4D1-3.3.2 Maintenance Release and Installation Notes

Document Version 1.0

Document Number 007-3613-010

Technical Publications:

Lorrie Williams

Engineering:

Bob Green

Restricted Rights Legend

4D1-3.3.2 Maintenance
Release and Installation Notes
Document Version 1.0
Document Number 007-3613-010

Silicon Graphics, Inc.
Mountain View, California

Contents

1. Introduction

These maintenance release and installation notes contain information that is vital for properly updating your system to software release 4D1-3.3.2. They provide information on the following topics:

- software installation
- backup of your work before installing the update
- compatibility and configuration
- bugs that have been fixed
- changes to the software since the last maintenance release
- existing problems in the software and ways to work around them
- documentation errors

Silicon Graphics, Inc., provides a comprehensive product support and maintenance program. For further information, please contact your local service organization.

Urgent Notice

This maintenance release contains important bug fixes to the 4D1-3.3 software release. Unless you are specifically advised by your software vendor not to install the 4D1-3.3.2 release, Silicon Graphics, Inc., recommends that you install it.

The 4D1-3.3.2 release software can be installed only after you have successfully installed the 4D1-3.3 software.

2. Installing Software

This chapter explains how to install software on your IRIS-4D™ Series workstation or server. It covers:

- items to consider before you begin
- setting up the installation tools
- installing your software
- finishing up after you install new software
- troubleshooting installation problems

2.1 Read This BEFORE You Begin

This section contains important information explaining several items you should consider *before* you start to install software on your IRIS-4D Series workstation or server. Reading this section before you begin will help you avoid problems and confusion as you install your software.

To install new software, you must shut down your system (if it's running) and install special software installation tools. These tools include *inst* and a temporary operating system called the *miniroot*. **You MUST use the installation tools contained in the Installation Tape of the 4D1-3.3.2 set to install 4D1-3.3.2.**

Make sure your tape drive is clean. Most tape read errors are caused by dirty drives.

Important: Before you begin to install your new software, make a complete backup of your system. See your owner's guide for instructions on making backups.

Important: Before installing this maintenance release, check with the suppliers of any third-party software installed on your system to assure that their software continues to operate. Some third-party software looks for the specific release number and does not operate unless that number is an exact match.

Important: Although the installation tools provide an ''automatic'' and a ''manual'' method for software installation, use only the ''automatic'' mode for maintenance releases.

Important: Since this maintenance release contains patches for both standard system and optional product software, the maintenance tape must be reinstalled as a last step when you install a software product. The installation tools then make the proper changes to the existing software to correct the bugs listed in these release notes.

2.1.1 Finding Network Addresses

If you plan to install software from a remote tape drive or a directory on a remote system, *inst* might prompt you for the Internet Protocol (IP) network addresses of your system and, possibly, the name of the remote system you use for installation. *inst* looks to your */etc/hosts* file for these addresses. However, if the file doesn't exist, or if you are referencing a remote system not noted in your system's */etc/hosts* file, *inst* prompts you for the IP network addresses of the systems in question.

Note: If you want to install software remotely over a network through a gateway system, the systems must be able to communicate across the network. Check the file */usr/etc/inetd.conf* on the gateway system to make sure that the last column of the *bootp* entry is:

```
bootp -f
```

See Section 2.5.1, ''The *distcp* Command,'' your owner's guide, and the *Network Communications Guide* for more information about using the network.

Once your system is shut down to the PROM level to install software, you do not have ready access to these addresses. Therefore, before you begin to install software, find and record the IP network addresses of the workstations or servers directly involved with your installation.

You can get these addresses by looking at any */etc/hosts* file on your network. You are looking for four numbers separated by periods. For example, a typical network address might look like this:

```
192.35.73.96
```

If you have a workstation named solaris, either of the following commands should provide you with what you need:

```
grep solaris /etc/hosts
```

or if your system is running the network information service:

```
ypmatch solaris hosts
```

You should see something similar to this:

```
192.35.73.96     solaris.foo.com  solaris
```

Record the network addresses in case you need them.

2.1.2 Installing Software on Diskless Clients

Maintenance release software on diskless clients is very sensitive to the order in which it is installed. For both the share tree and the client tree, the maintenance tapes must be installed after the product images have been installed:

1. Install all required product tapes on the share tree.
2. Install the maintenance product on the share tree.
3. Install all required products on the client tree.
4. Install the maintenance product on the client tree.

During the installation of the Execution Only Environment from the share tree to the client tree, two error messages appear due to the inability of the software to find two files. These files were deliberately removed from the release, but the error messages could not be removed from diskless installations. You can ignore these messages and continue installation.

After you install the 4D1-3.3.2 maintenance product, you might see an error message concerning the inability of *mkfontdir* to execute properly. You can ignore this message.

2.1.3 Installing Earlier System Software Releases

Once you've installed the 4D1-3.3 system software, you cannot re-install a previous system software release (or portions of a previous release) without first completely removing IRIX™ 4D1-3.3. If, for some reason, you must re-install a previous system software release, make a complete backup of all user data, reboot your system with the distribution tape of the earlier release, and use the *clean* option in *inst* from that earlier release to erase all 4D1-3.3 system and user files. You can then restart the installation procedure from scratch. When you have finished re-installing the previous release, use the backup tapes you made to put the user data back on the system. Read the following section for important information on the compatibility of installation tools across different system software releases.

Note: The on-line installation history format is slightly different in Release 4D1-3.3 from what it was in previous releases. When you use the 4D1-3.3 installation tools, the on-line installation history will contain this new format. Once that format is installed, you cannot use older versions of the installation tools because they will not understand the new format. If it is necessary to install older software once your system has been updated to 4D1-3.3 or newer, use the new installation tools to do so.

2.1.4 Determining Which Installation Mode to Use

You can use either of two modes for installing software: *automatic* and *manual*. The easiest way to install software is to use the *automatic* mode. The *automatic* mode will replace all out-of-date subsystems with the current corresponding subsystems. It will not install subsystems that in the previous release were specifically selected not to be installed, unless those subsystems are considered mandatory. If it fails (for lack of disk space, for example) or if you want more control over your installation than the *automatic* mode provides, you should use the *manual* mode.

Software is divided into groups of related files called *subsystems*. The *manual* installation mode lets you preview a list of the available subsystems, the subsystems selected for installation, and the ones that have not been selected at all. The *manual* mode also provides other functions that you might need during installation. Use the *manual* mode if you need to:

- use less disk space than is required for installing the currently selected set of subsystems
- override the default subsystem selections
- verify *inst*'s default subsystem selections
- specify (or respecify) the installation source. The installation source is a place, such as a tape drive (device) or directory, in which components (such as software images and product descriptors) of a product reside. A product descriptor file is a list of the components in a product that *inst* uses to install that product.
- make new file systems, control file system mounting or unmounting, or perform additional network initialization tasks
- delete unnecessary subsystems

2.1.5 Using Special Characters in Installation

The software products you receive from Silicon Graphics, Inc., are divided into discrete components, which you must deal with before, during, and after installation.

A broad, internal division of a product, such as on-line manual pages, is called an *image*. An image is divided into *subsystems*, which are made up of related files. Each subsystem has a name that follows this format:

```
product.image.subsystem
```

For example, *ftn.man.fedgetut* is the name of the Fortran *edge* on-line manual page subsystem.

Using the *manual* mode, you can specify on which subsystems you wish to operate. Often, you will need to specify more than one subsystem at once.

You can specify a subsystem name individually or use a ''pattern'' to specify many at one time. A pattern is a subsystem name that contains shell-style metacharacters. These metacharacters allow you to match zero or more regular characters in the available subsystem name(s).

The metacharacters are:

?	matches one character
*	matches any combination of characters
[ ]	matches any enclosed characters or range of characters separated by a dash

Missing components in subsystem names are treated by the system as though asterisks (*) are in their places. The following list contains two hypothetical products, to help you see how special characters and missing components affect subsystem selection. Both products, *abc* and *xyx,* contain two images: one for manual pages (on–line documentation) and one for software.

2.2 The Installation Procedure

This section describes how to prepare your system for installation and use *inst* to install new software.

2.2.1 Setting Up the Installation Tools

To install software on an IRIS-4D Series workstation or server, you first need to set up the installation tools. The main component of this tool set is the *miniroot.* The *miniroot* is a small, temporary operating system that prevents *inst* from overwriting system files that would be active if the IRIX operating system were running.

When you start setting up the installation tools, you will have to bring the system down to the PROM level. Once the *miniroot* is set up, it automatically invokes *inst*, and you can then install new software. When you've finished installing your software and rebooting your system, IRIX overwrites the *miniroot* (which is stored in *swap*), and your newly installed software is ready for use.

In the event of a power failure during this procedure, rebooting the system returns you to the *miniroot* and *inst* by default. To quit out of *inst* after a power failure, you may select the *quit* option from any of the installation menus, and your original system configuration should be restored. If the system does not recover properly, see Section 2.3.1, ''Error Recovery.''

Warning: Some systems include a bank of eight switches on the front panel for use in certain debugging operations. If you have such a system, make sure that these switches all point away from the word OPEN before you bring your system down to the PROM level. If you change the switch position, you must reset the system and then reboot.

1. If your IRIS-4D Series workstation or server is not already running, turn it on and bring it to PROM menu. See your owner's guide if you need help with this. If you see this message:

   ```
   Press any key to restart.
   ```

   then press any key on the keyboard. If you see a message similar to this one:

   ```
   Starting the system, press <Esc> to stop.
   ```

   then press **<Esc>**.

2. If your system is already running, log in as *root* and then use *who*(1) to determine whether anybody else is logged into the system. If so, post a system-wide message with *wall*(1) asking them to log off while you install new software. Give people a few minutes to finish what they are doing and to log off.

   If you have not already done so, make a complete backup of the system. See your owner's guide or the *IRIS-4D System Administrator's Guide* for more information about making backups.

   Type this command to shut down the system:

   ```
   /etc/halt
   ```

3. If your system displays this message:

   ```
        Starting up the system...
   To perform system maintenance instead, press <Esc>.
   ```

   then press **<Esc>**.

   Depending on the type of IRIS-4D Series workstation or server you have, you see either the PROM Monitor prompt, which looks like this:

   ```
   >>
   ```

   or the System Maintenance Menu.

The conditions listed below require that you begin your installation at Section 2.2.3, ''Using the PROM Monitor'':

- You see the PROM monitor prompt >>.
- Your system has more than one tape drive (you need to specify which tape drive to use for installation).

If neither of these conditions is true, continue on to step 4.

4. If you plan to use the tape drive on your system, select 2, ''Install System Software.''

   If you plan to install software over a network (that is, from a remote system) and *your* system has a tape drive, select 5, ''Enter the Command Monitor,'' and type the following at the Command Monitor prompt:

   ```
   setenv notape 1
   ```

   then type **`exit`** to return to the System Maintenance Menu. Select 2, ''Install System Software,'' and go to step 5.

5. If you see this message:

   ```
   Insert the installation tape.
   ```

   go to step 6.

