Chapter 5
Installing Add−on Software

Introduction

Software management responsibilities include setting up your software installation environment, installing software on your computer, keeping track of it while it is on your system, removing it as necessary, and (optionally) setting up your system as an Install Server. In particular, you will need to do the following tasks:

- Create and use ‘‘admin’’’’ files, which define values for your installation default parameters.
- Store interactions with a package or set (using the `pkgask` command) in a file to be used when installing software in non−interactive mode.
- Install software packages or sets using the command `pkgadd`. It can be used to execute both interactive and non−interactive installation.
- Install software from a remote computer. Examples show how to use the `pkginstall` command and Network File System (NFS), which make remote filenames appear to be local, to install packages from a remote computer.
- Use the `pkgchk` command to check the integrity of packages or sets after they have been installed.
- Display information about installed packages or sets using the `pkginfo` command. Various types of information about packages or sets installed on your system can be displayed.
- Setting up your system as an Install Server. This allows remote installation of UnixWare onto other systems in the network.
- Store packages without installing them. (You can spool a package onto your computer for future installation.)
- Remove packages or sets with the `pkgrm` command.

If you’re using the UNIX Desktop (a graphical user−interface with windows and icons), some of these tasks may be available there. See descriptions of the System Setup folder in Desktop User Handbook for details.

If the Operations, Administration and Maintenance (OA&M) package (a non−graphical user−interface) is installed on your system you can use it to complete many of these tasks. See *Installing Add−On Software Through OA&M Menus* later in this chapter.

Basic Software Management Terminology and Concepts

To install software packages or sets properly you need to understand the basic terms defined in the following sections. This section also describes the installation software database and the differences between interactive and non−interactive installation, and explains how to name packages or sets on the command line.

Add−On Package Terminology

A ‘‘software package’’ is a separately installable software component that provides one or more capabilities or services to users. The package may contain compiled programs, files, and installation scripts.

A ''set'' is a special–purpose package, referred to as a Set Installation Package (SIP), and a collection of one or more packages that are members of the set. The SIP controls the installation of the set.

Both packages and sets are delivered on an ''installation medium.'' An installation medium is any physical storage device, such as a diskette or tape, on which packages can be stored.

A ''network installation'' is a process to install software that resides on one system in a network onto another system on the network.

Packages are delivered on a medium in one of two formats: a ''file system format'' or a ''datastream format.'' (The datastream format consists of a header and a series of `cpio` archives, and it can be read from any raw device.) `pkgadd` automatically determines the format of the medium when it reads the first volume. The format of a package can be translated (either from file system to datastream or vice–versa) by running the `pkgtrans` command. [See pkgtrans(1) in the Command Reference and Software Development Tools for details.] Sets are only delivered in datastream format.

Software packages or sets can be installed from a directory. This is quite useful in remote file sharing environments where a directory containing packages or sets is advertised from a server computer to a large network.

You can also ''spool'' a package for later installation. Spooling causes the contents of the package to be copied from the installation medium to a spool directory. No other installation action is taken. (For example, the installation scripts are not executed.)

You cannot spool a set directly. Use `pkginfo` to determine which packages are members of the set, and then spool the package(s).

Package Instances

Variations of a software package may reside on your system simultaneously. Each variation of a package is known as an ''instance'' and is treated as a separate entity. Three parameters, defined by the package developer, combine to uniquely identify each entity. You cannot install on the same system two instances of a package that have identical values for all three parameters. These parameters are:

`PKG`
: The package abbreviation (remains constant for every instance of a package)

`VERSION`
: the software package version

`ARCH`
: the software package architecture

For example, two versions of a package that run on the same system might be identified as:

```
PKG="abbr" PKG="abbr"
VERSION="4.2 v1" VERSION="4.2 v1.1"
ARCH="(386)" ARCH="(386)"
```

Each package instance has a ''package identifier.'' This ID maps the three identifying parameters to one name, the name of this package instance on your system.

Relocatable Packages

Some packages, or parts of a package (files), must be installed in particular directories. Other packages have no requirement for where to install the package. The package developer can specify that files are relocatable (that is, that they may be placed in arbitrary locations).

The directory portion of a relocatable object pathname is filled in as the package is installed. There are four places from which the directory value can originate:

- The package itself may ask where you want to install relocatable objects.
- `pkgadd` asks you where you want to install a relocatable package if no default is set in the admin file and you are in interactive mode.
- The admin file can assign a value to an installation default parameter that defines the directory in which to install relocatable packages. If set, the value of this parameter is used as the directory portion of the name.
- The package also delivers the name of a directory in which to install relocatable objects if there is a question about where to place them (for example, if you have no default defined in the admin file and are operating in non−interactive mode).

At installation, all relocatable objects are given full pathnames and any variable value associated with them is resolved.

The Installation Software Database

The installation software database stores information about all packages and sets installed on the system. Entries for every component of a package or set contain a record of the package to which the component belongs, the component name, where it resides, its type, and a list of other packages that might reference the component. The database also keeps ‘‘attribute information’’ (such as the component’s access permissions, owner ID, and group ID) and content information (such as the file size and the time of last modification).

The system uses this database to see if an object is shared by more than one package, to see if other packages depend on it, or to perform a number of other checks when adding or removing a package or set. When a package or set is installed or removed with `pkgadd` or `pkgrm`, information about it is automatically added to, or removed from, this database.

The installation software database keeps track of the status of packages. A package can be either fully installed, meaning it has successfully completed installation, or partially installed, meaning it has not successfully completed installation. In the latter case, you can remove the portions that were installed, based on the information stored in the installation software database.

You can use the pkginfo(1) command to survey the contents of the installation software database. The commands installf(1M) and removef(1M) can be used to modify its contents.

Methods of Installation

You can install software packages or sets in either of two modes: ‘‘interactive’’ or ‘‘non−interactive.’’ In interactive mode `pkgadd` queries you during installation and receives input on how to proceed. In non−interactive mode, you give `pkgadd` the information it needs by supplying two files: an admin file and a response file (discussed later in the sections Creating an Admin File and Creating a Response File). You can also spool a package for later installation.

If `pkgadd` encounters a problem and cannot find instructions in the admin file or response file, it terminates the installation.

Network Installation

Packages and sets can be installed from CD−ROM, cartridge tape, diskette, or over a network from an already configured Install Server. An Install Server is an Application Server that is configured to provide software products and packages to other systems in the network. Once an Install Server is configured, subsequent software installation of additional machines can be performed over the network.

Network installation is particularly useful if you have to install the same packages on many machines, or if your network includes multiple sites. Network installation avoids the hassle of transporting media from site to site.

The entire Personal Edition (PE) and/or Application Server (AS) can be installed over the network. Although this requires the install and network installation diskettes, no other installation media (CD−ROM, cartridge tape, or diskette) is needed. This provides an efficient way to install UnixWare, especially if the system you are installing on does not have a CD−ROM or cartridge tape drive.

Installation Default Parameters

The commands to install and remove packages and sets, `pkgadd` and `pkgrm`, check for errors during their execution. When a problem occurs, they refer to an installation administration file for instructions on how to proceed. This admin file defines values for parameters, each of which supplies a resolution for a potential problem.

The default admin file, `default`, specifies that no checking will be done, except to see if there is enough room to install the package. (`default` causes the installation to proceed without prompting.) You can name an alternative admin file by using the `−a` *`admin`* option, where *`admin`* is the name of the alternative admin file. For package installation, for example, you may choose to install interactively by specifying the `check` admin file as the argument to `−a`. (`check` causes a prompt to appear whenever a problem or conflict occurs.)

For set installation or removal, always use the `default` admin file. Both `check` and `default` are located in `/var/sadm/install/admin`.

If you want to assign different values for your installation default parameters, you can create your own admin file(s).

> **NOTE:** Do not make changes to the system−supplied admin files `default` or `check.` If you want to define different parameter values, create a new admin file. [See admin(4) for details.]

In some instances, non−interactive installation of a package or set requires a response file. This file holds answers to questions for which the installation script would prompt you to provide answers if you were installing the same package or set interactively.

For further details, see the following sections:

- Creating an Admin File (instructions for creating an admin file)
- Creating a Response File (instructions for creating a response file)

Creating an Admin File

When you invoke the `pkgadd` command, it automatically uses the system−supplied admin file, `default`. If you want to use a set of other defaults for different installation situations, you can create an alternative admin file(s).

Procedure

Use the following procedure to create an admin file:

1. Create a file with the editor of your choice. You can use any name for a new admin file.
2. Define each parameter on a single line in the following format:

   `param=value`

   A description of each parameter you can define, along with a list of permissible values, is given on theadmin(4) manual page. Any of these parameters may be assigned the value `ask`, which means that if the situation occurs you will be asked to give instructions at that time.

