg property -H mirror-discovery
mirror-discovery True
```

Resetting the Value of an Image Property

Use the `pkg unset-property` command to reset the values of the specified properties to their default values.

```
$ pfexec pkg unset-property mirror-discovery
$ pkg property -H mirror-discovery
mirror-discovery False
```

Creating an Image

An image is a location where IPS packages and their associated files, directories, links, and dependencies can be installed, and where other IPS operations can be performed.

An image can be one of three types:

- Full images are capable of providing a complete system. In a full image, all dependencies are resolved within the image itself, and IPS maintains the dependencies in a consistent manner. After you have completed an installation of the Oracle Solaris OS, the root file system and its contents are contained in a full image.
- Partial images are linked to a full image (the parent image), but do not provide a complete system on their own. A non-global zone is a partial image. Use the `-z` or `--zone` option to set an appropriate variant. In a zone image, IPS maintains the non-global zone consistent with its global zone as defined by dependencies in the packages. See Part II, "Oracle Solaris Zones," in *Oracle Solaris Administration: Oracle Solaris Zones, Oracle Solaris 10 Zones, and Resource Management* to learn about non-global zones.
- User images contain only relocatable packages.

```
/usr/bin/pkg image-create [-FPUfz] [--force]
    [--full | --partial | --user] [--zone]
    [-k ssl_key] [-c ssl_cert] [--no-refresh]
    [--variant variant_name=value ...]
    [-g path_or_uri | --origin path_or_uri ...]
    [-m uri | --mirror uri ...]
    [--set-property name_of_property=value]
    [--facet facet_name=(True|False) ...]
    [(-p | --publisher) [name=]repo_uri] dir
```

At the location given by *dir*, create an image suitable for package operations. The default image type is user (`-U` or `--user`). The image type can be set to a full image (`-F` or `--full`) or to a partial image (`-P` or `--partial`) linked to the full image enclosing the given *dir* path.

To run the new image in a non-global zone context, use the `-z` or `--zone` option to set an appropriate variant.

A package repository URI must be provided using the `-p` or `--publisher` option. If a publisher name is also provided, then only that publisher is added when the image is created. If a publisher name is not provided, then all publishers known by the specified repository are added to the image. An attempt to retrieve the catalog associated with this publisher is made following the initial creation operations.

Use the `-g` option to specify additional origins. Use the `-m` option to specify mirrors.

For publishers using client SSL authentication, use the `-c` or `-k` options to register a client key and client certificate. This key and certificate are used for all publishers added during image creation.

Use the `-f` option to force the creation of an image over an existing image. Use this option with caution.

When you specify the `-no-refresh` option, the repositories for the image's publishers are not contacted to retrieve the newest list of available packages and other metadata.

Use the `--variant` option to set the specified variant to the indicated value. Use the `--facet` option to set the specified facet to the indicated value.

Viewing Operation History

Use the `pkg history` command to view the command history in the current image.

```
/usr/bin/pkg history [-HNl] [-t [time | time-time],...]
    [-o column,...] [-n number]
```

Use the `-l` option to display more information, including the outcome of the command, the time the command completed, the version and name of the client used, the name of the user who performed the operation, and any errors encountered while executing the command.

Use the `-n` option to display only the specified number of most recent operations.

```
$ pkg history -n4
START                    OPERATION                CLIENT             OUTCOME
2012-08-06T16:32:03      fix                      pkg                Succeeded
2012-08-06T16:41:47      revert                   pkg                Succeeded
2012-08-06T17:56:22      set-property             pkg                Succeeded
2012-08-06T17:56:53      unset-property           pkg                Succeeded
```

Use the `-o` option to display output using the specified comma-separated list of column names. See the list of column names in pkg(1).

```
$ pkg history -o start,time,operation,outcome -n4
START                    TIME      OPERATION                OUTCOME
2012-08-06T16:32:03      0:00:27   fix                      Succeeded
2012-08-06T16:41:47      0:00:43   revert                   Succeeded
2012-08-06T17:56:22      0:00:00   set-property             Succeeded
2012-08-06T17:56:53      0:00:00   unset-property           Succeeded
```

Use the `-t` option to log records for a comma-separated list of timestamps, formatted with `%Y-%m-%dT%H:%M:%S` (see `strftime`(3C)). To specify a range of times, use a hyphen (`-`) between a start and finish timestamp. The keyword `now` can be used as an alias for the current time. If the timestamps specified contain duplicate timestamps or overlapping date ranges, only a single instance of each duplicate history event is printed.

Use the `-N` option to display any release note text for the operation. The `-N` option cannot be used with the `-o` option.

Use the `pkg purge-history` command to delete all command history information.

```
$ pfexec pkg purge-history
```