   If you see this message instead:

   ```
   Are you using a remote tape? (y/n):
   ```

   you may:

   - Type **`n`** if you plan to install software from your system's tape drive, in which case you see:

     ```
     Enter the name of the machine...
     ```

     Enter the name of your system and go to step 6.

   - Type **`y`** if you want to use the tape drive on a remote system. You see:

     ```
     Enter the name of the machine...
     ```

     Enter the name of the remote system and go to step 8.

- Type `n` if you want to install software from a directory on a remote system. You see:

  ```
  Enter the name of the machine...
  ```

  Enter the name of the remote system and the distribution directory in this format:

  `solaris:`*/dir_path*

  Replace `solaris` with the name of the computer from which you plan to copy your software. Replace *dir_path* with the path name of the directory where the software image resides. Press **`<enter>`**.

6. You see a message similar to this:

   ```
   Insert the installation tape, then press <enter>:
   ```

   Insert the 4D1-3.3.2 Maintenance Tape 1 tape, lock it in place, and then press **`<enter>`**.

7. You should see this message followed by several lines of dots that print across your screen as the installation software is loaded onto the system:

   ```
   Copying installation program to disk.
   ```

   After several minutes, the dots stop printing across the screen, and the system boot process begins. At this point, you may see some messages similar to this:

   ```
   xy1757 ctlr 3:missing
   ```

   ```
   enp0:missing
   ```

   These harmless messages may be ignored.

   When the system boots, you see the *inst* menu, which indicates that *inst* is running and that you may install your software. You see:

   ```
   Ready to install software.
    Choose an item, then press <enter>:
      1. Automatically install software
      2. Use manual installation features
      3. Help
      4. Quit
   ```

Go to Section 2.2.2, ''Using *inst*,'' for further instructions.

Note: If the system has trouble reading the tape, see Section 2.1.1, ''Finding Network Addresses.''

8. Continue with this step if you are installing from a remote tape.

Before you type anything, make sure the installation tape is locked in the remote system's tape drive. On the remote system, type:

```
mt rewind
```

The system uses the TFTP protocol to access remote tape devices. As of IRIX release 4D1-3.3, TFTP file accesses are restricted to certain directories by default. The tftp entry in the */usr/etc/inetd.conf* file on the remote system controls which directories can be accessed by other workstations. The default entry is:

```
tftp dgram udp wait guest /usr/etc/tftpd tftpd -s /usr/local/boot
```

which limits access to files in */usr/etc/boot* and */usr/local/boot.* You can eliminate this restriction if you are not concerned about security by editing *inetd.conf* and removing:

```
-s /usr/local/boot
```

from the tftp entry. In order to do an automatic installation with remote tape, where the machine with the tape drive is running 4D1-3.3 or later, you must remove the -s option on the *tftpd* line. If security concerns prevent this, all remote installs must be done manually, as detailed below. If you are concerned about security, you must change the default entry to read:

```
tftp dgram udp wait guest /usr/etc/tftpd tftpd -s /usr/local/boot /dev/nrtape
```

to allow access to the tape device by workstations doing a remote installation.

After any change to */usr/etc/inetd.conf*, you must signal the internet daemon, *inetd*, to reread the file. Issue the command:

```
killall 1 inetd
```

on the system with the remote tape drive.

9. You can start installing on your system once TFTP access to the remote tape device is assured. In response to the following:

   ```
   Enter the name of the machine with the tape drive:
   ```

   type the name of the remote system and the tape device name, then press **<enter>**. For example:

   ```
   solaris:/dev/nrtape/
   ```

   (The trailing / is required.) You will see:

   ```
   Copying installation program to disk...
   ```

   After a few minutes, the dots stop printing across the screen and you see the *inst* menu, which indicates that *inst* is running and that you can install your software. You see:

   ```
   Ready to install software.
    Choose an item, then press <enter>:
      1. Automatically install software
      2. Use manual installation features
      3. Help
      4. Quit
   ```

   Go to Section 2.2.2, ''Using *inst*,'' for further instructions. If you have a problem at this point, see Section 2.3.3, ''Remote Installation Failure.''

2.2.2 Using *inst*

Once *inst* is automatically invoked, installing software is quite straightforward. This section explains how to use *inst* to install new software on your IRIS-4D Series workstation or server.

1. *inst* displays the main installation menu on your screen.
2. If you are installing from a tape cartridge, insert the tape in the tape drive and lock it in place.

3. Choose installation mode from the menu. Remember, you can install most software automatically. This is the simplest way to install software.

   If *automatic* installation fails (because not enough space remains on the hard disk, for example), or you want interactive control over what you install on your hard disk, use the *manual* mode. See Section 2.1, ''Read This Before You Begin,'' to determine which installation mode you should use.

   Note: You can switch installation modes while you are using *inst*. This is useful when you want to install software from multiple sources.

   Enter the number, the name, or an abbreviation of the menu item you choose and then press **`<enter>`**.

   You can ask *inst* for *help* at any point during the installation process, except when software is actually being copied to the hard disk. Section 2.4, ''The Installation Menus,'' provides detailed information on the *help*.

   inst runs a series of pre-installation checks to determine which subsystems to install by default. If you are installing a product for the very first time, the default subsystems are predefined. If you are updating a product, the default subsystems are those that replace previously installed, corresponding subsystems.

   New software directly replaces corresponding subsystems already on the disk though they might have different names from their older counterparts. *manual* mode allows you to redefine which subsystems are to be installed.

   The pre-installation check tests to:

   - ensure that mandatory subsystems are selected
   - determine whether prerequisite subsystems are already installed
   - determine whether enough space exists in the file system to install all of the selected subsystems

   Note: If any of these pre-installation checks fail, you see a warning message and the installation stops. You can use the *manual* mode to resolve such problems and then continue installing your software.

If you are attempting to install from a remote tape or remote directory, *inst* might prompt you for network addresses. (See Section 2.1.1, ''Finding Network Addresses,'' for help with this.)

4. If you selected the *manual* mode, skip ahead to step 7 to continue with the installation.

   If you chose to install your software automatically, the software is copied from the source to the hard disk. *inst* notifies you as old versions of subsystems are being removed and new versions are installed.

   If *inst* has trouble getting software across the network, refer to Section 2.1.1, ''Finding Network Addresses,'' or Section 2.3.3, ''Remote Installation Failure'' (whichever is applicable to your situation).

   At the end of the software installation, *inst* checks for compatibility between the new software being installed and software already on the system. If any incompatibilities are found, *inst* notifies you by printing out the names of the incompatible subsystems. These incompatibilities must be removed or replaced before *inst* will let you *quit*. For information on resolving incompatibilities, type:

   ```
   help incompatible
   ```

   When the installation process is complete, you see:

   ```
   Done.
   Is there more software to install?
   ```

5. Indicate whether to install additional software.

   If you don't have any more software to install, type **`no`** and press **`<enter>`**.

   If you have more software to install, type **`yes`** and then press **`<enter>`**. You see:

   ```
   Insert the next tape, then press <enter>:
   ```

Insert the tape and press **<enter>**. You see the following menu again:

```
Choose an item, then press <enter>:

    1. Automatically install software
    2. Use manual installation features

    3. Help
    4. Quit
```

Repeat the installation procedure until you are finished installing your software.

Note: When you use *quit* to exit, *inst* checks for compatibility with existing software on the system. If any of the products are incompatible with the new software, *inst* notifies you of the incompatibility. Before you are allowed to quit *inst*, you must resolve the incompatibilities.

6. Restart the system.

   When all of your software is installed, you see:

   ```
   Please wait...
   Ready to restart the system.  Restart? [y, n]
   ```

   Type **y**, then press **<enter>**. The system reboots, and your new software is available for use.

   Next, you see a message reminding you to use *versions changed* to see which configuration files changed during installation. Go to Section 2.3, ''Finishing Up the Installation,'' for further instructions.

7. Continue with this step if you are installing your software using the *manual* mode.

   The general procedure for using the *manual* mode is outlined below, but can vary depending on your particular needs. You must be familiar with the various menus to perform a successful installation using the *manual* mode. See Section 2.4.1, ''Manual Installation Menus,'' for a complete explanation of each menu item function.

8. Use *admin* to perform any necessary file system operations, such as making new file systems, mounting, and unmounting.

2.2.3 Using the PROM Monitor

If your system displays the System Maintenance menu, choose 5, ''Enter Command Monitor,'' to get the PROM monitor prompt >>.

Follow the instructions that apply to your software installation in one of the following subsections:

- From a Local Tape Drive
- From a Remote Tape Drive
- From a Remote Distribution Directory

From a Local Tape Drive

To set up the *miniroot* from a local tape drive, follow these steps:

1. Put the installation tape in the tape drive.
2. Set the environment variable *tapedevice* to the name of your tape device by typing in the following command at the >> prompt and pressing **<enter>**. Replace *device_name* in the command below with the name of the device that corresponds to your tape drive. See Table 2-2 for standard tape drive device names.

   ```
   setenv tapedevice device_name
   ```

   You can use the *hinv* command to determine the SCSI number for your drive:

   ```
   hinv
   ```

   Table 2-2 lists the device names for QIC and SCSI tape drives. *n* is the controller number and *ID* is the ID number.

Type of Drive	Device Name
QIC tape drives	tpqic(*n*,*ID*)
SCSI tape drives	tpsc(*n*,*ID*)

   Table 2-2. Tape Drive Device Names

Refer to the documentation from the tape drive manufacturer to determine which type of drive you have.

3. Boot the standalone shell, *sash*, from the tape device. Replace *cpu* in the command below with the appropriate CPU type from Table 2-3.

```
boot -f ${tapedevice}(sash.cpu) --m
```

Model Number	CPU Type
4D/60	R2300
4D/50, 60T, 70, 80, 85	IP4
4D/120	IP5
4D/20, 25	IP6
4D/220, 240, 280, 320, 340, 380	IP7
4D/210	IP9

Table 2-3. CPU Types

If you encounter errors in booting *sash*, see the Section 2.3.2, "Trouble Reading a Local Tape."

This command copies the *miniroot* to your hard disk and boots your system from it. After about 10 minutes, you see this message:

```
Ready to install software.
  Choose an item, then press <enter>:
     1. Automatically install software
     2. Use manual installation features
     3. Help
     4. Quit
```

Now you can install your software. See Section 2.2.2, "Using *inst*," for further instructions.

From a Remote Tape Drive

A remote system must be running system software release 4D1-3.2 or later to load the installation tools from a remote tape drive. Follow these steps:

1. If the *netaddr* variable on your system is not set, or is incorrect, you need to set it. (Refer to Section 2.1.1, ''Finding Network Addresses,'' for help doing this):

   ```
   setenv netaddr 192.35.73.96
   ```

2. Make sure the installation tape is locked securely in the tape drive on the remote system.

   If the remote system is running system software release 4D1-3.3 or later, make sure TFTP access to the tape drive is allowed. See step 8 in Section 2.2.1 for details.

3. Set the *tapedevice* environment variable as shown in the example below. Replace `solaris` with the name of the remote computer.

   ```
   setenv tapedevice bootp()solaris:/dev/nrtape/
   ```

   The trailing / is required.