You do not have to assign values to every parameter. If `pkgadd` needs a parameter value and one is not assigned in the admin file, the default value `ask` is used.

> **NOTE:** The value `ask` cannot be defined in an admin file to be used for non−interactive installation. If it is, installation will fail when a problem occurs.

Example

The `check` admin file (`/var/sadm/install/admin/check`) defines `mail` as `root` and all other parameters as `ask`. The following sample admin file defines parameters differently than `check`.

Figure 5−1. Sample admin File

```
basedir=default
runlevel=quit
conflict=quit
set-UID=quit
action=quit
partial=quit
instance=unique
idepend=quit
rdepend=quit
space=quit
```

Creating a Response File

Before you install a package or set in non−interactive mode, you must prepare answers for the questions that a package installation script would ask you during the installation process. The `pkgask` command executes the appropriate installation script, thus showing you the questions and allowing you to respond to them. Your answers are stored in a file or a directory called a ‘‘response file.’’

You supply a name for the response file on the command line when you execute `pkgadd` to install the package or set in non−interactive mode. The installation script will use the response file to access the information when it is needed.

Procedure

Use this procedure to create a response file:

> **NOTE:** You must use a package identifier with a numerical suffix if multiple versions reside on the installation medium. When there is only one version of a package on a medium, the package identifier is the package abbreviation without a suffix. The package identifier suffix defines the package instance only on that particular medium. A new suffix will be assigned when this package is installed on your system. To find out what instances are available on a medium, run `pkginfo -d` *`device`*.

- Execute the following command:

  ```
  pkgask -r response pkginst
  ```

  where *`response`* is the full pathname of the file or directory in which your responses will be saved and *`pkginst`* is the package identifier or Set Installation Package (SIP) for the package or set to be installed.

- If a SIP is specified as the `pkginst`, the response file must be a directory. If `pkginst` specifies a SIP, request scripts are run for all packages that are members of the set, and the resulting response files are placed in the directory specified in the −r option. (Use pkginfo to find out the package identifier or SIP if you do not know it. See Displaying Information About All Installed Packages for details on pkginfo.) If *response* is a directory, a file named response/pkginst is created.

Software Package and Set Installation

Software package and set installation is the process of copying a software package from an installation medium (such as a diskette or tape cartridge) onto your computer, performing any actions requested by installation scripts, and recording package objects in the installation software database. Installation can be performed in either interactive or non−interactive mode.

Installing a Package or Set in Interactive Mode

You can install a package in interactive mode. This choice, which is the default mode, takes full advantage of the sophistication of the pkgadd command. You simply specify −a check with the pkgadd command, and if problems occur during installation, you will be notified so you can instruct pkgadd how to proceed. For example, if you supply a device name but not a package name or SIP, pkgadd will show you a list of packages or sets on the device (or in the directory) specified and ask you to choose one to install.

In addition, pkgadd checks for a number of potential problems. If pkgadd finds a potential problem, it needs instructions on how to proceed. You can create a list of default solutions in an admin file and specify that file on the command line. [See admin(4) for a description of admin files.]

Before You Begin

When installing a package in interactive mode, first decide how you want to define your installation defaults for this installation.

Choose an admin file that establishes these defaults, or create a new admin file by using an editor. (This is not applicable for sets since the default file is used.)

Procedure

Use the following instructions to interactively install a software package or set:

- Execute the pkgadd command. Example: to install a package or SIP named pkgA from a tape device named ctape1, enter:

  ```
  pkgadd −d ctape1 pkgA
  ```

  Example: you can install multiple packages at one time by separating package names with white space. The named packages must all be on one medium (one tape or one diskette), or pkgadd will fail with an error. A sample command for installing multiple packages at one time follows:

  ```
  pkgadd −d ctape1 pkgA pkgB pkgC
  ```

  NOTE: You must use a package identifier if multiple versions co−reside on the installation medium. In most cases, there will be only one instance of a package on a medium and the package identifier will be 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 will be assigned this package when it has been installed on your system. (Run pkginfo −d *device* to find out which instances are available on a medium.)

- 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 or directory or the device alias (as shown in the previous example).

- When `pkgadd` encounters a problem, it first checks the admin file you have named with the `-a` *`admin`* option for instructions. If you have specified that the `check` admin file should be used, or if the instructions in any admin file say to consult the administrator (meaning a parameter is defined as `ask`), `pkgadd` gives you a message describing the problem and prompts you for a reply. (If the `default` admin file is used, no prompts will be displayed).

  The prompt is usually `Do you want to continue with this installation`. Respond with `yes`, `no`, or `quit`.

  If you have given more than one package (such as `pkgA`, `pkgB`, and `pkgC` in the previous example) and you respond with `no`, the current installation is stopped but `pkgadd` is instructed to continue with installation of the other packages. If you enter `quit`, `pkgadd` stops installation of all packages.

  See Installation Default Parameters, earlier in this chapter, and the admin(4) manual page for a description of the potential problems for which `pkgadd` checks and the default instructions you can define for each problem.

- Unless you specify differently, `pkgadd` uses the admin file `default`. To inform `pkgadd` that you want to use an alternative admin file, invoke it with the `-a` option.

  To install a package named `mypkg` using an admin file named `myadmin`, execute the following:

  ```
  pkgadd -d diskette1 -a myadmin mypkg
  ```

  If you want more checking to be done, specify the `check` admin file:

  ```
  pkgadd -a check pkginst
  ```

- If you are installing a set, use the `default` admin file. `default` will be used automatically in the absence of the `-a` *`admin`* option. Example:

  ```
  pkgadd -d diskette1 sipname
  ```

  is equivalent to

  ```
  pkgadd -d diskette1 -a default sipname
  ```

- If you put your admin file in the `/var/sadm/install/admin` directory, you need only specify the filename as the argument to the `-a` option when executing `pkgadd`. You can create admin files in other directories but if you do, `-a` *`admin`* must specify the full pathname of the file.

  NOTE: These examples show the use of a device alias, `diskette1`, in place of a device pathname, such as `/dev/dsk/c0b0t0d0s0`. The device alias is defined in the device database (`/etc/device.tab`), as described in the Managing Storage Devices chapter in Storage Devices and File System Administration.

Installing a Package or Set in Non−interactive Mode

You can install a package or set in non−interactive mode. For some types of installation−−−such as those done in background or by batch execution−−−interactive mode cannot be used; non−interactive mode is mandatory. In non−interactive mode, you do not supply input to the `pkgadd` command or monitor its output.

When you do run `pkgadd` non−interactively (with the `-n` option), you must specify a response file. A response file contains a list of answers to problems not addressed by the instructions in the admin file. When `pkgadd` encounters problems during a non−interactive installation it checks both the admin file (for default solutions) and the response file (for system specific solutions). Before installing a package non−interactively, you need to determine whether either or both types of files are needed. The advantage of non−interactive mode is that a package can be installed with no need for your intervention. As a result, the installation may be done more quickly than an interactive installation. The drawback is that you cannot supply instructions if unanticipated problems arise. For sets, all interaction is accomplished immediately after the `pkgadd` command is executed, so you do not have to provide input later in the process.

Before You Begin

When installing a package in non−interactive mode:

1. decide how to define installation defaults for this installation. Choose an admin file that establishes these defaults, or create a new admin file. This step is important in a non−interactive installation because the system will not be able to question you if problems arise.

2. determine whether you need to create a response file. Some packages include scripts that ask for input during installation. If you are installing such a package or set, you will have to store answers to the script's questions in a response file. To create a response file see the procedure in the section Creating a Response File.

Procedure

Use this procedure to install a package or set in non−interactive mode:

- Install the package by running a command like the following:

  ```
  pkgadd -n -d device -r response -a admin pkginst
  ```

  where the `−n` option requests non−interactive mode, the `−d` *`device`* option specifies the name of the device (or directory) on which the package or set currently resides, the `−r` *`response`* option specifies a response file, the `−a` *`admin`* option specifies an admin file, and *`pkginst`* is the package identifier or SIP of the package to be installed. See the pkgadd(1M) manual page for complete information on other options you may also want to use, such as the `−l` option, which creates a log file of messages issued by `pkgadd` that you can review later.

  NOTE: As shown here, the full pathname of the response file must be provided.

  If `pkginst` is a package identifier (not a SIP), and `−r` names a specific response file, you can install only one package at a time. But, multiple packages can be installed with one command. To accomplish this, use the `−r` option to specify the name of a directory that holds several response files corresponding to the packages you want to install. Each response file must be given the same name as the package to which it corresponds. For example, if you enter `pkgadd -d` *`device`* −n −r *`response_dir`* `pkgA pkgB pkgC` `pkgadd` will use *`response_dir`*/`pkgA`, *`response_dir`*/`pkgB`, and *`response_dir`*/`pkgC` to install `pkgA`, `pkgB`, and `pkgC`, respectively.

