Installing and Checking Packages 3

This chapter describes how to install and check your software package. You should install from your CD-ROM image and verify that the installation is correct before having CD-ROMs manufactured. The following topics are discussed:

- Installation software database

  Describes the database that keeps track of the packages that have been installed.

- Installing software packages

  Briefly describes the installation command `pkgadd`. Installing software for clients on a server is also discussed.

- Checking installation accuracy and displaying information about installed packages

  Describes how to use the `pkgchk` command to check the integrity of your packages after they have been installed. Also describes the various types of information you can display with the `pkginfo` command.

3

The Installation Software Database

Information for all packages installed on a system is kept in the installation software database. There is an entry for every object in a package, with information such as the component name, where it resides, and its type. An entry contains a record of the package to which a component belongs; other packages that might reference the component; and information such as pathname, where the component resides and the component type. Entries are added and removed automatically by `pkgadd` and `pkgrm`. You can view the information in the database by using the `pkgchk` command.

Two types of information are associated with each package component. The *attribute information* describes the component itself. For example, the component's access permissions, owner ID, and group ID are attribute information. The *content information* describes the contents of the component, such as file size and time of last modification.

The installation software database keeps track of the package status. A package can be either fully installed, (it has successfully completed the installation process), or partially installed (it did not successfully complete the installation process).

When a package is partially installed, portions of a package may have been installed before installation was terminated; thus, part of the package is installed, and recorded in the database, and part is not. When you reinstall the package, you are prompted to start at the point where installation stopped because `pkgadd` can access the database and detect which portions have already been installed. You can also remove the portions that have been installed, based on the information in the installation software database.

You can use the `pkginfo` command to survey the contents of the installation software database. The commands `installf` and `removef` can be used to modify its contents.

Installing Software Packages

The default installation mode is interactive. To install a software package named `pkgA` from a disk device named `/dev/dsk/c0t0d0s0`, you would enter the following command:

```
# pkgadd -d /dev/dsk/c0t0d0s0 pkgA
```

You can install multiple packages at the same time, as long as you separate package names with a space, as follows:

```
# pkgadd -d /dev/dsk/c0t0d0s0 pkgA pkgB pkgC
```

If you do not name the device on which the package resides, the command checks the default spool directory (`/var/spool/pkg`). If the package is not there, installation fails. The name given after the `-d` option must be a full pathname to a device, directory (as shown in the example), or device alias.

Note – You must use a package identifier if multiple versions reside on the installation medium. In most cases, there is only one instance of a package on a medium and the package identifier is the package abbreviation without a suffix.

Be aware that the suffix of a package identifier defines the package instance on that particular medium. A new package identifier is assigned to this package when it has been installed on your system. (Use `pkginfo -d` *device* to find out what instances are on a medium.)

Interacting with `pkgadd`

When `pkgadd` encounters a problem, it first checks the `admin` file for instructions. (See the `admin`(4) manual page for details on the format of this file.) If no instructions exist, or if the parameter is defined as `ask`, `pkgadd` displays a message describing the problem and prompts for a reply. The prompt is usually `Do you want to continue with this installation`. You should respond with `yes`, `no`, or `quit`. If you have

specified more than one package, `no` stops installation of the package being installed but `pkgadd` continues with installation of the other packages. `quit` indicates that `pkgadd` should stop installation of all packages.

Installing Packages for Clients on a Server

This section describes how to install packages for a client that place files in the `root` file system. Packages that do not place files in `root` can be made available to clients by installing the package on the server with `pkgadd`. These packages are then made available when the file systems are mounted by the clients.

Unbundled software packages should be installed into `/opt/PKG`. However, some packages, such as a package containing a device driver, must be installed into `/` or `/usr`.

Installing Packages on a Server for Diskless Clients

You use `pkgadd` on a server to install software either for the use of clients. Software installed for the use of clients is installed in the client's `root` file system, not the server's `root` file system.

A diskless client's `root` file system is located on the server, in the directory `/export/root/client`. The client's `/usr` file system is located in `/export/exec/`*os_identifier*`/usr`, where *os_identifier* is a string that identifies the operating system, version, and instruction architecture of the client.

Use the `pkgadd` command with the `-R` option to specify the location of the client's root filesystem for the installation. For example:

```
# /usr/sbin/pkgadd -d device -R root_path
```

Files installed in the client's `root` file system appear in the client's software database as `installed`. Files that the client expects to find in its `/usr` file system are shown as shared in the client's database. The shared files must be installed on the server with a separate invocation of `pkgadd`.

You can use the `-R` option with other package commands, for example:

```
# /usr/sbin/pkgchk -R root_path
# /usr/sbin/pkgrm -R root_path
```

Installing Packages on a Server for Dataless Clients

Installing a dataless client is similar to the installation procedure covered under "Installing Packages on a Server for Diskless Clients." The client's `root` file system must be a remote mount on the server. The client must export its root file system with read and write access to the server's root.

After mounting the client's root filesystem on the server, use the `pkgadd` command with the `-R` option to specify the root filesystem of the client for the installation. For example:

```
# /usr/sbin/pkgadd -d device -R root_path
```

Files installed into the client's `root` file system appear in the client's software database as `installed`. Files that the client expects to find in the `/usr` file system are shown as shared in the client's database. The shared files must be installed on the server with a separate invocation of `pkgadd`.