4. Boot the standalone shell, *sash*, from the tape on the remote computer. Replace *cpu* with the appropriate CPU type from Table 2-3 in the previous section.

   ```
   boot -f ${tapedevice}(sash.cpu) --m
   ```

You see messages similar to these:

```
Obtaining /dev/tape from server remote.
Copying installation program to disk
```

followed by a series of dots.

If you have problems booting *sash*, see Section 2.3.3, ''Remote Installation Failure.''

After about ten minutes, you see this message:

```
Ready to install software.
Choose an item, then press <enter>:
     1. Automatically install software
     2. Use manual installation features
     3. Help
     4. Quit
```

Now you can install your software. See Section 2.2.2, ''Using *inst*,'' for further instructions.

From a Remote Distribution Directory

To load the *miniroot* from a directory located on a remote computer, follow these steps:

1. Use *distcp* to prepare a directory on the remote computer from which you plan to install your software. The system containing the distribution directory must be running 4D1-3.3 in order to read the date from the 4D1-3.3 distribution tapes. See Section 2.5.1, ''The *distcp* Command,'' for more details.

2. Set the *netaddr* variable on your system. (Refer to Section 2.1, ''Finding Network Addresses,'' for help doing this). To check the *netaddr* variable, type:

   ```
   printenv netaddr
   ```

   To set the *netaddr* variable, enter:

   ```
   setenv netaddr 192.35.73.96
   ```

3. Set the *notape* environment variable as shown below:

   ```
   setenv notape 1
   ```

4. Set the *tapedevice* environment variables as shown below. Replace *dir_path* with the complete path of the directory from which you want to install software:

   ```
   setenv tapedevice bootp()solaris:/dir_path/sa
   ```

5. Boot the standalone shell, *sash*, from the tape on the remote computer. Replace *cpu* with the appropriate CPU type from Table 2-3.

   ```
   boot -f ${tapedevice}(sash.cpu) --m
   ```

   You see messages similar to these:

   ```
   Obtaining /dir_path/sa from server remote.
   Copying installation program to disk
   ```

   followed by a series of dots.

   If you have problems at this point, see Section 2.3.3, ''Remote Installation Failure.''

   After about 10 minutes, you see this message:

   ```
   Ready to install software.
     Choose an item, then press <enter>:
        1. Automatically install software
        2. Use manual installation features
        3. Help
        4. Quit
   ```

   See Section 2.2.2, ''Using *inst*'' for further instructions.

2.3 Finishing Up the Installation

Adding and updating software often affects configuration files. The type and number of configuration files affected depend upon the product that you are installing.

When you finish installing software on your system, you might see a message reminding you to update your configuration files and remove obsolete ones.

The following list describes how *inst* handles configuration files that have been modified by the user prior to installation:

- If a configuration file on your system should not be updated and cannot be improved upon, *inst* changes nothing.
- If the product you are installing contains a new configuration file with optional improvements, *inst* leaves your files alone and installs the new file for comparison, with *.N* appended to the file name.
- If the product you are installing contains a new configuration file with mandatory improvements, *inst* installs the new file and renames your old one for comparison, with *.O* appended to the file name.
- Obsolete configuration files are renamed with a *.X* suffix. These files are *not* in the installation history of the product but can be located with the following command:

```
versions user | grep ".*\.X$"
```

See Section 2.5.2, ''The *versions* Command,'' for more information.

You can find detailed explanations of the purpose and format of configuration files in the following documents:

- *IRIX System Administrator's Guide*
- *System Tuning and Configuration Guide*
- *Network Communications Guide*

2.3.1 Error Recovery

If *inst* terminates abnormally, the system and the on-line installation history file is abandoned in an undefined state. This renders the product being installed or removed unusable. In such cases, you should correct the cause of the error if possible, and then either re-install or remove the software product that was being installed (or removed) when the problem occurred.

If the system is reset or has a power failure during installation, the system automatically reboots into the *inst* tool. This reduces the chances of having a partially updated system, which would probably result in strange behavior or system crashes later.

In the event that this happens, all that is necessary to restore the old boot partition information is to *quit* normally from *inst*. If, instead, you attempt to reload the *miniroot*, you will get a warning from *sash* that the root and swap partitions are the same. This is usually due to an interrupted installation.

If all else fails, the boot partition must be reset either by *fx*(1) (standalone or kernel) or by *dvhtool*(1M) under IRIX. The *System Tuning and Configuration Guide* contains information on booting from an alternate root partition.

2.3.2 Trouble Reading a Local Tape

If you have a problem in the early stages of the installation process, while the PROMS are attempting to load the *sash* program or copy the *miniroot* to the disk, it might be the result of a problem in reading the tape. Retry the operation at least one more time. Also, try cleaning the tape drive.

If the problem remains, and another system with a tape drive is available, try to read the tape on the other drive:

```
mt rewind
dd if=/dev/nrtape of=/dev/null bs=16k
mt rewind
```

This serves only to read through the data of the first file on the tape and copy it ''nowhere'', and may take several minutes. You should see no error messages, other than a count of records copied in and out.

If the problem occurs later in the process, after *inst* is running and an attempt is made to begin installing software, you might want to verify that the subsequent files on the tape can be read.

Note: Access is always achieved through the no-rewind variant of the tape device. If you are specifying a non-standard tape device, make sure you are using the no-rewind device.

Escape to a shell with the *sh* command, and issue these commands:

```
mt rewind
mt fsf 2
dd if=/dev/nrtape of=/dev/null bs=16k
dd if=/dev/nrtape of=/dev/null bs=16k
dd if=/dev/nrtape of=/dev/null bs=16k
dd if=/dev/nrtape of=/dev/null bs=16k
mt rewind
```

Each of the *dd* commands serves to read through the data of the next physical file on the tape. You should see no error messages other than a count of records in and out.

If the files cannot be read, there might be a problem with your copy of the installation tape and/or the tape drive. The most probable cause of this is a dirty tape drive.

If the files can be read, the tape and tape drive are probably OK. Exit the shell by typing *exit*. Issue a *from tape* command to cause *inst* to try and read from the tape again.

If the problem cannot be isolated using these techniques, call the Geometry Hotline at (800)345-0222.

2.3.3 Remote Installation Failure

If you have trouble reading from a remote tape drive early in the process, make sure that the tape is readable on the remote system's tape drive, as described in Section 2.3.2, "Trouble Reading a Local Tape." If the tape can be read properly, the problem is probably due to a poor network connection.

If you are attempting to install from a directory on a remote system, double-check the installation source system name and path names and then assume that the problem may be due to the network connection.

The PROMS require that the *netaddr* variable contain your system's IP address. If the remote system is being accessed through a network gateway system, the PROMS also require that the *gateaddr* variable contain the gateway system's IP address. The techniques in Section 2.1.1, ''Finding Network Addresses,'' will help you get these addresses. Use the PROM monitor prompt (>>) to examine this (with *printenv*) and/or change the current environment variables (with *setenv*). Double-check the addresses and correct them as necessary.

If you are installing through a network gateway system, make sure the gateway system is configured to forward *bootp* requests. Refer to Section 2.1.1.

If network access fails later in the process (for example, when *inst* is attempting to read the installation software), it might be due to a permissions problem. If you haven't explicitly specified an account name to be used on the remote system (through the *from* menu item), *inst* tries the *guest* account. If *guest* account access is not granted, this attempt fails.

When *inst* is running in the *miniroot*, it initializes TCP/IP just before the first attempt is made to read from the directory on the remote system. Once this occurs, you should be able to test various network connections using some of the standard TCP/IP commands. If you want to verify that a connection can be made to a particular system, through a particular user account, type `sh` at any *inst* prompt to escape to a shell and then issue this command:

```
rsh solaris -l user date
```

The remote system should respond with the current date. If this command fails, *inst* will also fail for the same reason. It might be that your system is not configured correctly. Examine the */etc/hosts* file.

If the error is permissions related, the user *inst* on your system is not allowed to use the *user* account on the remote system. There might be a more appropriate *user* account on the remote system. Sometimes, administrators set up a user account (and restricted shell) for use only by *inst*.

To direct *inst* to get software through a particular user account, use the *from* menu item:

from *user*@*server*:**/dev/nrtape**

or

from *user*@*server*:*dir_path*

If you wish, you can issue this command as soon as *inst* starts running. See Section 2.4.1, ''Manual Installation Menus,'' for more information on the *from* command.

2.3.4 Installing the Kernel Debugger

The following three error messages occur when a user has chosen to install *dev.sw.debug*, either in this, or an earlier installation. Part of the *dev.sw.debug* subsystem is the symbolic monitor portion of the kernel debugger *symmon.*

```
no room in volume header for symmon
[Aux cmd] if test @$instmode = @normal ; then /etc/dvhtool -v\
creat $rbase/stand/symmon symmon $vhdev ; fi

no room in volume header for sash
[Aux cmd] if test @$instmode = @normal ; then /etc/dvhtool -v\
creat $rbase/stand/sash sash $vhdev ; fi

no room in volume header for ide
[Aux cmd] if test @$instmode = @normal ; then /etc/dvhtool -v\
creat $rbase/stand/ide ide $vhdev ; fi
```

The first error occurs only while installing *dev.sw.debug*. If you do not plan to use the kernel debugger, you should simply not install this subsystem (or you may simply enter 'continue' on the interrupt menu, which completes the *dev.sw.debug* installation but does not install *symmon* in the volume header).

If you need to use the symbolic debugger, you need to repartition your drive with the 4D1-3.3 version of *fx* and completely reinstall your system. All data on the disk is lost in this case, so you must back up your system first. See the *IRIX System Administrator's Guide* for information on how to repartition your disk.

The second two forms of the error message may occur during the installation of *eoe1.sw.unix*, if you have installed *dev.sw.debug* during an earlier installation.

These files are installed in both */stand*, and in the volume header of the disk. If the system's disk layout was configured before the 4D1-3.2 release, then the volume header may not be large enough to contain *symmon* in addition to *ide* and *sash.*

symmon was first shipped in 4D1-3.2, and at that time the disk partitioning tool was changed to create a larger default volume header partition.

In some cases, earlier volume headers may have been just barely large enough, but increased sizes of *sash* and *symmon* may cause them not to fit.

If the error occurs on *sash* or *ide*, you should also choose the ''continue'' choice on the Interrupt/Error menu. You have *sash* from your previously installed release still installed. If you need the new *sash* and *symmon*, you need to repartition your disk (see above).

Otherwise, when the installation is complete, before quitting from the installation tool, use the *shroot* command, and then execute the command

```
dvhtool -v d symmon -v c /stand/sash sash
```

which removes *symmon* from the volume header, and replace the older *sash* with the new version (where 'sash' would be replaced in both places with 'ide' if the *ide* installation had failed).

2.3.5 Recovering from an Unbootable Kernel

For help recovering from an unbootable kernel, refer to Chapter 1 of the *System Tuning and Configuration Guide.*

2.4 The Installation Menus

inst provides a number of menus that let you control the installation process. Most of these menus apply only to using the manual installation features, but some, such as the Interrupt/Error menu, apply to both installation modes.