  To install `mypkg` in non−interactive mode, supplying a response file named `myresponse`, enter the following command:

  ```
  pkgadd −d diskette1 −n −r /pkgA/myresponse mypkg
  ```

Installing Software From a Remote Machine: An Example With pkginstall

Some software packages and sets can be installed from a remote computer. The following example uses the pkginstall(1M) command to do so.

In this example, the administrator of computer_A wants to install all the packages that are on a cartridge tape at another site. An administrator at the other site has agreed to spool the package onto computer_B (which is also on the network) so that the software can be installed on computer_A.

NOTE: For this example, computer_B must be configured as an Install_Server and network installation must be enabled. For details, see the section Setting Up and Administering an Install Server later in this chapter.

1. The computer_B administrator inserts the cartridge tape into computer_B's cartridge tape drive.
2. The computer_A administrator logs in to computer_A as `root`.
3. The computer_A administrator enters:

```
pkginstall -s computer_B:/dev/rmt/ctape1 all
```

Installing Software From a Remote Machine: An Example With NFS

The previous section, Installing Software From a Remote Machine: An Example With pkginstall, described how to use pkginstall(1M) to install software. An alternate method is to use the Network File System (NFS) utilities. Consider the same scenario as described in the previous section except this time NFS is used. (For full details on NFS, see the Network Administration guide.)

1. The computer_B administrator inserts the cartridge tape into computer_B's cartridge tape drive.
2. The computer_B administrator spools the packages onto computer_B:

```
pkgtrans ctape1 spool all
```

   This places the packages in the default spool directory, `/var/spool/pkg`.

3. The computer_B administrator advertises the directory containing the spooled packages to computer_A:

```
share -F nfs /var/spool/pkg computer_A
```

4. The computer_A administrator logs in to computer_A as `root`.
5. The computer_A administrator mounts the directory that the computer_B administrator just advertised onto computer_A's mount directory, `/mnt`:

```
mount -F nfs computer_B:/var/spool/pkg /mnt
```

6. The computer_A administrator installs the packages on to computer_A:

```
pkgadd -d/mnt
```

Troubleshooting Software Installation

The `pkgadd` command performs numerous checks as it installs a software package or set and, in a sense, performs troubleshooting for you. You will know a problem has occurred when

- `pkgadd` displays a descriptive message. Decide what actions to take based on the message. In most cases, `pkgadd` will prompt you for instructions and wait. If the `-l` (log mode) option is specified, no message will be displayed; rather, a log file containing error information is created.
- installation results in a partially installed package. When this happens, remove the partially installed package, then attempt to reinstall it.

Common Installation Errors

This section describes some common installation errors and illustrates that, in most cases, you will need to either answer a prompt or attempt to reinstall the package or set to correct an error.

Problem:	Accidental or purposeful termination of installation
Cause:	Break key used, program terminated, system difficulties
Fix:	Results in a partially installed package or set. Reinstall.
Problem:	Ran out of space on system during installation
Cause:	Inadequate space
Fix:	Results in a partially installed package or set. Reinstall after you have made space available.
Problem:	Read error occurs during installation
Cause:	Corrupt media, hardware problems
Fix:	Attend to media or hardware problems. Attempt reinstallation.
Problem:	Content verification error
Cause:	Temporarily out of space, bad media, file changed by editing before check made
Fix:	Attempt reinstallation.
Problem:	Incorrect diskette
Cause:	Installing diskettes out of sequence or using a diskette not part of the package or set being installed
Fix:	Will receive message requesting that the correct diskettes be inserted. Follow the instructions supplied by `pkgadd`.

Determining the Status of a Package or Set

The status of a package or set will be either fully or partially installed. A package or set is considered fully installed, and is noted as such in the installation software database, when the system is notified that all actions needed to install the package or set have executed successfully. If necessary actions have not taken place or have been incomplete or unsuccessful, then the package is considered partially installed and is noted as such in the database.

Procedure

Use the following procedure to determine the status of a package or set:

- Execute the following command:

  `pkginfo -l` *`pkginst`*

  to show whether the package or set was fully or partially installed. The status field in the display from this command gives this information.

Reinstalling a Package

Any time package or set installation results in partially installed software, attempt to reinstall the package or set. If you know the reason it was only partially installed, correct the problem first and then reinstall.

Follow these instructions when you have to reinstall a package or set:

NOTE: In some cases, you will have to remove a partially installed package before reinstalling it. If so, you will be notified when you execute `pkgadd`.

- Go to the section Software Package and Set Installation and use the procedure for interactive installation or the procedure for non–interactive installation presented there.

Checking Installation Accuracy

You may want to check the integrity of a package after it has been installed on your system. The `pkgchk` command determines whether an object has been modified by software or other actions since its installation.

`pkgchk` performs two kinds of checks. It checks file attributes (the permissions, ownership, and security attributes of a file, and major/minor numbers for block or character special devices) and the file contents (the size, checksum, and modification date of a file). By default, the command checks both the file attributes and the file contents. See the `pkgchk` manual page for complete details about available options.

Procedure

Use this procedure to check the integrity of a package after it has been installed on your system:

- Execute the `pkgchk` command with the `–n` option.

  The `–n` option indicates that volatile files should not be checked. The `–n` option should be used for most post–installation checking.

  Example: the command line for checking a package named `pkgA` would look like this:

  ```
  pkgchk –n pkgA
  ```

  NOTE: If `pkgA` is a SIP, then the SIP itself will be checked, but the packages that are members of the set will not be checked. `pkgchk` checks packages, not sets of packages. If you want to check the packages that comprise a set, run `pkginfo` to determine the name of each package in the set, and then run `pkgchk` for the individual packages.

  - You can name more than one package on the command line by separating the package names with spaces.
  - You can check specific pathnames, instead of all components of a package, by using the `–p` option to name the paths you want to check. If `pkgchk` is not given any name at all, meaning no package names or pathnames, then it checks the entire contents of a computer. You can name more than one pathname as long as the names are separated by commas (with no white space).

  NOTE: Packages must be identified by their package identifier. All instances of a package can be requested by adding .* to the package abbreviation. If you use the asterisk (*) in your command line, enclose the command line in single quotes or insert a backslash (\) before the asterisk to prevent the shell from interpreting the * character. The `PKGINST` field on the `pkginfo` display shows the package identifier.

Checking Against the pkgmap File

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

Procedure

Use this procedure to compare the file attributes and contents of the installed package against the installation software database:

- To compare the current integrity of a package to its integrity at the time it was originally installed, execute the `pkgchk` command with the `-m` and `-e` options to specify the original description files.

  Example: if `pkgA` is mounted or spooled in the `/install` directory, the following command would be used.

  ```
  pkgchk -m /install/pkgA/pkgmap \
   -e /install/pkgA/pkginfo
  ```

  (This command should be entered on one line. It is shown here on two lines for the sake of legibility.)

Correcting Differences While Checking Accuracy

Procedure

Use the following procedure to correct differences while checking accuracy:

> **NOTE:** Specify the `-n` option so that no attempt is made to correct volatile files, which may have undergone legitimate changes as a result of system activity, and which should not be set back to their original state.

- To correct file attributes when discrepancies are found, invoke `pkgchk` with the `-f` and the `-n` options.

  Example:

  ```
  pkgchk -n -f mypkg
  ```

  attempts to correct any differences between the package components and the installation database or, if the `-m` and `-e` options have been used, between the components and the original `pkgmap` and `pkginfo` files.

Displaying Information about Installed Packages or Sets

You can display information about installed packages or sets by using the `pkginfo` command. It has a number of options that allow you to customize both the format and the contents of the display.

You can request information about any number of package instances if each name is separated by white space.

Multiple Instances of the Same Package

More than one instance of a package can be installed on your system. The first instance of a package installed on a system is identified by the package abbreviation. If there are subsequent installations, `pkgadd` assigns a numerical suffix to the package abbreviation, beginning with `.2`.

When asked for *`pkginst`* in any procedure in this chapter, use the package identifier or SIP (for a set). Remember that when you have only one instance of a package on a system, the package identifier is the package abbreviation.

To execute a single command for all instances of a package, specify the abbreviation of the desired package, followed by an asterisk (*). If you use the asterisk in your command line, enclose the command line in single quotes or insert a backslash (\) before the asterisk. For example, if you want to display information about all instances of package mypkg, run the pkginfo command as follows:

```
pkginfo mypkg.\*
```

If you want to display information about only one instance of this package, specify the specific package identifier, as, for example:

```
pkginfo mypkg.3
```

Displaying Information About All Installed Packages

When pkginfo is executed without options or package names, it displays the category, package instance, and package name of all packages (but not SIPs) that have been completely installed on your system.