Checking Installation Accuracy

`pkgchk` (1M) enables you to check the accuracy of installed files or display information about package files. It checks the integrity of directory structures and the files. `pkgchk` can list or check the following:

- Contents or attributes, or both, of objects currently installed on the system
- Contents of a spooled, uninstalled package
- Contents or attributes, or both, of objects described in the specified `pkgmap` file

For more detailed information about this command, refer to the `pkgchk`(1M) manual page.

pkgchk performs two kinds of checks. It checks file attributes (the permissions and ownership of a file and major/minor numbers for block or character special devices) and the file contents (the size, checksum, and modification date). By default, the command checks both the file attributes and the file contents.

The pkgchk command also compares the file attributes and contents of the installed package against the installation software database. The entries concerning a package may have been changed since the time of installation; for example, another package may have changed a package component. The database reflects that change.

If you use the -f option to pkgchk, file attributes are corrected when discrepancies are found.

Displaying Information About Installed Packages

You can use two commands to display information about packages:

- pkgparam displays parameter values
- pkginfo displays information from the software database

The pkgparam *Command*

pkgparam enables you to display the values associated with the parameters you have requested on the command line. The values are retrieved from either the pkginfo file for pkginst or from the file you name. One parameter value is shown per line. You can display the values only or the parameters and their values.

For detailed information, refer to the pkgparam (1) manual page.

The pkginfo *Command*

You can display information about installed packages with the pkginfo command. pkginfo has several options that enable you to customize both the format and the contents of the display.

You can request information about any number of package instances.

Parameter Descriptions for the `pkginfo` *Display*

Table 3-1 describes the package parameters that can be displayed for each package. A parameter and its value are displayed only when the parameter has a value assigned to it.

*Table 3-1*Package Parameters

Parameter	Description
`ARCH`	The architecture supported by this package.
`BASEDIR`	The base directory in which the software package resides (shown if the package is relocatable).
`CATEGORY`	The software category, or categories, of which this package is a member (for example, `system` or `application`).
`CLASSES`	A list of classes defined for a package. The order of the list determines the order in which the classes are installed. Classes listed first will be installed first (on a media by media basis). This parameter may be modified by the request script.
`DESC`	Text that describes the package.
`EMAIL`	The electronic mail address for user inquiries.
`HOTLINE`	Information on how to receive hotline help concerning this package.
`INTONLY`	Indicates that the package should only be installed interactively when set to any non-NULL value.
`ISTATES`	A list of allowable run states for package installation (for example, `S s 1`).
`MAXINST`	The maximum number of package instances that should be allowed on a machine at the same time. By default, only one instance of a package is allowed. This parameter must be set to allow for multiple instances of a package.
`NAME`	The package name, generally text describing the package abbreviation.
`ORDER`	A list of classes defining the order in which they should be put on the medium. Used by `pkgmk` in creating the package. Classes not defined in this parameter are placed on the medium using the standard ordering procedures.
`PKGINST`	Abbreviation for the package being installed.
`PSTAMP`	The production stamp for this package
`RSTATES`	A list of allowable run states for package removal (for example, `S s 1`).

*Table 3-1*Package Parameters

ULIMIT	If set, this parameter is passed as an argument to the `ulimit` command, which establishes the maximum size of a file during installation. This applies only to files created by control scripts.
VENDOR	The name of the vendor who supplied the software package.
VERSION	The version of this package.
VSTOCK	The vendor-supplied stock number.

You can request that all spooled packages on a particular device, or in a particular directory, be included in the `pkginfo` list by using the `-d` option. For example, the following command shows information in the extracted format for all the packages in the spool directory `/opt/spooldir`:

```
# pkginfo -d /opt/spooldir -x
```

For detailed information about the `pkginfo` command, refer to the `pkginfo`(4) manual page.

The Default `pkginfo` *Display*

When `pkginfo` is executed without options, it displays the category, package instance, and package name of all packages that have been completely installed on your system. The display is organized by categories as shown in the following example.

```
$ pkginfo
system int Installation Utilities
system backup Backup/Restore Utilities
application pkgA Package A
application pkgA.2 Package A
application anpkg Another Package
$
```

Customizing the Format of the `pkginfo` *Display*

You can get a `pkginfo` display in any of three formats: short, extracted, and long.

The short format is the default. It shows only the category, package abbreviation, and full package name. It presents one line of information per package.

The extracted format shows the package abbreviation, package name, package architecture (if available), and package version (if available). Use the `-x` option to request the extracted format as shown in the next example.

```
$ pkginfo -x pkgA anpkg
pkgA Package A
(SunOS) Release 2, Version 3
anpkg Another Package
(SunOS) Release 4
```

Using the `-l` option produces a display in the long format showing all of the available information about a package, as in the following example.

```
$ pkginfo -l mypkg
PKGINST: pkgA.3
NAME: Package A
CATEGORY: application
ARCH: SunOS
VERSION: Version 3
INSTDATE: Tue Apr 14 08:41:40 MDT 1988
BASEDIR: /opt/pkgA
VSTOCK: sdr9000
STATUS: completely installed
FILES: 31 installed
                3 linked files
               10 directories
               13 executable
               nnnn blocks used (approx)
               SERIALNUM: 201-790b
$
```

Customizing the Contents of the `pkginfo` *Display*

You can use the `pkginfo` options to specify packages to be included in the display. See the `pkginfo` (1M) manual page for a description of the options.

3