The menu items *help*, *set*, *quit*, *sh*, and *shroot* are available from every menu even though they are not listed on the menu. (Two exceptions to this are the *sh* and *shroot* items, which are never available from the Interrupt Menu.) These ''hidden'' menu items provide administrative support without cluttering up your screen. These items also are available from the Administration Functions menu, which is covered in detail in Section 2.4.3.

Some menu items can be used only from the *miniroot*. For example, you can invoke items that affect file systems and network initialization only from the *miniroot*.

You can choose an item on a menu by name or by number, confirming the selection by pressing `<enter>`. The names of the menu items are the first words on the line and can be abbreviated if you like.

Some of the menu items accept an argument (a name or number), which affects the way the menu item reacts. *inst* prompts you for an argument if it expects one and you have not supplied it.

2.4.1 Manual Installation Menus

The Manual Installation menu contains these items:

from [source]
Specifies the distribution source. The three kinds of source are the no-rewind tape device, a specific product descriptor file in a distribution directory, or a distribution directory. Any of these can be local or remote.

If the distribution source is tape, *inst* tries to position the tape and read the *product descriptor* file. The product descriptor is an internal representation of the product, image, and subsystem hierarchy, providing all necessary information on the product to *inst*.

To prepare a distribution directory, use the *distcp* command to copy software products from tape to disk. You can use the product descriptor files in the directory as distribution sources. You can name the distribution directory itself as a source. If you do this, *inst* reads all product descriptors in the directory. For more information on preparing a distribution directory, see Section 2.5, ''Special Features.''

If the distribution source is on a remote system that you can access over an Ethernet™ via TCP/IP, prefix the source name with *host:* or *user@host:*.

Here are some example distribution source names:

```
Manual> from /dev/nrtape
Manual> from dserver:/dev/nrtape
Manual> from guest@dserver:/u/dir_path
```

If you encounter problems accessing the distribution, see Section 2.3.3, ''Trouble Reading a Local Tape,'' or Section 2.3.3, ''Remote Installation Failure.''

list [names]

Lists information for the subsystems of the current distribution source. By default, all known subsystems are listed. You can use names or patterns to restrict the listing to specific subsystems.

You can use several special keywords to identify the subsystems you want to list, including all of the keywords in Table 2-5 (in ''The Subsystem Selection Menu'' section) and in Table 2-4.

Keyword	Function
products	lists products, not subsystems
images	lists images, not subsystems
sizes	list absolute subsystem sizes (rather than disk space deltas

Table 2-4. Subsystem Keywords

All remaining items are taken as subsystem names or patterns.

Each line in the output of the list menu selection describes one subsystem and contains several columns.

If the subsystem is currently selected for installation, the first column is ''i''. If a version of the subsystem is already installed and will be left alone during this installation, the first column is ''k''. If an ''r'' is present, then the subsystem will be removed from your machine.

If the subsystem is already installed on the disk, the second column is ''I''. If a subsystem from a previous release has been installed on the disk, the second column is ''X.'' An ''N'' in this column indicates that you are trying to install an older version of the subsystem than the one currently installed on your machine. If there is no related version on the disk, the second column is blank.

The third column is the subsystem name in the form *product.image.subsystem*. This name identifies the subsystem to the various menu items during subsystem selection.

The next columns are the disk space ''deltas'' for the subsystem. There is one column for the parent directory of each file system, usually / and */usr*. The heading shows the parent directory names. The delta is a number followed by a plus or minus sign to indicate the estimated change in disk space, in 512 byte blocks, that would result from installing that subsystem. (The numbers are estimates based on size of the existing files on the disk and the size of the new files to be installed.) These numbers can be negative, indicating that the new subsystem requires less room on that file system than the subsystem(s) it replaces.

The final column is the description of the subsystem, indicating the function of the files in the subsystem.

If the *verboselist* option is set, the subsystem list includes the product and image names, delta totals, and descriptions, along with the subsystems they contain.

standard — Installs the default set of subsystems. *inst* performs the standard pre-installation checks, then begins installation.

all — Installs all of the subsystems that are supplied with the distribution.

select

When both *standard* and *all* are inappropriate, use the *select* item to invoke the Subsystem Selection menu, from which you can choose the specific subsystems you want to install and perform other related functions.

recalculate

Recalculates the disk space deltas. If you remove or otherwise alter system files (that is, files that have been installed as part of a distribution) during a shell escape, any previous disk space computations will become invalid. If you have altered the sizes of system files during a shell escape, you should issue a *recalculate* command to recompute the disk space deltas. There is no harm, other than the delay, in recalculating more often than necessary. A reminder is given after each shell escape during which the free disk space changes significantly.

clean

Clears all files and directories from the / and */usr* file systems. All existing files and directories on the *root* and *user* file systems are lost. (See *remove* in the Subsystem Selection menu to remove specific subsystems.) *inst* requests confirmation when a file system already exists on the disk and does not proceed unless you explicitly answer `yes`.

Caution: Clean destroys all files on the / and */usr* file systems. Use it, with extreme care, only when you want to discard the entire contents of the disk. For peace of mind, back up the entire system before you invoke the clean item. See ''Backing Up and Restoring Your System'' in your owner's guide.

admin [cmd]

Invokes the Administration Functions menu. You can return to the Manual menu when you are finished.

If you give arguments, *inst* treats them as an administrative function to be executed immediately, without leaving this menu.

return

Returns to previous menu.

help [item]

The standard *help* item. The default help from this menu is a brief description of the menu and the items on it. For additional information, give the topic you want to find out about as an argument to *help*.

quit — Performs certain cleanup tasks and exits the software installation tool, returning control to IRIX or the *miniroot* special configuration program.

2.4.2 The Subsystem Selection Menu

The Subsystem Selection menu lets you control which subsystems to install on the disk, as well as other supporting tasks. You can examine the list of available subsystems, select or de-select them by name, examine a list of the file names in a subsystem, or remove previously installed subsystems from the disk. Once you remove the unwanted subsystems and make the desired selections, you can start the installation.

To determine whether to install a particular subsystem, consider the need for the functionality provided by the subsystem and the subsystem's size in relation to the available disk space. The subsystem listing should help you by showing the description of each subsystem and the change in disk space that results from installing it. You can remove at any time subsystems that have been installed but that you no longer need or want. Also, you can install at a later date any subsystems that you did not install initially. The Subsystem Selection menu contains these items:

list [names] — Lists current subsystem selections. This is the same as the *list* item of the Manual Installation menu; see the complete description above.

files [names] — Lists the names of files in subsystems. By default, lists the names of all files in all subsystems. If subsystem names or patterns are given, *inst* lists only the files in those subsystems.

install [names] — Selects the named subsystems for installation. The names can be

- product, image, or subsystem names
- patterns
- any of the special keywords listed in Table 2–5

Here are some examples:

```
install eoe2.sw.demos
```

Selects the demos subsystem of the *eoe2.sw* image.

```
install eoe1
```

All of the subsystems in the eoe1 product.

```
install eoe1.*.*
```

Equivalent; all subsystems in eoe1.

```
install eoe[12].sw
```

All subsystems in the *eoe1.sw* and *eoe2.sw* images.

```
install X
```

Selects those subsystems for which an older version is installed on the disk. See Table 2-5.

remove [names] Selects the named subsystems for removal from your machine.

keep [names] Cancels any *install* or *remove* selections for the identified subsystems. Use *keep* to identify subsystems that should be kept as is on the disk.

You can use several special keywords to specify the subsystems operated on by the *list*, *install*, *remove*, and *keep* menu items. Most of these keywords also have a one-letter abbreviation that may be used. Note that these one-letter abbreviations are case-sensitive. Table 2–5 lists the keywords and their abbreviations.

Letter	Keyword	Identifies subsystems that:
i	install	are marked for installation
k	keep	are marked for "keep"
r	remove	are marked for removal
I	installed	are already installed
U	uninstalled	are not installed
X	replaces	are replacements for an older installed version
N	replaced	are replaced by a newer installed version
d	default	are declared "default"
c	candidate	are subsystems for which this or an older version could (and may) have been installed at some point in the past (In other words, ‘‘not new’’)
(none)	inplace	are declared as being installable without the miniroot

Table 2-5. Keywords

Where applicable, you can combine these keywords (or letters). However, all tests must pass in order for the subsystem to be included. For example, to identify all subsystems that could have been installed at some point in the past but were not, you would use the keywords `candidate uninstalled` (or `c u` together).

default [names]

Sets default selections. The pre-installation checks determine which subsystems to install by default. If you install a product for the very first time, the default subsystems are predefined on the installation tape(s). If you update a product, the default subsystems are those that replace previously installed, corresponding subsystems, plus any pre-defined default subsystems on tapes that you have not installed before. New software directly replaces corresponding subsystems already on the disk.

Note: The *install*, *remove*, *keep*, and *default* menu items change the state of subsystem selection as reflected in the subsystem list; they do not start installation. You can review and alter the selections until you are satisfied with them, and then use the *go* item to initiate installation.

step [names]

Selects subsystems using interactive step mode. This item provides a convenient method of traversing the lists. The easiest way to select individual subsystems within a product or image is to use *step*. You can use the following keyword abbreviations in step mode:

i	install; select this subsystem for installation
r	remove; select this subsystem for removal
k	keep; cancel any install or remove request
d	default; use the default operation
p, -	go to the previous subsystem; `<ctrl-p>` has the same effect
n, +	go to the next subsystem; `<enter>` and `<ctrl-n>` have the same effect
/pat	search for a subsystem matching the pattern *pat* and continue from there. This is a convenient means of starting at a specific subsystem.
l	list subsystems, up to the current subsystem
f	list the file names in the current subsystem
h	display help for step mode
q	quit interactive step mode

step mode also supports the use of the arrow keys:

up-arrow	same as ''p'': go to previous subsystem
down-arrow	same as ''n'': go to next subsystem
left-arrow	same as ''k'': keep this subsystem as is
right-arrow	same as ''i'': select subsystem for installation

Using the **`<shift>`** key with any of these keys applies that command to the remainder of the product, rather than just the currently displayed subsystem.

(An exception to this is **`<shift>`** right-arrow, which is equivalent to **`D`** rather than **`I`**.)

Note: The easiest way to select subsystems over large, easily identifiable parts of the product is to use patterns and the *install* or *remove* items. For example, `remove *.man` will select all on-line manual pages for removal.

recalculate

Recalculates the disk space deltas. If you remove or otherwise alter system files (that is, files that have been installed as part of a distribution) during a shell escape, any previous disk space computations will become invalid. If you have altered the sizes of system files during a shell escape, you should issue a *recalculate* command to recompute the disk space deltas. There is no harm, other than the delay, in recalculating more often than necessary. A reminder is given after each shell escape during which the free disk space changes significantly.

go

Performs the standard pre-installation checks and then installs or removes subsystems based on the current selections.

admin [cmd]

Performs miscellaneous administrative functions. This item invokes the Administrative Functions menu, which is useful for dealing with file systems and network initialization. See ''The Administration Functions Menu'' below.