Procedure

Use the following procedure to display information about all installed packages:

- Execute this command:

  ```
  pkginfo
  ```

  The output of this command is organized by categories. (Categories are defined by the package developer.)

  Example output:

  ```
  system          int       Installation Utilities
  system          backup    Backup/Restore Utilities
  application     pkgA      Package A
  application     pkgA.2    Package A
  application     anpkg     Another Package
  Custom Installed Packages:
  acad            yes       Entire AutoCad package
  $
  ```

  In this example, pkgA.2 is the package identifier for the second installed version (instance) of pkgA.

  An instance is given the lowest extension available and so may not reflect the order of installation. For example, if pkgA.2 was deleted after pkgA.3 was installed, the next instance would be named pkgA.2.

  As shown in the last line of this example, the display for non–System V packages (for example, XENIX packages) is different. For these custom installed packages, the abbreviated package name is shown, followed by the word yes (which indicates that the package is installed), and the full package name.

Displaying Set Installation Packages (SIPs)

You can use the pkginfo command with the –c *category* option, and specify *category* as set to display information about Set Installation Packages (SIPs).

Procedure

Use the following procedure to display information about sets:

- Execute the following command:

  ```
  pkginfo –c set
  ```

  Example output:

```
set         admin          Administration Set
set         fnd            Foundation Set
set         graphics       Graphical Applications Set
set         util           Utilities Set
$
```

Customizing the Format and Contents of the pkginfo Display

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

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

You can use the following `pkginfo` options to specify the format of the display. Full details on these options are given on the pkginfo(1) manual page.

−x

Use the `−x` option to request the extracted format, which shows the package abbreviation, package name, package architecture (if available), and package version (if available).

Example:

```
$ pkginfo −x lp dfs
dfs              Distributed File System Utilities
                 (386) 4.1
lp               Printer Support
                 (386) 4.1
$
```

NOTE: If `pkginfo` is invoked to obtain information on set member packages located on tape media, all options are allowed since the information is readily available on the tape. But in the case of diskettes, if the SIP and each of its member packages are on separate disks, not all information about set member packages is available. In this case, the `pkginfo` command will only display a short listing.

−l

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

```
$ pkginfo −l lp
   PKGINST:  lp
      NAME:  Printer Support
  CATEGORY:  system
      ARCH:  386
   VERSION:  4.1
   BASEDIR:
    VENDOR:  USL
    PSTAMP:  4.1 02/17/92
  INSTDATE:  Feb 18 1992 07:52
    STATUS:  completely installed
     FILES:    386 installed pathnames
                28 shared pathnames
                 6 linked files
                41 directories
                31 executables
```

```
  11 setuid/setgid executables
2661 blocks used (approx)
```

See the pkginfo(4) manual page for complete details on these fields.

You can use the following `pkginfo` options to specify the format of the display. Full details on these and other options are given on the pkginfo(1).

−c
: Select packages based on their category membership.

−i
: Request that only completely installed packages be included in the display.

−p
: Request that only partially installed packages be included in the display.

Showing the Value of a Parameter

A number of pieces of information about each installed package is maintained on your system. Each piece of information can be individually retrieved if you know the parameter name and the package name.

The long output of `pkginfo` shows many of the available parameter names you can use to get information about a package.

See pkginfo(4) for complete details on the parameters.

Procedure

Use this procedure when you want to know the value of only one parameter of a package

- Execute this command:

  pkgparam −v *pkginst param*

  where *pkginst* is the package identifier and *param* is the name of the parameter you want displayed. *param* must match the parameter definition; in most cases, this means it must be entered in all capital letters. You can name multiple parameters when executing `pkgparam` as long as they are separated by white space. The `−v` option requests the verbose format, which shows the parameter name and its value.

  Example: the following example displays the value of the `NAME` parameter.

  ```
  $ pkgparam −v lp NAME
  NAME='Printer Support'
  $
  ```

Setting Up and Administering an Install Server

This section describes setting up an Application Server (AS) to be an Install Server. An Install Server allows you to "stage" software so that other systems in your network can perform a network installation of UnixWare or of specific packages.

NOTE: Licensing terms and conditions may limit the distribution and installation of software. Before performing the procedures in this section, local policies must be in place to prevent violation of the licensing terms. Contact your system administrator to ensure that your planned use of these procedures are in accordance with those policies.

The following procedures are provided to set up and administer the UnixWare network installation capabilities:

- Enabling and Disabling an Install Server
- Adding Products and Packages to an Install Server
- Removing Products or Packages from an Install Server
- Adding a Driver to the Network Installation Utilities Diskette

Once an operating system product is added to an Install Server and the Install Server is enabled, you can use the Install Server to perform a network installation of additional Personal Edition (PE) or Application Server (AS) systems on your network. For details on performing a network installation of UnixWare, see the *Installation Handbook.*

Also, once an add−on package is added to an Install Server and the Install Server is enabled, other systems in your network can install these packages from your Install Server. For details on installing these packages, see the the chapter *Installing and Sharing Applications* in the book System Owner Handbook.

Before Setting Up an Install Server

Before you set up and administer an Install Server, the following must be done:

- Either the Internet Utilities (`inet`) package or NetWare Networking (`nwnet`) package must be installed on your Application Server. These packages are provided with the Application Server.
- The Application Server must be configured for Transmission Control Protocol/Internet Protocol (TCP/IP) networking, Internetwork Packet eXchange/Sequenced Packet eXchange (IPX/SPX) networking, or both.

  NOTE: If you are configuring the Application Server for network installation over a TCP/IP network, you can simplify the network installations that are performed from this Install Server by properly configuring an Internet Boot Protocol (BOOTP) server on your network.

  To do so, on the BOOTP server, edit the `/etc/bootptab` configuration file and set `:sa=`*`IP`*`:` (where *`IP`* is the Install Server Internet Protocol address).

  For more information see bootptab(4) or the section Setting Up a BOOTP Server in the chapter *Setting Up TCP/IP* in part *TCP/IP Administration* in the book *Network Administration.*

- The Server Utilities package (`server`) must be installed on your system. This package is provided with the Application Server.
- If an external CD−ROM or tape drive is going to be used to load the packages, this drive must be configured for use by the system.
- To administer the Install Server through the UnixWare Desktop, you must have Setup Network Installation permissions (these permissions are assigned to the system owner by default). For details, see the section *Using User Setup* in the chapter *Configuring Your User Environment* in the book System Owner Handbook.

Commands for Administering an Install Server

The preferred method for setting up and administering an Install Server is to use the UnixWare Desktop, as described in this section. However, you can also use the `installsrv` command. For details, see the installsrv(1M) manual in this section. However, you can also use the `installsrv` command. For details, see the installsrv(1M) manual page. The pkgcat(1M), pkgcopy(1M), pkglist(1M), and pkginstall(1M) commands provide additional Install Server set−up and administration capabilities.

Enabling and Disabling an Install Server

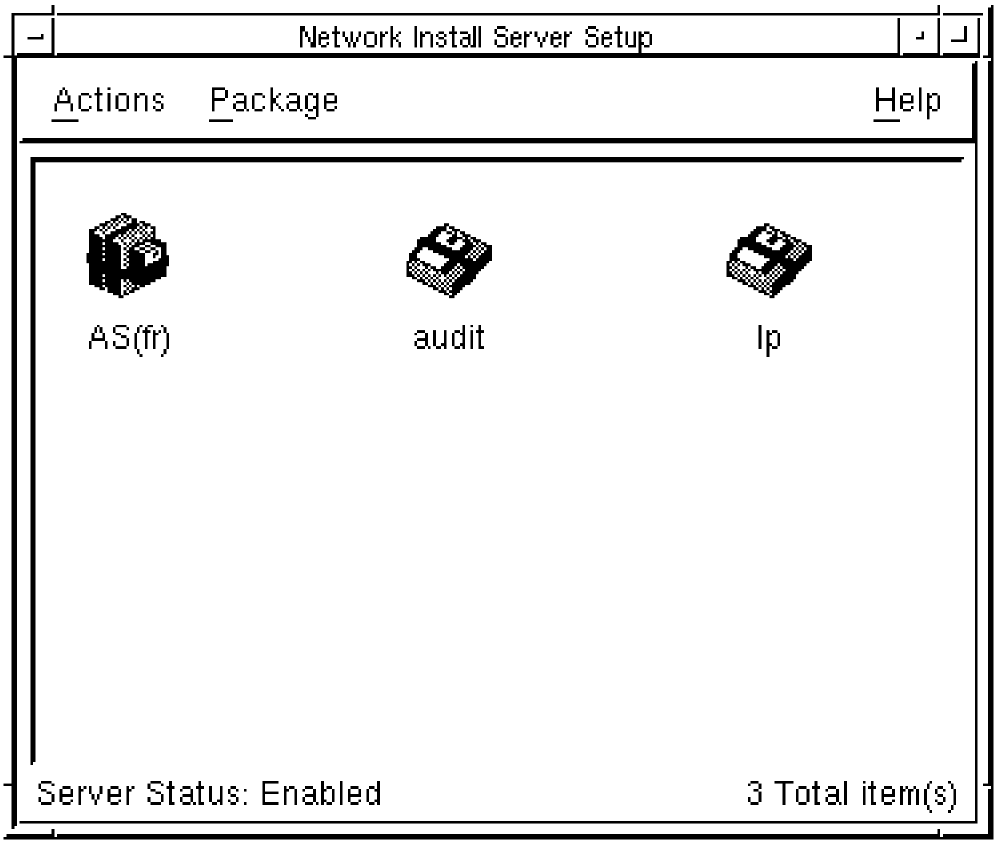
You can enable or disable an Install Server at any time. When an Install Server is enabled, other systems in the network can install an operating system product (Personal Edition or Application Server) or add−on packages that are staged on the Install Server. When an Install Server is disabled, other systems can not access the products or packages that are staged on this Install Server. This allows you to control when network installation of products and packages from your Install Server can be performed.