If you supply arguments, *inst* treats them as an administrative command to be executed immediately, without leaving this menu.

return

Returns to the previous menu from which the Subsystem Selection menu was invoked.

help [keyword]

Displays help information for general or specific subjects. Use *help topics* for a list of available topics and keywords.

quit

Terminates software installation.

2.4.3 The Administration Functions Menu

The Administration Functions Menu contains three kinds of items:

- Items that are hidden on other menus. Such items are included on the administration menu for reference.
- Items that initialize TCP/IP network access.
- Items that work with file systems.

The items related to the network and file systems are useful only in the *miniroot*. The administration menu displays these items:

set [options]

Sets, clears, or lists options. Options take values that are Boolean, integer, or string, depending on their function. Boolean valued functions are set to *on* or *off*.

If you give no arguments, *inst* lists the current settings.

If you give one argument, it should be the name of an option to be set to the default function. The default for Booleans is always *on*. The default for other types depends on their value. If you supply two arguments, *inst* uses the first as the option name and the second as the new value.

For example, the following command sets the number of lines displayed on the screen at a time (that is, between continuation prompts such as `more`):

```
Admin> set lines 24
```

Use *help set* to find out about the available list of options and how to use them.

sh [cmd]

Escapes to a shell or immediately runs the given command.

If the free disk space count changes significantly during a shell escape, you will be reminded to use the *recalculate* command if you altered or removed any system files.

shroot [cmd]

Escapes to a chrooted shell or immediately runs the given command, chrooted to the standard file system.

host [name] [addr]

Identifies local host name and/or address, setting the hostname of the system and making the appropriate entries in the *miniroot*'s */etc/hosts* file. If you do not provide a *name*, *inst* prompts for it. If the address of the given name is not in the *miniroot*'s */etc/hosts* file, or in the */etc/hosts* from the normal IRIX file system, *inst* prompts for it.

server [name] [addr]

Identifies remote host name and/or address, making appropriate entries in the *miniroot*'s */etc/hosts* file. If you do not provide *name*, *inst* prompts for it. If the address is not given, and is not in the normal file system's */etc/hosts* file, *inst* prompts for it.

mount [name] [dir]

Mounts an additional file system relative to the normal file system's *root*. By default, all file systems listed in the normal file system's */etc/fstab* are mounted during *miniroot* installations. This item makes it possible to mount other file systems. During *miniroot* installations, all normal file systems are mounted relative to */root* but are displayed without the */root* prefix.

umount [name]

Unmounts a file system. You can use this item to unmount file systems that were automatically mounted, or were mounted explicitly with the *mount* item.

mkfs [names]

Makes new file systems on the named devices. By default, the *root* and *user* file systems are remade; this is aliased as *clean* in the Manual Installation menu. You can name a specific device on which to make new file systems. You can then mount the new file system.

Warning: This deletes command all files on the named device.

versions [options]   Runs the *versions* command. Using *versions remove* at this menu allows you immediately to remove a subsystem. This item is also available from all other *inst* menus as a hidden menu option. See Section 2.5.2, ''The *versions* Command,'' for more information.

return   Returns to the previous menu.

help [item]   The standard help item.

quit   The standard quit item; ends the software installation process.

2.4.4 The Interrupt/Error Menu

The Interrupt/Error menu pops up if you interrupt an installation or if *inst* detects an error. To interrupt the installation, press `<ctrl-c>` unless the interrupt characters have been redefined by the *stty* commands in your configuration files. The menu allows you to:

- stop the installation,
- proceed with the installation, or
- abort the installation, without cleaning up.

If you type `<ctrl-c>` (interrupt) while software is not actually being installed, you terminate operation of the current item. Sometimes, the current operation must be complete before the interrupt is acknowledged, so it might take a moment for the operation to stop. If an error occurs, *inst* displays an appropriate error message before it displays the menu.

Note: While *inst* is executing certain ''auxiliary commands'' it will ignore your `<ctrl-c>`. Some of these procedures, which are clearly identified if you *set verbose on*, can take a few minutes to complete at which time *inst* will allow interruptions to occur again.

The Interrupt/Error menu contains these items:

stop
: Stops the installation prematurely, before the next file is installed. (There might be a few moments delay while work on the current file finishes.) Use *stop* if the error condition is such that you cannot or should not proceed with installation.

continue
: Proceeds with installation, ignoring the interrupt or error condition. Recommended if the error condition is well understood and known to be innocuous, or if you have simply interrupted the installation to *set verbose* on or off.

help [item]
: The standard help command. The default help topic is an explanation of the Interrupt/Error menu.

abort
: Immediately terminates *inst*, with absolutely no cleanup actions. This rather brutal way of stopping installation should be used only in extreme cases where *stop* does not seem to work.

Warning: While installing software, *inst* does not let you use the *sh* or *shroot* items to run the *versions* command or another *inst* because the on-line installation history might become damaged. You can, however, access all of the *versions* functions from the administration menu within *inst*.

2.5 Special Features

inst is the main tool you use to install software. Other tools, such as *versions*(1M) and *distcp*(1M), allow you to do certain types of system and installation administration. This section describes these and other installation tools.

2.5.1 The *distcp* Command

distcp(1M) is an IRIX command that lets you copy a software product from the installation tape(s) to a directory on a remote IRIS-4D Series workstation or server. The word *remote* describes a computer that is connected to your IRIS-4D Series workstation or server over a network.

A remote computer might contain hardware or software that you want to use during the installation. Once a product is in such a directory, you can install it on your system from the remote directory instead of using the cartridge tape on your IRIS-4D Series workstation or server. The 4D1-3.3 version of *distcp* can read all current and older tapes. Earlier versions of *distcp* cannot read 4D1-3.3 version tapes.

Installing software from a remote directory is helpful in situations where many computers must be updated, because network access is generally faster than tape access and more than one computer can access a remote directory at a time.

The following example shows how to copy a product from a cartridge tape in the no-rewind tape device (*/dev/nrtape*) to a directory called *dir*. You must always use the no-rewind tape device with the *distcp* command.

```
distcp /dev/nrtape /dir
```

You must use the 4D1-3.3 version of *distcp* on 4D1-3.3 tapes and images. (4D1-3.3 *distcp* can be used to read 4D1-3.2 tapes and images.)

Since *eoe1* contains the standalone shell *sash* for this release, it should be the last tape read with *distcp*.

Caution: Do not alter any of the files in a remote distribution directory because they might become unusable.

See the *IRIX System Administrator's Reference Manual* for detailed information about tape drive device names and using *distcp*(1M).

2.5.2 The *versions* Command

inst maintains an on-line installation history of the products, images, subsystems, and files that are installed or removed from the system during installation. *versions* is an IRIX command that gives you access to that history and lets you perform limited administrative duties. Using *versions*, you can:

- see which subsystems are installed
- list the file names in a product or subsystem
- locate configuration files
- see whether a configuration file was modified since it was installed
- remove subsystems

One common use of *versions* is to determine which configuration files were changed when new software was installed. When you are finished installing your software, enter the following command to identify the configuration files in question.

```
versions changed
```

Use *diff*(1) to compare the configuration files and a text editor to modify the active version. *versions* is available as a menu item from all installation menus, whether it is listed or not. See *versions*(1M) in the *IRIX System Administrator's Reference Manual* for detailed information.

2.5.3 The *lboot* Command

The *lboot*(1M) command allows an experienced system administrator to rebuild a kernel to reflect changes that were made to the operating system or system configuration files. *lboot* configures a bootable kernel using the files in */usr/sysgen/system*, and the files in the directories */usr/sysgen/boot* and */usr/sysgen/master.d*.

Use *lboot* not only after changing a parameter, but also after adding configurable software subsystems, adding software drivers for new hardware devices, or removing software drivers for hardware devices that no longer exist.

To reconfigure the system, first become the superuser, and then follow the procedure below.

1. It is best to save the original kernel (in case, for some reason, you need it later):

   ```
   cp /unix /unix.save
   ```

2. Change your working directory to */usr/sysgen* and run *lboot*, specifying the new kernel as "unix.install":

   ```
   cd /usr/sysgen
   lboot -u /unix.install
   ```

3. Send a message to all users logged into your system that you are bringing it down to install a new kernel. After making sure everybody has logged off, reboot the system:

   ```
   reboot
   ```

   When you run the *lboot* command, the system overwrites the current kernel, */unix*, with the kernel you have just created, */unix.install.* An autoconfiguration script, found in */etc/rc2.d/S95autoconfig*, runs during the startup process. (To override the autoconfiguration, you can rename this file. The *autoconfig* script operates when there have been changes to files in */usr/sysgen* or a new device driver has been installed.)

3. Compatibility

The 4D1-3.3.2 Maintenance Release maintains the same, or better, level of software compatibility as the 4D1-3.3 System Software Release.

4. Bug Fixes

This chapter describes bug fixes to Installation, IRIX, Networking, NFS and NIS, Graphics, 4Sight™, X11, Fortran, 4DDN, Power Fortran Accelerator, and Emacs. A Silicon Graphics software change request (SCR) number appears with some of the bug fixes in this chapter.

4.1 Bug Fixes to Installation

- SCR 9997 – The *versions remove* command issued without a specific product name removed all installed software. Fixed in 4D1-3.3.1.
- The timeout value for network installation was too short (5 seconds) for some sites. This has been increased to 30 seconds. Fixed in 4D1-3.3.1.
- The *distcp* command in 4D1-3.3.1 was not able to copy data from tapes to file systems. Fixed in 4D1-3.3.2.
- The installation of maintenance releases on diskless clients would fail to link and copy all of the needed files, resulting in a nonfunctional system. Fixed in 4D1-3.3.2.
- If the USESRVROOT environment variable was set (an unsupported feature in 4D1-3.3), the system administrator would be unable in many cases to correctly configure a diskless client due to NFS nested exporting rules. All references to this unsupported variable have been removed. Fixed in 4D1-3.3.2.

4.2 Bug Fixes to IRIX

- Using a backslash to escape metacharacters in csh would kill the csh process. Fixed in 4D1-3.3.1.
- The IKON printer did not work on a 4D/210 system. Fixed in 4D1-3.3.1.
- *lvinit*(1M) could erase a disk if invoked twice. Fixed in 4D1-3.3.1.
- SCR 10072 – *su*(1M) did not reset group IDs. Fixed in 4D1-3.3.1.
- SCR 9827 – *make*(1) handled command line variables incorrectly. Fixed in 4D1-3.3.1.
- If a program did an infinite recursion on the stack, it never got a chance to be interrupted. Fixed in 4D1-3.3.1.
- *memcmp*(3) did not return < 0 and > 0 to indicate lexicographical ordering. Fixed in 4D1-3.3.1.
- The *getpgrp*(3) manual page did not describe the BSD version of the routine. Fixed in 4D1-3.3.1.
- A process could be requested to block itself (via *blockproc*(2)) or stop itself (via *dbx*(1)). It would do so while in the kernel and potentially hold some important locks. Fixed in 4D1-3.3.1.
- SCR 10049 – Using *who* **–u** to find the idle time of a shell did not work for the console window. Fixed in 4D1-3.3.1.
- The – option to the *multgrps*(1) and *newgrp*(1) commands was fixed and documented in the manual pages. Fixed in 4D1-3.3.1.
- The IPI driver was corrected to have better recovery from a fatal error. Fixed in 4D1-3.3.1.
- The csh and sh *limit* command let you raise hard CPU time limits provided that you did not raise them by a factor greater than 100 in one try. Fixed in 4D1-3.3.1.
- *setgroups*(2) failed when the owner and group had already been changed. Fixed in 4D1-3.3.1.
- Certain customer applications did not run due to the way *exec*(2) interpreted section headers. Fixed in 4D1-3.3.1.

- *mt erase* caused a SCSI timeout on an 8mm tape drive. Fixed in 4D1-3.3.1.

- The 8mm tape driver did not properly read tapes produced from 4D1-3.2. Fixed in 4D1-3.3.1.

- When a disk drive could not determine its capacity, the system got a division by zero panic when it tried to use the capacity. This showed up on removable media drives when trying to format a cartridge. Fixed in 4D1-3.3.1.

- SCR 10622 – Using *init s* on a VGX system (going to single-user mode from multiuser mode) resulted in an apparent system hang. Fixed in 4D1-3.3.1.

- Attempting to start SoftPC with no floppy in the 3.5" drive resulted in the SCSI bus being reset. Fixed in 4D1-3.3.1.

- Three separate bugs were fixed in the hlock system calls including properly handling error returns from ioctl, allowing additional locks for sproc'd processes, and lock deallocation. Fixed in 4D1-3.3.1.

- SCR 10733 – The *getitimer*() routine returned a wrong value if the timer was previously set for greater than 2147 seconds. Fixed in 4D1-3.3.1.

- The devscsi library contained an error that prohibited timeouts greater than 65 seconds from being processed properly.

- SCR 9628 – The *mkdepend* **–c** option no longer adds **–M** to its argument; thus if invoked with a typical *cc*(1) command line, it will attempt a full compilation/link of files being analyzed. It is up to the invoking environment (usually a makefile) to include **–M** or the appropriate option in **–c's** argument. The man page has been updated to reflect this change from 4D1-3.2 operation. Fixed in 4D1-3.3.1.

- SCR 10048 – The assembler failed on **very** large function calls. Fixed in 4D1-3.3.1.

- *pixie*(1) did not work on programs calling certain routines on a machine with VGX graphics. Fixed in 4D1-3.3.1.

- The assembler incorrectly interpreted *.noalias,* so it is now ignored. Fixed in 4D1-3.3.1.

- The C compiler did not recognize that "foo(b = a);" was the same kind of continued struct assignment as "c = b = a;". Fixed in 4D1-3.3.1.

- SCR 10645 – Pixified programs dumped core. Fixed in 4D1-3.3.1.
- Users were occasionally unable to use *dbx*(1) with programs linked with libX11.a. Fixed in 4D1-3.3.1.
- A possible (but unlikely) system crash on a VGX could occur when doing certain shared process operations that involved copying data down to the graphics pipe. Fixed in 4D1-3.3.2.
- A user could incorrectly raise the maximum amount of lockable pages per user above what the user previously specifically configured. Fixed in 4D1-3.3.2.
- A user program could inadvertently inherit another program's address space due to translation problems associated with shared address processes when those processes shrunk their virtual space. Fixed in 4D1-3.3.2.
- A deadlock could occur in the system when forking a process that had a lot of locked memory. Fixed in 4D1-3.3.2.
- A resource counting bug could occur when locking pages and getting interrupted. Fixed in 4D1-3.3.2.
- Security holes in *mail*, *getty* and *uugetty* were corrected. Fixed in 4D1-3.3.2.
- The diagnostics used by Field Engineers could not be used on a 4D/70G system running 4D1-3.3. Fixed in 4D1-3.3.2.
- The kernel panicked while debugging a program with *edge*, stopping after an *mmap*() call and printing *paddr. Fixed in 4D1-3.3.2.
- There was a performance degradation for the tablet caused by the *tiedevice*() call. Fixed in 4D1-3.3.2.
- There was a performance degradation for the *setdblights*() call. Fixed in 4D1-3.3.2.
- At least one of the Seiko printer drivers did not work correctly. Fixed in 4D1-3.3.2.
- SCR 10162 – The variable blocking driver for the 1/2-inch Cipher tape drive did not recover correctly under certain error conditions such as timeouts and end of tape. Fixed in 4D1-3.3.2.

- The man page for the IPI driver incorrectly listed some error messages and needed clarification in the use of c/h/s values for bad block mapping. Fixed in 4D1-3.3.2.

- Under high-stress conditions, the streams functionality hung or panicked. Fixed in 4D1-3.3.2.

- The serial port driver was corrected to return an EBUSY status on an already busy port and when the CPU had not been initialized. Fixed in 4D1-3.3.2.

- The command buffer size was increased to avoid overflowing when using a *ttyf** device. Fixed in 4D1-3.3.1.

- The system administration utility *passmgmt* **–a** utility was not functional. Fixed in 4D1-3.3.2.

- SCR 10355 – The command *fuser* required MAXUMEM to be larger than physical memory. Fixed in 4D1-3.3.2.

- The command *fsck* failed on logical volumes larger than 5.8G bytes. Fixed in 4D1-3.3.2.

- Several miscellaneous system panics could occur in stress conditions. Fixed in 4D1-3.3.2.

- Additional error checking was needed to track page deallocation errors. Fixed in 4D1-3.3.2.

4.3 Bug Fixes to Networking

- Deleting Ethernet multicast addresses was broken for the integral Ethernet drivers on the Personal IRIS and POWER Series machines. Fixed in 4D1-3.3.1.

- If the IP options processing routine discovers an error in a packet using record route, it sends an ICMP error packet back to the originating host. The ICMP routine incorrectly computed a buffer length, eventually causing a ‘read address’ panic. Fixed in 4D1-3.3.1.

- The raw protocol family did not allow a socket to be bound to a down interface. Fixed in 4D1-3.3.1.

- SCR 9576 – A *select*(2) race condition sometimes caused wsh to hang. Fixed in 4D1-3.3.1.

- SCR 10633 – Telnet from a dumb terminal did not work if the terminal did not use parity. Fixed in 4D1-3.3.1.

- The *ftp*(1) 'put' command to transfer binary files did not transfer any data. Fixed in 4D1-3.3.1.

- For IRIX domain sockets, *shutdown*(2) did not work the same as Internet domain sockets. Now, when *shutdown* is called to disallow further writes on the socket, the reader socket is notified with an end-of-file condition. Fixed in 4D1-3.3.1.

- The mbuf limit handling routines could cause the system to hang under heavy network load, e.g., running NetVisualyzer on a busy net. Fixed in 4D1-3.3.2.

- The CPUs (3, 5, and 7) on a multiprocessor system could be saturated due to an error in the tty handling routines. Fixed in 4D1-3.3.2.

- IP fragmentation could occur in some circumstances causing decreased network performance. Fixed in 4D1-3.3.2.

4.4 Bug Fixes to NFS and NIS

- NFS did not support multiple group IDs. Fixed in 4D1-3.3.1.

  Note: The RPC/NFS protocol allows a maximum of 16 group IDs. If you configure your system to use more than 16 IDs, NFS will use only the first 16. (IRIX supports a maximum of 32 group IDs.)

- The NFS command *exportfs*(1M) did not recognize the ''hide'' option. ''hide'' is the default value and is not required. It is recognized for backwards compatibility with previous IRIX releases. Fixed in 4D1-3.3.1.

- A *select*(2) on a named pipe that was on an NFS-mounted file system returned immediately (ready for read), but a subsequent read returned 0. Fixed in 4D1-3.3.1.

- NFS could crash the system in rare cases where the client had caching disabled. Fixed in 4D1-3.3.1.
- The result of a read on a very large NFS-mounted file could be corrupted data. The source data was not affected. Fixed in 4D1-3.3.1.
- A process could be killed with a ''bad page read'' error due to signals not being ignored during critical paging operations. Fixed in 4D1-3.3.1.
- SCR 10682 – The on-line release notes have been updated to be current with the hard copy version. Fixed in 4D1-3.3.1.
- Use of NIS netgroups could cause standard utilities such as *su* and *id* to dump core due to a NULL pointer being returned by the *getpwent*() routine in *libsun.a*. Fixed in 4D1-3.3.2.
- The *getpwent*() routine in libsun.a failed on NIS password maps with more than 1024 entries. Fixed in 4D1-3.3.2.

4.5 Bug Fixes to Graphics

- The *setmonitor*(1) command and the *setmon*(3) Graphics Library call occasionally caused a graphics subsystem crash with a ''FIFO timeout'' message on VGX machines under release 4D1-3.3. Fixed in 4D1-3.3.1.
- SCR 10614 – When trying to draw a curved line fairly quickly the line would end up with little spikes all along it. The X and Y coordinates were not being synchronized properly by `getdev`() [VGX graphics option]. Fixed in 4D1-3.3.1.
- VME bus errors were being generated during initialization of the GF3 graphics board [G graphics option]. Fixed in 4D1-3.3.1.
- Bilinear interpolation `imp` routines needed to fetch the extra fog value from the input fifo when the z test failed [VGX graphics option]. Fixed in 4D1-3.3.1.
- `polysmooth`(PYSM_ON) and polymode(0) hung the graphics pipe [VGX graphics option]. Fixed in 4D1-3.3.1.
- `fbtool` did not properly find and display the differences between image files [VGX graphics option]. Fixed in 4D1-3.3.1.

- 5+ sided polygons were too slow with concave TRUE [VGX graphics option]. Fixed in 4D1-3.3.1.

- The man page for getgde(GDCLIP) stated that if user-definable clipping planes were supported, getgde should return a *1*. On a VGX the return value is 6 [VGX graphics option]. Fixed in 4D1-3.3.1.

- The man page for getgde (FORTRAN) had the parameter GDDDBO as one of the inquires for getgde. The correct parameter is GDDBBO. Fixed in 4D1-3.3.1.

- Fast rectcopy on the VGX was actually normal speed [VGX graphics option]. Fixed in 4D1-3.3.1.

- Fastpath lighting was incorrectly disabled in at least one case [VGX graphics option]. Fixed in 4D1-3.3.1.

- SCR 10449 – The VGX graphics code was slower than the GTX in one lighting model example [VGX graphics option]. Fixed in 4D1-3.3.1.

- If aapoly was on, turning *tex* on, then off requires new aapol [VGX graphics option]. Fixed in 4D1-3.3.1.

- VGX points clip incorrectly in fastest path [VGX graphics option]. Fixed in 4D1-3.3.1.

- The light demo showed problem with cxt switching [VGX graphics option]. Fixed in 4D1-3.3.1.

- SCR 10612 – getgdesc returned incorrect number of overlay planes on a system without alpha planes [VGX graphics option]. Fixed in 4D1-3.3.1.

- Several defects in the atom and sphere demos were corrected [VGX graphics option]. Fixed in 4D1-3.3.1.

- Flat shade + 5 vertex poly gave bad geometry [VGX graphics option]. Fixed in 4D1-3.3.1.