Procedure

To enable or disable an Install Server, do the following:

1. From the UnixWare Desktop, double−click on the Admin_Tools icon. The Admin_Tools menu is displayed.
2. Double−click on the Install_Server icon.

   The Network Install Server Setup window is displayed:



   An icon is shown for each product or package that is currently "staged" on the system (available to other systems for installation over the network).

3. Click on Actions in the window header. Three options are displayed:

- Select Enable to enable the Install Server.
- Select Disable to disable the Install Server.
- Select Exit to exit the Network Install Server Setup window.

Adding Products and Packages to an Install Server

To perform a network installation of UnixWare you must configure an Application Server as an Install Server. Once an Install Server is configured with the Personal Edition or Application Server, you can install UnixWare across the network from that server by following the instructions in the *Installation Handbook.*

An Install Server can also be configured to install additional packages beyond those provided in the Personal Edition or Application Server products. These products can then be installed by following the instructions in the chapter *Installing and Sharing Applications* in the book System Owner Handbook.

Several options are provided when setting up an Install Server, including:

- You can stage the software from media or from a previously configured Install Server on your network. If staging the Personal Edition, at least 40 MB of space must be available in `/var/spool/dist`; if staging the Application Server, at least 60 MB of space must be available in `/var/spool/dist`.
- You can either store the software on your system or create a link to a mountable CD−ROM. When storing a remote CD−ROM, system space is not used for storing copies of the packages; however, the CD−ROM drive is not available for other uses.

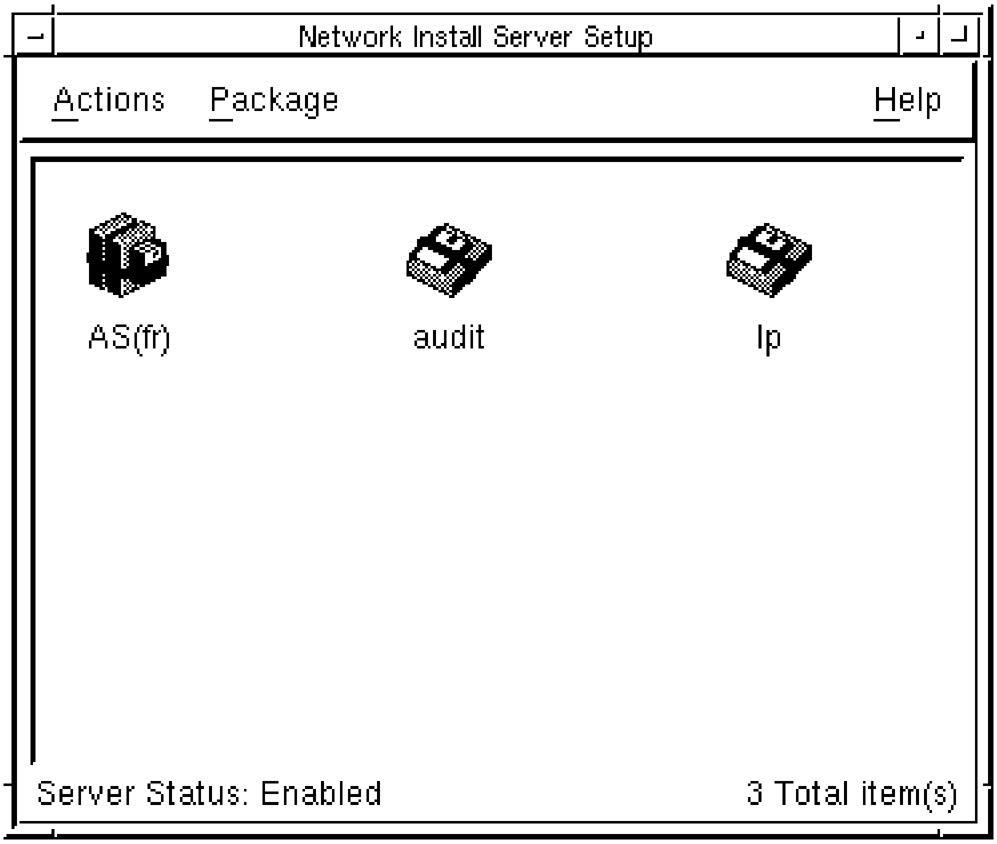
Procedure

To configure an Application Server as an Install Server, do the following:

NOTE: You may want to temporarily disable the Install Server before performing this procedure and then enable the Install Server after completing this procedure. This prevents the unlikely possibility that someone encounters problems because they tried to install a product before it was completely staged on the Install Server. For details on enabling and disabling an Install Server, see the previous section, Enabling and Disabling an Install Server.

1. From the UnixWare Desktop, double−click on the Admin_Tools icon. The Admin_Tools menu is displayed.
2. Double−click on the Install_Server icon.

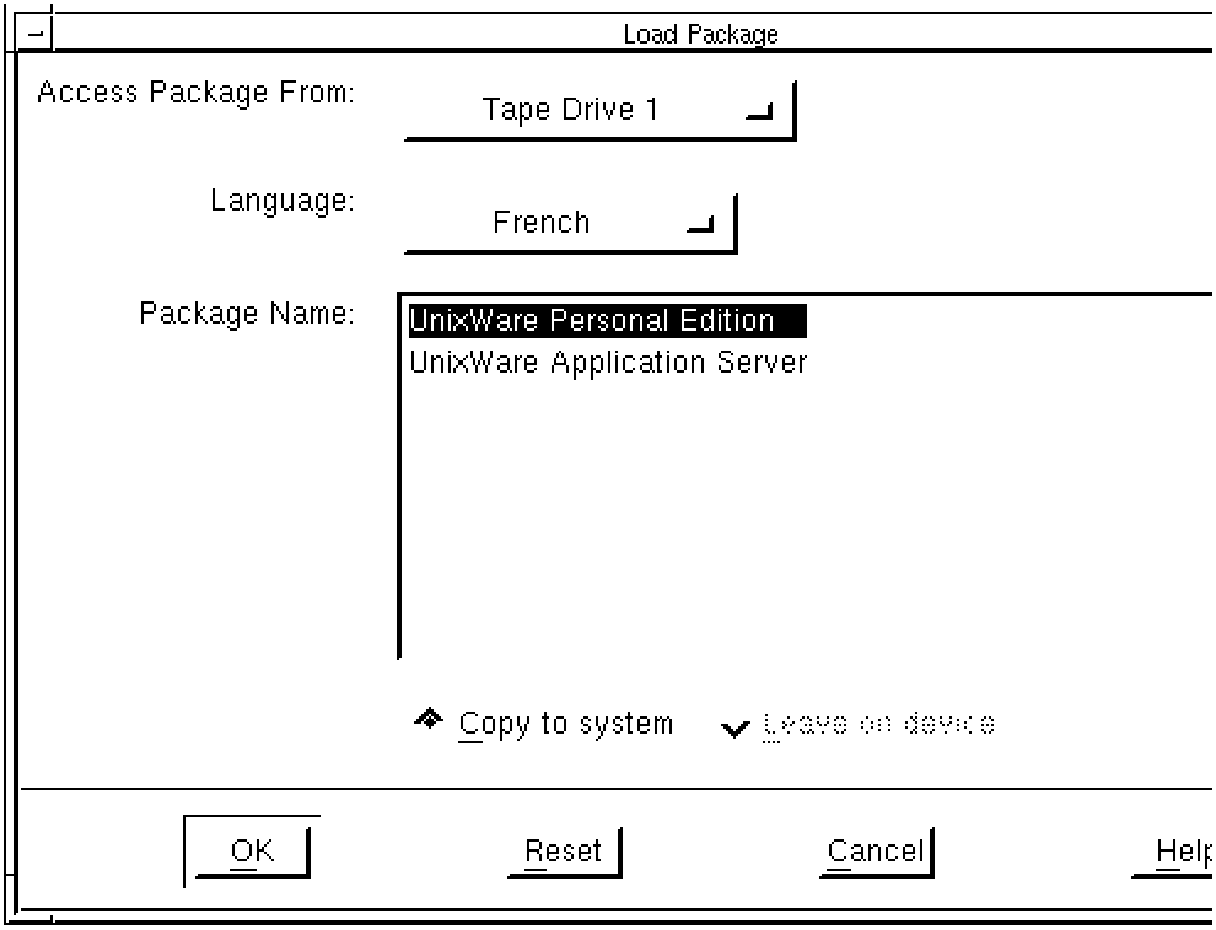
   The Network Install Server Setup window is displayed:



An icon is shown for each product or package that is currently "staged" on the system (available to other systems for installation over the network). For example, if this procedure was previously run to stage the Application Server, then an icon is shown for that product.

3. If the product or package(s) you want to load onto the Install Server are provided on media (CD−ROM, cartridge tape, or diskette), insert the media.
4. Click on Packages in the window heading; then, from the pop−up window:
   - If you want to add the Personal Edition or Application Server to your Install Server, select Load System.
   - If you want to add additional packages to your Install Server, select Load Add−On Package.

The Load Package window is displayed:



NOTE: The Language: field is displayed when staging the Personal Edition or Application Server on your Install Server. This field is not displayed when staging individual packages.

5. Select where to load the packages from.

   The Access Package From: field displays the location of the product or package you are copying. To select a location:

   A. Click on the rectangle in the right−most portion of the Access Package From: field. Depending on the hardware configured on your system, some or all of the following choices are displayed:

      - Floppy Drive 1
      - Floppy Drive 2
      - Tape Drive 1
      - Tape Drive 2
      - CD−ROM Drive 1
      - CD−ROM Drive 2
      - Another Install Server

For example, if your system does not have a CD−ROM drive then CD−ROM Drive 1 and CD−ROM Drive 2 are not displayed.

B. Use the mouse to select the location and then press **<ENTER>**.

C. If you choose Another Install Server, then a field is displayed to the right of the Access Package From: prompting for the name of the other Install Server. Type in the name of the other Install Server.

The products and/or packages that are available for staging on your Install Server are displayed.

6. If you are staging the Personal Edition or Application Server, choose the language.

   The Language: field allows you to use an Install Server to stage versions of the Personal Edition and Application Server for more than one language. To select a language:

   A. Click on the rectangle in the right−most portion of the Language: field. The following choices are displayed:

      - Default (C locale, including English)
      - German
      - Spanish
      - French
      - Italian
      - Japanese

   B. Use the mouse to select the location and then press **<ENTER>**.

7. Click on the product or package that you want to stage.
8. Select whether to copy or mount the product/package:
   - If you want to copy the product or package to your Install Server, click on Copy to system.
   - If you want to mount the installation media, click on Leave on `media` where `media` is the installation location you selected in Step 5.
9. Set up the Install Server by clicking on OK. (Click on Reset if you want to redo your selections; or click on Cancel to exit this menu without staging the software on your Install Server.)
10. Exit the Install Server.

    From the Network Install Server Setup menu, click on Actions in the window heading; then from the pop−up window, click on Exit.

Removing Products or Packages from an Install Server

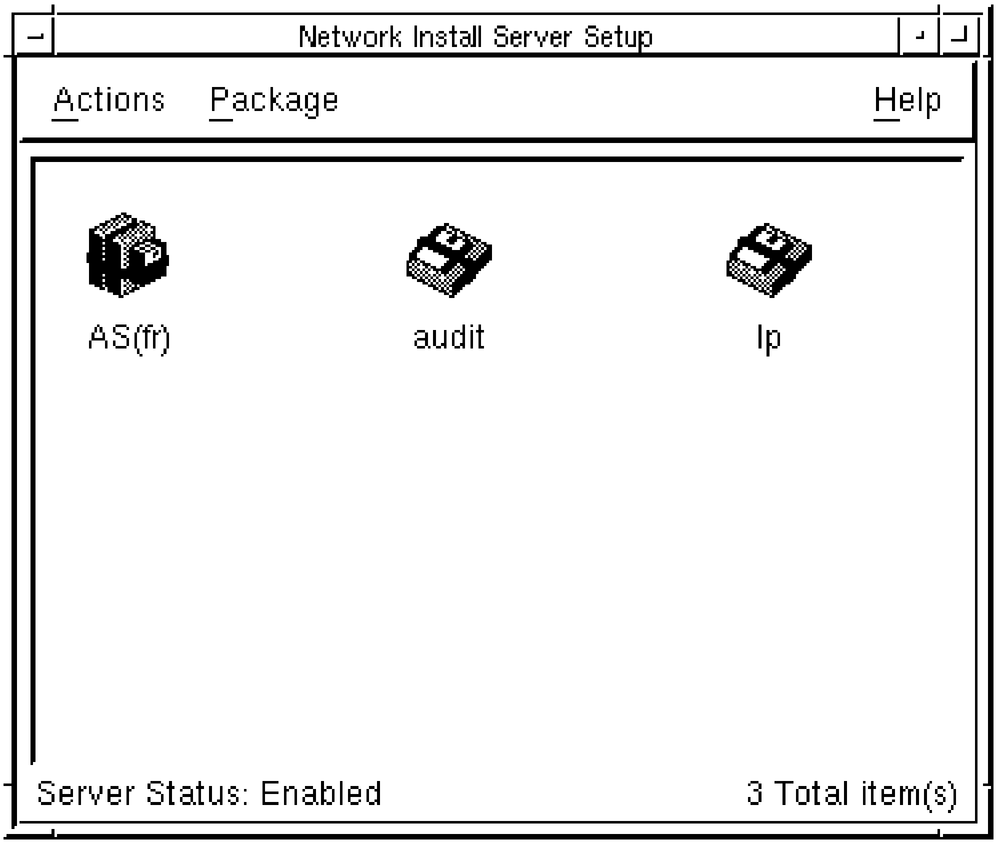
You can remove products or packages that were previously staged on your Install Server.

Procedure

To remove a product or package from an Install Server, do the following:

1. From the UnixWare Desktop, double−click on the Admin_Tools icon. The Admin_Tools menu is displayed.
2. Double−click on the Install_Server icon.

   The Network Install Server Setup window is displayed:



   An icon is shown for each product or package that is currently "staged" on the system (available to other systems for installation over the network).

   NOTE: If you want to view additional information about a product or package before deciding whether to remove it, highlight the package or product, click on Packages in the window heading, and then select Info.

3. Highlight the package or product that you want to remove.
4. Click on Packages in the window heading; then, from the pop−up window, click on Remove.
5. Exit the Install Server.

   From the Network Install Server Setup menu, click on Actions in the window heading; then from the pop−up window, click on Exit.

Adding a Driver to the Network Installation Utilities Diskette

Remote sites installing the UnixWare PE or AS over the network will need the network installation utilities diskette. This diskette contains the supported drivers that allow machines with the appropriate network interface card (NIC) to install UnixWare from the install server.

Certain configurations require drivers other than those contained in the network installation utilities diskette. UnixWare allows you to create a network installation utilities diskette containing the needed driver using the `mknetflop` command. See mknetflop(1) for more information on this command.

What You Will Need

To be able to create the network installation utilities diskette with the new driver, you will need the following:

- original network installation utilities diskette
- driver for the NIC (either on–line or from the NIC IHV diskette provided by the hardware vendor)
- writable diskette which will become the new network installation utilities diskette

Procedure

To add a driver to the network installation utilities diskette from a NIC IHV diskette, do the following:

1. Type the `mknetflop` command with the appropriate parameters:

   ```
   mknetflop –s diskette1  –t diskette1
   ```

   where `–s diskette1` is the source from which the NIC driver and configuration files are read. `–t diskette1` specifies the target from which the original network installation utilities diskette is read and to which the new network installation utilities diskette is written. Both parameters may be device identifiers for a diskette device or in this case, device aliases. They do not have to be the same diskette drive (for a multiple–drive system).

2. When prompted, insert the NIC IHV diskette in the appropriate diskette drive (in this example, drive 1). Then press the ENTER key.

3. When done reading the driver files, the system will prompt you to remove the NIC IHV diskette, insert the original network installation utilities diskette in the diskette drive, and press the ENTER key.