- foreground() coredumped when linked to gl_s from a server. Fixed in 4D1-3.3.1.

- SCR 10648 – When specifying the S direction for contouring, the plane equation was not interpreted correctly [VGX graphics option]. Fixed in 4D1-3.3.1.

- The alphaplanes were not set to 0xff for the VGX `c3s()` command [VGX graphics option]. Fixed in 4D1-3.3.1.
- SCR 9859 – 200+ vertex polf fills incorrectly with some pixels missing [VGX graphics option]. Fixed in 4D1-3.3.1.
- A Live Video Digitizer stress condition would crash GT and GTX graphics [GT and GTX graphics options]. Fixed in 4D1-3.3.1.
- SCR 9578 – Rectzoom's of (1.2,1.2) kill the Window Manager on a 4D/20 [Personal IRIS graphics]. Fixed in 4D1-3.3.1.
- SCR 10441 – A customer application could crash a 4D/25 Turbo when shading a large model. The specular table look up was overflowing due to bad normal [Personal IRIS graphics]. Fixed in 4D1-3.3.1.
- The color map for some Personal IRIS systems was not initialized, causing all RGB applications to break [Personal IRIS graphics]. Fixed in 4D1-3.3.1.
- If a child process after an sproc following a winopen does a winopen and tries to draw, the Personal IRIS would panic [Personal IRIS graphics]. Fixed in 4D1-3.3.1.
- SCR 10613 – Total window confusion could occur on large applications due to a window reference not being maintained properly [Personal IRIS graphics]. Fixed in 4D1-3.3.1.
- Several problems with the `setvideo()/getvideo()` feature in the VGX system microcode were fixed. Fixed in 4D1-3.3.2.
- The `compactify()` utility caused application level software to core dump. Fixed in 4D1-3.3.2.
- SCR 10799 – The VGX swapinterval did not correctly set the delay time. Fixed in 4D1-3.3.2.
- SCR 10085, 9976, 9977 – Many of the demos were modifed to work correctly in a 14" monitor environment. Fixed in 4D1-3.3.2.
- A bug in the VGX overlay plane routines could cause a core dump. Fixed in 4D1-3.3.2.
- The Personal IRIS could panic on exit of some graphics programs. Fixed in 4D1-3.3.2.

4.6 Bug Fixes to 4Sight

- SCR 10003 – In 4Sight, crossing a window boundary with the right mouse button down generated INPUTCHANGE events. Fixed in 4D1-3.3.1.

- Heavy use of subwindows killed 4Sight with PostScript®—a stack overflow error. Fixed in 4D1-3.3.1.

- When entering a subwindow, the inputchange event value for leaving the parent window was returned to the user the first time, but not on subsequent in/out of the subwindow. Fixed in 4D1-3.3.1.

- *fullscrn* disrupted the refresh of subwindows on the Personal IRIS. Fixed in 4D1-3.3.1.

- If a psroff document directly included some PostScript, *psview* did not display any of the document text that came after the included PostScript. Fixed in 4D1-3.3.1.

- The font manager was forced to use printermatched fonts when casting a large font even when the application turned printermatching off. Fixed in 4D1-3.3.1.

- SCR 10671 – The man pages for `swinopen`(3G) and `winposition`(3G) have been updated to document that `viewport`() or `reshapeviewport`() must be called after `winposition`() for a subwindow, and that the arguments to `winposition`() for a subwindow are relative to the origin of the main window, not screen absolute. Fixed in 4D1-3.3.1.

- SCR 10691 – WorkSpace would fail on NFS-mounted home directories when it tried to issue a lock. Fixed in 4D1-3.3.1.

- The 4Sight copyarea primitive could cause portions of a screen to display incorrectly depending on the overlapping of windows. Fixed in 4D1-3.3.2.

- Icons for the windows of NeWS clients could disappear under other windows when stowed. Fixed in 4D1-3.3.2.

- The 4Sight version number was changed from 1.5 to 1.51 to reflect the changes in functionality from the previous bug fixes. Fixed in 4D1-3.3.2.

- The /First utility for 4Sight's /TidyState functionality was not correctly implemented, causing unpredictable results. Fixed in 4D1-3.3.2.
- SCR 10876 – 4Sight subwindows would not keep focus when soft attached. Fixed in 4D1-3.3.2.
- The font manager code was found to be too inaccurate to fully support WYSIWYG applications. The font manager and many of the fonts available to the user were updated. Fixed in 4D1-3.3.2.
- A *gethostbyname*(3B) call during 4Sight's initialization was removed to reduce network traffic. Fixed in 4D1-3.3.2.
- Zero width characters caused the NeWS server to abort. Fixed in 4D1-3.3.2.

4.7 Bug Fixes to X11

- SCR 9577 – Large menus did not get properly drawn the second and subsequent times after they were popped up. Fixed in 4D1-3.3.1.
- The X server could dump core with xmultibiff. Fixed in 4D1-3.3.1.
- SCR 9517 – The X11 server returned the wrong dimensions on 14" monitors. Fixed in 4D1-3.3.2.
- SCR 9951 – Xfig labels did not display. Fixed in 4D1-3.3.2.
- The X libraries were updated with patch 17 from MIT for miscellaneous bug fixing, including bug fixes required for support of Motif 1.1 from OSF. Fixed in 4D1-3.3.2.
- XDefineCursor and XUndefineCursor would not display certain large cursors. Large cursors are now clipped to the top left 16x16 pixels and used as defined in the X spec. Fixed in 4D1-3.3.2.

4.8 Bug Fixes to Fortran

- SCR 9457 – A Fortran regression occurred in DOWHILE/ENDDO parsing. Fixed in 4D1-3.3.1.
- SCR 9459 – Logical*1 did not work in Fortran. Fixed in 4D1-3.3.1.
- SCR 9546 – Rewinding a file after writing to it did not cause the desired truncation of the file. Fixed in 4D1-3.3.1.
- SCR 10159 – Fortran intrinsic function *IIOR*() performed 'AND' for OR and XOR. Fixed in 4D1-3.3.1.
- Fortran **–r8** option produced wrong results. Fixed in 4D1-3.3.1.
- When a Fortran variable with length shorter than 4 bytes was initialized with a typeless constant (that is, a hex. constant) wrong results would occur. Fixed in 4D1-3.3.1.
- SCR 9541 – A Fortran problem occurred when dealing with C-style #includes. Fixed in 4D1-3.3.1.
- SCR 10621 – A bug was fixed in the runtime I/O library where changing from read to write mode did not involve an *fseek*() operation to reset the FILE structure. Fixed in 4D1-3.3.1.
- SCR 10611 – Only the first element of an array was initialized. Fixed in 4D1-3.3.1.
- Performance corrections were made for unformatted writes in Fortran. Fixed in 4D1-3.3.1.
- Fortran namelist I/O destroyed the user's ERR = recovery. Fixed in 4D1-3.3.1.
- SCR 10735 – Fortran varargs with multiple mixed entries would core dump. Fixed in 4D1-3.3.1.
- Backspace execution failed to reset the I/O mode to reading, causing the file to be truncated if many backspaces were executed consecutively after a write. Fixed in 4D1-3.3.2.
- SCR 11028 -The *len*() function returned a wrong answer if used in an argument list inside a multiple entry subroutine. Fixed in 4D1-3.3.2.

4.9 Bug Fixes to 4DDN

- SCR 10688 – After a 210 system was upgraded to 4D1-3.3 with VGX, the 4DDN startup script */usr/etc/dn/dninstall.sh* was run. The execution of *dninstall.sh* hung after a series of NCP was displayed on screen. Fixed in 4D1-3.3.1.

4.10 Bug Fixes to Power Fortran Accelerator

- The Power Fortran Release Notes and man page were updated to document known errors. Fixed in 4D1-3.3.1.
- SCR 10586 – The Power Fortran Accelerator translator could die while in the ''-unroll=4 -roundoff=2'' mode. Fixed in 4D1-3.3.1.
- The *mp*(3f) man page did not document the new **mp_setlock** and **mp_unsetlock** routines. Fixed in 4D1-3.3.1.
- The blocktime was incorrectly set to 0 when using the MP_BLOCKTIME environment variable. Fixed in 4D1-3.3.2.

4.11 Bug Fixes to Emacs

- The emacs binary was not tagged properly to enable execution from the WorkSpace. Fixed in 4D1-3.3.1.

5. Changes

This chapter describes changes to the software since the last release.

5.1 Changes to IRIX

- *mkdepend*(1) has been changed to work better with different language translators. Formerly, it appended a **–M** option to the compiler specified by the **–c** option, showing a pronounced bias toward *cc*(1) usage. Now *mkdepend*'s invoker must include any option(s) needed to make the compiler emit raw *make*(1) dependencies as part of the **–c** option's quoted argument.

5.2 Changes to Networking

- The network startup script, */etc/init.d/network*, has been changed to support the optional FDDIXPress controller and software. If the FDDI controller is installed, it will be the primary interface, while the ethernet controller will be the first gateway interface.

  The other important change to the script eliminates the need to edit the script for systems with more than 2 network interfaces. The user-editable file */etc/config/netif.options* lets you override defaults for interface names and addresses of all the network interfaces. See the comments in the file for details.

  Also, the routing daemon, *routed*(1M), can be disabled with the new *routed* configuration flag. Use *chkconfig*(1M) to change the state of the flag.

Palatino-Roman
Symbol
Times-Bold
Times-BoldItalic
Times-Italic
Times-Roman
ZapfChancery-MediumItalic
ZapfDingbats

The names shown above are used in PostScript programs to select a particular style or typeface, and create a font in that style at some specified point size.

It is very important that the symbols displayed on the screens of SGI workstations match as closely as possible the symbols printed on SGI printers. Matching screen fonts for all 35 printer typefaces listed above are now available in the following six standard point sizes:

8, 10, 12, 14, 18 and 24

Additional point sizes are available for some typefaces.

Each of the 35 typefaces listed above uses one of the following three character sets:

- The SGI character set for text typefaces.
- The SGI character set for the Symbol typeface.
- The SGI character set for the Zapf Dingbats typeface.

The SGI character set for text typefaces contains the following characters in specified positions:

Position	**PostScript Character Name**
0-31	UNUSED
32	space
33	exclam
34	quotedbl
35	numbersign
36	dollar
37	percent
38	ampersand
39	quoteright

40	parenleft
41	parenright
42	asterisk
43	plus
44	comma
45	hyphen
46	period
47	slash
48	zero
49	one
50	two
51	three
52	four
53	five
54	six
55	seven
56	eight
57	nine
58	colon
59	semicolon
60	less
61	equal
62	greater
63	question
64	at
65	A
66	B
67	C
68	D
69	E
70	F
71	G
72	H
73	I
74	J
75	K
76	L
77	M
78	N
79	O
80	P

81	Q
82	R
83	S
84	T
85	U
86	V
87	W
88	X
89	Y
90	Z
91	bracketleft
92	backslash
93	bracketright
94	asciicircum
95	underscore
96	quoteleft
97	a
98	b
99	c
100	d
101	e
102	f
103	g
104	h
105	i
106	j
107	k
108	l
109	m
110	n
111	o
112	p
113	q
114	r
115	s
116	t
117	u
118	v
119	w
120	x
121	y

122	z
123	braceleft
124	bar
125	braceright
126	asciitilde
127	Aacute
128	Acircumflex
129	Adieresis
130	eacute
131	Aring
132	Atilde
133	Ccedilla
134	edieresis
135	Ecircumflex
136	Edieresis
137	Egrave
138	Iacute
139	Icircumflex
140	Idieresis
141	copyright
142	registered
143	trademark
144	egrave
145	ntilde
146	scaron
147	uacute
148	Scaron
149	ucircumflex
150	ugrave
151	zcaron
152	logicalnot
153	Ydieresis
154	Zcaron
155	aacute
156	brokenbar
157	onesuperior
158	twosuperior
159	threesuperior
160	atilde
161	exclamdown
162	cent

163	sterling
164	fraction
165	yen
166	florin
167	section
168	currency
169	quotesingle
170	quotedblleft
171	guillemotleft
172	guilsinglleft
173	guilsinglright
174	fi
175	fl
176	degree
177	endash
178	dagger
179	daggerdbl
180	periodcentered
181	mu
182	paragraph
183	bullet
184	quotesinglbase
185	quotedblbase
186	quotedblright
187	guillemotright
188	ellipsis
189	perthousand
190	ecircumflex
191	questiondown
192	Agrave
193	grave
194	acute
195	circumflex
196	tilde
197	macron
198	breve
199	dotaccent
200	dieresis
201	Eacute
202	ring
203	cedilla

204	Igrave
205	hungarumlaut
206	ogonek
207	caron
208	emdash
209	Ntilde
210	Ograve
211	Oacute
212	Ocircumflex
213	Otilde
214	Odieresis
215	multiply
216	otilde
217	Ugrave
218	Uacute
219	Ucircumflex
220	Udieresis
221	Yacute
222	Thorn
223	minus
224	agrave
225	AE
226	acircumflex
227	ordfeminine
228	adieresis
229	aring
230	Eth
231	ccedilla
232	Lslash
233	Oslash
234	OE
235	ordmasculine
236	igrave
237	iacute
238	icircumflex
239	idieresis
240	eth
241	ae
242	ograve
243	oacute
244	ocircumflex

245	dotlessi
246	odieresis
247	divide
248	lslash
249	oslash
250	oe
251	germandbls
252	udieresis
253	yacute
254	thorn
255	ydieresis
256	onequarter
257	onehalf
258	threequarters
259	plusminus
260	infinity
261	lozenge
262	Delta
263	Omega
264	product
265	summation
266	partialdiff
267	integral
268	radical
269	radicalex
270	pi
271	approxequal
272	greaterequal
273	lessequal

Some typefaces may not contain all these characters, but each available character is placed into its specified position. The above character positions are the default character positions. You can use the PostScript encoding vector to rearrange available characters.

The SGI character set for the Symbol typeface contains the following characters in specified positions:

Position	PostScript Character Name
0-31	UNUSED
32	space
33	exclam
34	universal
35	numbersign
36	existential
37	percent
38	ampersand
39	suchthat
40	parenleft
41	parenright
42	asteriskmath
43	plus
44	comma
45	minus
46	period
47	slash
48	zero
49	one
50	two
51	three
52	four
53	five
54	six
55	seven
56	eight
57	nine
58	colon
59	semicolon
60	less
61	equal
62	greater
63	question
64	congruent
65	Alpha
66	Beta
67	Chi

68	Delta
69	Epsilon
70	Phi
71	Gamma
72	Eta
73	Iota
74	theta1
75	Kappa
76	Lambda
77	Mu
78	Nu
79	Omicron
80	Pi
81	Theta
82	Rho
83	Sigma
84	Tau
85	Upsilon
86	sigma1
87	Omega
88	Xi
89	Psi
90	Zeta
91	bracketleft
92	therefore
93	bracketright
94	perpendicular
95	underscore
96	radicalex
97	alpha
98	beta
99	chi
100	delta
101	epsilon
102	phi
103	gamma
104	eta
105	iota
106	phi1
107	kappa
108	lambda

109	mu
110	nu
111	omicron
112	pi
113	theta
114	rho
115	sigma
116	tau
117	upsilon
118	omega1
119	omega
120	xi
121	psi
122	zeta
123	braceleft
124	bar
125	braceright
126	similar
127-160	UNUSED
161	Upsilon1
162	minute
163	lessequal
164	fraction
165	infinity
166	florin
167	club
168	diamond
169	heart
170	spade
171	arrowboth
172	arrowleft
173	arrowup
174	arrowright
175	arrowdown
176	degree
177	plusminus
178	second
179	greaterequal
180	multiply
181	proportional
182	partialdiff

183	bullet
184	divide
185	notequal
186	equivalence
187	approxequal
188	ellipsis
189	arrowvertex
190	arrowhorizex
191	carriagereturn
192	aleph
193	Ifraktur
194	Rfraktur
195	weierstrass
196	circlemultiply
197	circleplus
198	emptyset
199	intersection
200	union
201	propersuperset
202	reflexsuperset
203	notsubset
204	propersubset
205	reflexsubset
206	element
207	notelement
208	angle
209	gradient
210	registerserif
211	copyrightserif
212	trademarkserif
213	product
214	radical
215	dotmath
216	logicalnot
217	logicaland
218	log
220	arrowdblleft
221	arrowdblup
222	arrowdblright
223	arrowdbldown
224	lozenge

225	angleleft
226	registersans
227	copyrightsans
228	trademarksans
229	summation
230	parenlefttp
231	parenleftex
232	parenleftbt
233	bracketlefttp
234	bracketleftex
235	bracketleftbt
236	bracelefttp
237	braceleftmid
238	braceleftbt
239	braceex
240	apple
241	angleright
242	integral
243	integraltp
244	integralex
245	integralbt
246	parenrighttp
247	parenrightex
248	parenrightbt
249	parenrightbt
250	bracketrighttp
251	bracketrightex
252	bracketrightbt
253	bracerighttp
254	bracerightmid
255	bracerightbt

The SGI character set for the Zapf Dingbats typeface contains the following characters in specified positions:

Position	PostScript Character Name
0-31	UNUSED
32	space
33	a1
34	a2
35	a202
36	a3
37	a4
38	a5
39	a119
40	a118
41	a117
42	a11
43	a12
44	a13
45	a14
46	a15
47	a16
48	a105
49	a17
50	a18
51	a19
52	a20
53	a21
54	a22
55	a23
56	a24
57	a25
58	a26
59	a27
60	a28
61	a6
62	a7
63	a8
64	a9
65	a10
66	a29
67	a30

68	a31
69	a32
70	a33
71	a34
72	a35
73	a36
74	a37
75	a38
76	a39
77	a40
78	a41
79	a42
80	a43
81	a44
82	a45
83	a46
84	a47
85	a48
86	a49
87	a50
88	a51
89	a52
90	a53
91	a54
92	a55
93	a56
94	a57
95	a58
96	a59
97	a60
98	a61
99	a62
100	a63
101	a64
102	a65
103	a66
104	a67
105	a68
106	a69
107	a70
108	a71

109	a72
110	a73
111	a74
112	a203
113	a75
114	a204
115	a76
116	a77
117	a78
118	a79
119	a81
120	a82
121	a83
122	a84
123	a97
124	a98
125	a99
126	a100
127	a86
128	a85
129	a95
130	a205
131	a89
132	a87
133	a91
134	a90
135	a206
136	a94
137	a93
138	a92
139	a96
140	a88
141-160	UNUSED
161	a101
162	a102
163	a103
164	a104
165	a106
166	a107
167	a108
168	a112

169	a111
170	a110
171	a109
172	a120
173	a121
174	a122
175	a123
176	a124
177	a125
178	a126
179	a127
180	a128
181	a129
182	a130
183	a131
184	a132
185	a133
186	a134
187	a135
188	a136
189	a137
190	a138
191	a139
192	a140
193	a141
194	a142
195	a143
196	a144
197	a145
198	a146
199	a147
200	a148
201	a149
202	a150
203	a151
204	a152
205	a153
206	a154
207	a155
208	a156
209	a157

210	a158
211	a159
212	a160
213	a161
214	a163
215	a164
216	a196
217	a165
218	a192
219	a166
220	a167
221	a168
222	a169
223	a170
224	a171
225	a172
226	a173
227	a162
228	a174
229	a175
230	a176
231	a177
232	a178
233	a179
234	a193
235	a180
236	a199
237	a181
238	a200
239	a182
240	UNUSED
241	a201
242	a183
243	a184
244	a197
245	a185
246	a194
247	a198
248	a186
249	a195
250	a187

251	a188
252	a189
253	a190
254	a191
255	UNUSED

Since the positions of some unencoded PostScript characters, such as the copyright character in text fonts, have been changed, unencoded characters may not be displayed correctly by all programs.

The name of a .fm file used to contain a number that corresponded to the point size of the screen font stored in that file. The name of each .fm file now contains a number that corresponds to the pixel size of the screen font stored in that file, where the pixel size is defined as:
pixel-size = point-size * font-resolution-in-dots-per-inch / 72.

6. Known Problems and Workarounds

This chapter describes known problems and suggestions for avoiding them.

6.1 Known Bugs in IRIX

If *syslogd*(1M) is enabled to output messages to */dev/console* and the console is a *wsh*(1) window, the system might become unusable due to an interaction between *syslogd* and the graphics console daemon, *grcond*. The problem doesn't exist on machines that use a terminal as the console, such as POWER Series servers.

Here's a workaround using the filter mechanism in *syslogd*. *grcond* messages are in the ''daemon'' facility. Any selector lines in */etc/syslog.conf* that output daemon messages to the console must be filtered to remove *grcond* messages. A typical example of such an entry is:

```
*.debug;kern.none    /dev/console
```

The steps are:

1. Create the shell script called */usr/adm/grcond.filt* containing:

   ```
   #!/bin/sh
   # this syslogd filter rejects grcond messages
   read line
   set $line
   case "$1" {
       grcond\[*)
           exit 0
           ;;
   }
   echo "$line\c"
   exit 0
   ```

2. Make the shell script executable:

   ```
   chmod +x /usr/adm/grcond.filt
   ```

3. Change *syslog.conf* entries that print daemon messages to */dev/console* to use the filter. Using the example above:

   ```
   *.debug;kern.none    |/usr/adm/grcond.filt   /dev/console
   ```

   Note: Use tab characters to separate the fields in the *syslog.conf* entries.

4. Tell *syslogd* to re-read */etc/syslog.conf*:

   ```
   killall 1 syslogd
   ```

6.2 Known Bugs in NFS

The *automount*(1M) direct maps do not work. The workaround is to use explicit entries in */etc/fstab* to mount the desired directories.

6.3 Known Bugs in X

There are server problems in the XCopyArea functionality when copying from window to window with raster ops.

7. Documentation Errors

This chapter describes errors in documentation.

The *Network Communications Guide* contains the following error:

- On page 7-13, the maximum number of gids in the *auth_unix* structure is 16, not 10.