4. The system will then prompt you to remove the original network installation utilities diskette, insert the new diskette in the diskette drive, and press the ENTER key.

5. Upon completion of the process, a new network installation diskette is created with the new driver fully supported from the installation menu. This diskette can be used to perform network installations of UnixWare with the new NIC.

To add a driver to the network installation utilities diskette from a new NIC driver found in a directory (for example, `/home/user1/new_driver`) on your UnixWare system, do the following:

1. Type the `mknetflop` command with the appropriate parameters:

   ```
   mknetflop –s /home/user1/new_driver  –t diskette1
   ```

   where `–s /home/user1/new_driver` is the directory containing all the files needed by idinstall(1M) to build a driver for the new NIC. `–t diskette1` specifies the target from which the original network installation utilities diskette is read and to which the new network installation utilities diskette is written.

2. When prompted, insert the original network installation utilities diskette in the diskette drive, and press the ENTER key.

3. The system will then prompt you to remove the original network installation utilities diskette, insert the new

diskette in the diskette drive, and press the ENTER key.

4. Upon completion of the process, a new network installation diskette is created with the new driver fully supported from the installation menu. This diskette can be used to perform network installations of UnixWare with the new NIC.

For more information on what can be specified with the `–s` and `–t` parameters of `mknetflop`, see the mknetflop(1) manual page.

Storing Packages Without Installing Them

When you store a package, you copy its components directly from an installation medium to a spool directory without invoking any installation actions, such as running installation scripts or updating the installation software database.

Spooling a Package

Spooling copies a software package from the installation medium (without installing it) to a directory on your system for later installation.

You cannot spool a set directly. Rather, use `pkginfo` to determine which packages are members of the set, verify that you have sufficient disk space to spool to, and then spool the package(s).

Procedure

Use this procedure to store packages without installing them:

- Execute this command:

  `pkgadd [`*`other_options`*`] –s `*`spool_dir_name pkgname`*

  Example: to copy a package named `mypkg` from diskette drive `diskette1` to a spool directory named `/var/temp/spooldir`, run

  ```
  pkgadd –d diskette1 –s /var/temp/spooldir mypkg
  ```

  When you follow the `–s` option with the word `spool`, `pkgadd` copies the package into the default spool directory (`/var/spool/pkg`).

  NOTE: You must use a package identifier with a numerical suffix if multiple versions reside on the installation medium. When there is only one version of a package on a medium, the package identifier is the package abbreviation without a suffix. The package identifier suffix defines the package instance only on that particular medium. A new suffix will be assigned when this package is installed on your system. To find out what instances are available on a medium, run pkginfo –d *`device`*

Checking the Accuracy of a Spooled Package

You can check the accuracy of a spooled package instead of an installed package with the `pkgchk` command.

The checks made for a spooled package are limited because not all information can be audited until a package is installed.

Procedure

Use this procedure to check the accuracy of a spooled package:

- Execute the following command:

  `pkgchk -d` *`spool_dir_name pkgname`*

  where *`spool_dir_name`* is the directory into which the package was spooled (or the device onto which it was spooled), and *`pkgname`* is the name of the package you want to check.

  Example:

  `pkgchk -d spooldir pkgA`

  looks in the spool directory `spooldir` and checks the accuracy of a package named `pkgA`.

  > **NOTE:** If `pkgA` is a SIP, then the SIP itself will be checked, but the packages that are members of the set will not be checked.
  >
  > `pkgchk` checks packages, not sets of packages. If you want to check the packages that comprise a set, run `pkginfo` to determine the name of each package in the set, and then run `pkgchk` for the individual packages.

Displaying Information About Spooled Packages

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.

Procedure

Use this procedure to display information about spooled packages:

- Execute this command:

  `pkginfo -d` *`spool_dir_name`*

  To show information in the default format for all the packages in the spool directory `/opt/spooldir`, run

  `pkginfo -d /opt/spooldir`

Removing a Spooled Package

You can remove a package that has been spooled.

Procedure

Use this procedure to remove a spooled package:

> **NOTE:** If you name only the spool directory (but no packages), all spooled packages will be removed from the named directory.

- Execute this command:

  `pkgrm -s` *`spool_dir_name pkgname`*

  To remove `pkgA` from the spool directory `/opt/spooldir`, run

```
pkgrm −s /opt/spooldir pkgA
```

Package or Set Removal

The `pkgrm` command removes both fully and partially installed packages or sets from the system. It attempts to return the system to the same state in which it was running before the package or set was installed.

Removing a Package or Set

A package or set may need to be removed because you are preparing to install a new version of the package. It is good practice to remove the old package before you begin installing the new one.

You may also need to remove a partially installed package. Perhaps the system crashed while you were installing the package and you are not sure how much of the installation has been completed. It is good practice to remove the partly installed package before you begin re−installing it.

Procedure

Use these instructions to remove a software package or SIP from your system:

- Enter the following command:

  `pkgrm pkgname`

  where *`pkgname`* is the name of the package you want to remove.

  If *`pkgname`* is the name of a SIP, all packages that are members of the set, and the SIP itself, are removed in reverse order from which they were installed.

- To remove multiple packages (which are not members of a set) with one command, separate package names with white space, as follows:

  `pkgrm pkgA.3 mypkgB mypkgC.2`

  NOTE: Packages must be identified by their package instance, while sets must be identified by their SIP. All instances of a package can be requested by adding an asterisk (*) to the end of the package abbreviation. If you use the asterisk (*) in your command line, enclose the command line in single quotes or insert a backslash (\) before the asterisk to prevent the shell from interpreting the * character. The `PKGINST` field on the `pkginfo` display shows the package identifier.

- To remove a package in non−interactive mode, use the `−n` option. If the package cannot be removed in this mode (because some interaction is required), `pkgrm` exits and removal fails.

Removing a Package or Set With an Alternative Admin File

Two of the parameters set in the admin file affect package or set removal. They are `runlevel` (which gives instructions for what to do when the run level is not met) and `rdepend` (which defines whether or not to check for other packages with dependencies on this package). You can have alternative admin files to assign different values to these two parameters.

`pkgrm` uses the admin file `default` unless you use the `−a` option to inform `pkgrm` that you want to use an alternative admin file.

Procedure

Use these instructions to remove a package or set that has an alternative admin file:

- Execute this command:

  `pkgrm -a` *`admin_file_name pkgname`*

  Unless you specify a full pathname after the `-a`, `pkgrm` looks for the admin file in the `/var/sadm/install/admin` directory.

  To remove a package or set named `mypkg` that uses an admin file named `myadmin`, execute the following:

  ```
  pkgrm -a myadmin mypkg
  ```

- To remove an individual package, you can specify the `check` admin file with the `-a` option.

  Example:

  ```
  pkgrm -a check pkginst
  ```

- To remove a set, use the `default` admin file. (The `default` file is used automatically.)

  Example:

  `pkgrm` *`sipname`*

  where *`sipname`* is the name of the set you want to remove, and the admin file `default` is used by default.

  NOTE: If a SIP is specified, all packages that are members of the set, and the SIP itself, are removed in reverse order from that in which they were installed.

Installing Add−On Software Through OA&M Menus

The system administration menus are only available if the Operations, Administration and Maintenance (OA&M) package is installed on your system. To access the menu for installing and removing software, type `sysadm software`. The following menu will appear on your screen:

Figure 5−2. Menu for Software Installation and Information Management

```
1   Software Installation and Information Management
check    - Checks Accuracy of Installation
defaults - Sets Installation Defaults
install  - Installs Software Packages
interact - Stores Interactions with Package
list     - Displays Information about Packages
read_in  - Stores Packages Without Installing
remove   - Removes Packages
```

The following table shows how the tasks listed on the `software` menu correspond to the shell commands discussed throughout this chapter.

Task to Be Performed	`sysadm` Task	Shell Command

Set installation defaults	`defaults`	vi(1) admin(4)
Store interaction with packages or sets	`interact`	pkgask(1 M)
Install software package or set	`install`	pkgadd(1 M)
Check accuracy of installation	`check`	pkgchk(1 M)
Show installed packages or sets	`list`	pkginfo(1)
Store packages or sets (not install them)	`read_in`	pkgadd(1 M)
Remove packages or sets	`remove`	pkgrm(1M)

Details about the format of admin files are available in the System Files and Devices Reference. For details about `vi` and other commands, see the Command Reference.

Quick Reference to Software Management

- Setting installation defaults: Installation defaults are defined in an admin file. This file can be created with any editor and should contain a list of parameter definitions in the format of *`param=value`*.

  You can create this file in any directory; however, `pkgadd` normally looks in the `/var/sadm/install/admin` directory for admin files. If you put your admin file in this directory, you only need to supply the filename after the `-a` option when executing `pkgadd`. If you create your file in a different directory, you must supply the full pathname after the `-a`.

  The parameters that can be set in this file, along with their description and possible values, are shown below.

  - `basedir` (package relocation information) directory name or `$PKG` or `$PKGINST` or `ask`
  - `mail` (who should receive mail after installation) space−separated list of user IDs, defined as NULL (no mail sent), or not defined (mail is sent to root)
  - `runlevel` (run−level dependencies not met) options: `nocheck`, `quit`, `ask`
  - `conflict` (name conflict because a file or directory with the same name already exists) options: `nocheck`, `quit`, `nochange`, `ask`
  - `setuid` (check for set−UID or set−GID execution) options: `nocheck`, `quit`, `nochange`, `ask`
  - `action` (check security impact of package scripts) options: `nocheck`, `quit`, `ask`
  - `partial` (partially installed version exists) options: `nocheck`, `quit`, `ask`
  - `instance` (instance already exists) options: `quit`, `overwrite`, `unique`, `ask`
  - `idepend` (package dependencies not met at installation time) options: `nocheck`, `quit`, `ask`
  - `rdepend` (package dependencies not met at removal time) options: `nocheck`, `quit`, `ask`
  - `space` (disk space requirements not met) options: `nocheck`, `quit`, `ask`

- Storing interactions with a package or set:

  `pkgask -d device -r response pkginst`

  where *device* is the device on which the package is stored, *response* is the name of the file or directory in which your answers to questions from a package installation script will be written, and *pkginst* is the package identifier of the package, or the SIP of the set, to be installed. Use the file or directory created with this procedure in the procedure to install a package or set in non−interactive mode.

- Installing a software package:

  `pkgadd -a check -d device pkginst`

  where *device* is the full pathname of the device or directory on which the package is stored, and *pkginst* is the package identifier of the package to be installed. *device* can also be the device alias. You may also specify `-a check` if you want to install the package interactively.

- Installing a software set:

  `pkgadd -d device pkginst`

  where *device* is the full pathname of the device or directory on which the package is stored, and *pkginst* is the SIP of the set to be installed. *device* can also be the device alias. The `default` admin file is automatically used.

- Installing a software package or set in non−interactive mode:

  `pkgadd -n -d device pkginst`

  where *device* is the name of the device on which the package is stored, and *pkginst* is the package identifier of the package or the SIP of the set to be installed. The `-n` option requests the non−interactive mode.

- Installing a software package in non−interactive mode with a response file:

  `pkgadd -n -d device -r response pkginst`

  where *device* is the name of the device on which the package is stored, *pkginst* is the package identifier of the package, or the SIP of the set, to be installed, and *response* is the full pathname of the response file to be used during the installation process.

- Installing a software package or set and using an alternative admin file:

  `pkgadd -d device -a adminfile pkginst`

  where *device* is the name of the device on which the package is stored, *pkginst* is the package identifier of the package, or the SIP of the set, to be installed, and *adminfile* is the name of the alternative admin file to be used. pkgadd looks in the `/var/sadm/install/admin` directory for *adminfile* unless you supply a full pathname.

- Checking the installation accuracy of an installed package:

  `pkgchk -n pkginst`

  where *pkginst* is the package identifier of the package you want to check. You can name more than one package ID if the names are separated by white space. (If no package name is supplied, the entire contents of the computer will be checked.)

- Checking the installation accuracy of a specific pathname:

  `pkgchk -n -p pathname`

  where only *pathname* will be checked. You can name more than one pathname if the names are separated by commas.

- Checking the installation accuracy of only the file contents or only the file attributes of a package:

  `pkgchk [-a|-c] pkginst`

where `-a` will check only the file attributes of a package and `-c` will check only the file contents. *`pkginst`* is the package identifier of the package to be checked.

- Checking the installation accuracy of a file against the original description files instead of the installation software database:

  `pkgchk -m /install/`*`pkginst`*`/pkgmap -e /install/`*`pkginst`*`/pkginfo`

  where `-m` and `-e` check the package instance, *`pkginst`*, against the original `pkgmap` and `pkginfo` files from the package's distribution medium that are mounted or spooled at `/install`.

- Correcting file attributes as they are checked:

  `pkgchk -n -f` *`pkginst`*

  where *`pkginst`* is the package identifier of the package to be checked. When the `-f` option is used, the command attempts to correct any differences in the file attributes between the package and the installation software database.

- Spooling a package:

  `pkgadd -d` *`device`* `-s` *`spooldir pkginst`*

  where *`device`* is the name of the device on which the package is stored, *`spooldir`* is the name of the directory into which the package is to be spooled, and *`pkginst`* is the package identifier of the package to be installed.

- Checking the installation accuracy of a spooled package:

  `pkgchk -d` *`spoolarea pkginst`*

  where *`spoolarea`* is the name of the directory into which the package was spooled (or the device onto which it was spooled), and *`pkginst`* is the package identifier of the package. You can name more than one package if the names are separated by white space.

- Showing information about installed packages or sets:

  `pkginfo` *`pkginst`*

  where *`pkginst`* is the package identifier of the package for which you are requesting information. You can name any number of instances separated by spaces. The display shows the package category, package instance, and package name for the each instance requested. If *`pkginst`* is a SIP, information about its member packages is displayed, if available. Executing `pkginfo` without naming a package instance displays information about all fully installed packages (but not SIPs) on your system.

- Showing information about installed SIPs:

  `pkginfo -c set` *`pkginst`*

  where `-c` specifies the category `set`, and *`pkginst`* is the SIP of the set for which you are requesting information.

- Showing information about spooled packages or packages on a particular device:

  `pkginfo -d` *`device`*

  where *`device`* is the name of a device, directory, or spool directory. Using the `-d` option with `pkginfo` displays information about all packages on that device or in that directory.

- Showing information about a spooled SIP or SIPs on a particular device:

  `pkginfo -c set -d` *`device`*

  where `-c` specifies the category `set`, and *`device`* is the name of a device, directory, or spool directory. Using the `-d` option with `pkginfo -c set` displays information about all sets on that device or in that directory.

- Removing a spooled package:

  `pkgrm -s` *`spooldir`* `[`*`pkginst`*`]` where *spooldir* is the name of the spool directory and *pkginst* is the name of the package to be removed. If no package identifier is supplied, all packages spooled in the named directory will be removed.

- Customizing the format of a `pkginfo` display (example 1):

  `pkginfo [-x|-l] pkginst`

  where *pkginst* is the package identifier of the package, or packages, for which you are requesting information. The `-x` option requests the extracted display consisting of the package instance, package name, package architecture (if available), and package version (if available). The `-l` option requests the long display that consists of all available information about a package. Executing `pkginfo` without the `-x` or `-l` option causes the information to be displayed in the default format that shows the category, package instance, and full package name.

- Customizing the contents of a `pkginfo` display (example 2):

  ```
  pkginfo [-c category] [-i|-p] [-n] [-a arch]
      [-v version] pkginst
  ```

  where the options define the contents of the `pkginfo` display as follows:

  - `-c` *category* includes all packages in the category defined. Specify `-c set` for information about sets.
  - `-i` includes only fully installed packages.
  - `-p` includes only partially installed packages.
  - `-a` *arch* includes all packages with the specified architecture.
  - `-v` *version* includes all packages with the specified version.

  pkginst is the package identifier of the package(s) for which you are requesting information. The `-a`, `-l`, and `-v` options will not work for sets whose member packages span over several diskettes.

- Showing the value of only one parameter:

  `pkgparam -v pkginst param`

  where *pkginst* is the package identifier and *param* is the name of the parameter, or parameters, you want displayed. Separate multiple parameters by white space. `-v` shows the parameter value with a label (for example, `BASEDIR='/opt/pkgAdir'`). Without the `-v` option, only the parameter value is shown.

- Storing a package without installing it:

  `pkgadd -d device -s spooldir pkginst`

  where *device* is the name of the device on which the package is stored, *spooldir* is the name of the directory into which the package is to be read, and *pkginst* is the package identifier of the package to be read.

- Removing a package or set:

  `pkgrm pkginst`

  where *pkginst* is the package identifier of the package or the SIP of the set to be removed. When *pkginst* is a SIP, all packages that are members of the set, and the SIP itself, are removed in reverse order from which they were installed.

- Removing a software package or set in non–interactive mode:

  `pkgrm -n pkginst`

  where *pkginst* is the package identifier of the package to be removed. The `-n` option requests the non–interactive mode.

- Removing a software package using an alternative admin file:

  `pkgrm -a admin pkginst`

where *`pkginst`* is the package identifier of the package to be removed and *`admin`* is the name of the alternative admin file. Use only the `default` admin file for set removal. `pkgrm` looks in the `/var/sadm/install/admin` directory for the admin file unless you supply a full pathname.