Sun386*i* Administrator's & Developer's Notes

Sun386*i* SunOS Release 4.0.1

The *Sun386i Administrator's & Developer's Notes for SunOS 4.0.1* are the "release notes" for topics covered in the Owner's Supplement Documentation Set (*Sun386i Advanced Administration* in particular) and Developer's Toolkit Documentation Set. This document replaces the *Sun386i Administrator's & Developer's Notes* for SunOS™ 4.0. The notes include information about new features not covered in these documents, differences in the way the software operates from the way it's documented in the manuals, and additional tips and helpful suggestions. Read this document if you are setting up a Sun386i™ network (with a Sun386i, Sun-3™, or Sun-4™ system as a server) or multiuser standalone system, if you are an advanced user, or if you are developing software for the Sun386i system. Be sure to always use the latest version of these notes.

Problems?
If you encounter problems not covered in the Sun386i documents you have received, please call us at: **(800) USA-4SUN**. Outside the U.S., call your local Sun Microsystems service office.

Contents

Sun386i Advanced Administration

Sun-3 and Sun-4 Server Kit for Sun386i Diskless Systems

Sun386i Developer's Notes

More About Help Fonts

Index

Sun386i Advanced Administration

There are two books that cover administration for the Sun386i workstation—*Sun386i SNAP Administration* and *Sun386i Advanced Administration*. This section contains corrections and additions to *Sun386i Advanced Administration*. For corrections and additions to *Sun386i SNAP Administration*, see the *Sun386i Owner's Bulletin for SunOS 4.0.1*.

The *Sun386i Advanced Administration* manual may be useful to you as a system or network administrator. This book provides information on topics such as manually installing systems and creating user accounts, installing third-party software, repartitioning disks, Yellow Pages, and advanced network administrative tasks. Every site should have at least one copy of *Sun386i Advanced Administration*. This book is part of two documentation sets—the Sun386i Owner's Supplement Documentation Set (part no. SR-9B) and the Sun386i Documentation Conversion Set (part no. SR-9D). The Conversion Set includes books specific to the Sun386i workstation, that is, books not included in the standard SunOS 4.0 documentation set.

System Installation

This section contains software and documentation notes pertaining to Chapter 1 of *Sun386i Advanced Administration*.

Connecting systems to a network – You can install systems more quickly and use fewer resources if you connect them one at a time to the network. Although you can install multiple diskful systems simultaneously, the process will take longer per system and will make it more difficult for the server to do anything else. However, you *cannot* install multiple diskless systems simultaneously. Connect diskless systems to the network one at a time.

Installing diskless Sun386i workstations on other networks – You can now install a diskless Sun386i system on a non-Sun386i network. Sun386i SunOS 4.0.1 includes a server kit that allows a Sun-3 or a Sun-4, running SunOS 4.0 and Yellow Pages, to support a diskless Sun386i system. See the section on the server kit, within these notes, for details on how to install the server kit and subsequently install diskless Sun386i systems.

Installing diskful Sun386i systems on Sun386i networks – The directions for manually installing a diskful Sun386i system as a network client are incomplete. Step 6 on page 5 of *Sun386i Advanced Administration* should state:

6. Connect the system to the Ethernet. The system then displays a list of choices. Select choice 1–"`Create or join a network.`" Then, the system displays several messages and finishes booting. It is now installed on the network.

Diskless client support – SNAP does not display both of the parameters—disk space limitation and client number limitation—that control whether a boot server can accept a diskless client. It only shows the client number limitation. After the diskless client is installed, there must still be at least 40 Mbytes of free space on the disk that the diskless client is using. The diskless client's root and swap areas may be on different disks, affecting this calculation. Swap space defaults to 16 Mbytes.

The defaults are set so that a 91 Mbyte server will not accept diskless clients and a 327 Mbyte server, with no extra software installed, will accept more clients than are advisable for performance reasons.

The /etc/bootservers file sets the parameters for how the boot server will serve its clients. See the Yellow Pages chapter of the *Sun386i Advanced Administration* manual for an explanation of this file. The sixth field will usually have the value "40000" (40 Mbytes). If this is set to zero, then the only disk space requirement is that there is 8 Mbytes of free space for a diskless client's root area. (Note that the *Sun386i Advanced Administration* manual contains an error in the description of the /etc/bootservers file—the descriptions of the last three fields are in the wrong order. See the note under the "Yellow Pages" section later in these notes, for a correct description of this file.)

System Installation on Mixed Networks

The following are general things to consider when installing Sun386i systems on a mixed network.

Set up a Sun386i as a YP master – If possible, you should set up a Sun386i system as the YP master for other Sun386i clients on your network. This is because:

♦ You'll be better able to organize and control installation.

♦ Automounting features are more consistent with the Sun386i systems being served.

♦ The YP domain can take advantage of Sun386i YP maps not available on Sun-3 and Sun-4 YP masters (for example, ypprintcap).

♦ Sun386i systems in this Yellow Pages domain can take advantage of SNAP, Automatic System Installation, and New User Login features.

If you already have a YP domain, you should either convert your network to use the Sun386i YP domain, or you should set up the Sun386i systems on the existing network. Do not set up two YP domains.

System administration with non-Sun386i YP masters – If you are using a Sun-3 or Sun-4 workstation as the YP master for Sun386i systems, you must administer Sun386i systems manually.

You cannot use SNAP, Automatic System Installation, and New User Login features; these are only available if a Sun386i system is the YP master for the domain.

Networks without Yellow Pages – If your network is not using Yellow Pages, you must set up a Sun386i master server. See page 166 of *Sun386i SNAP Administration.*

Turning off automatic installation features – Even on all-Sun386i domains, it's sometimes desirable to disable Automatic System Installation or New User Login. See Chapter 1 of *Sun386i Advanced Administration* for instructions on how to turn off these features.

Even with these features disabled, you can continue to use a Sun386i master server.

Diskless clients – If you are using a Sun-3 or Sun-4 as the bootserver for a Sun386i diskless system, keep the following points in mind:

♦ The bootserver must be running SunOS release 4.0 or greater.

◆ The `setup_client`, `setup_exec`, and `suninstall` procedures do not currently accommodate Sun386i diskless clients—you must use the Sun386i server kit instead.

For more information on support of diskless clients, see the section "Sun-3 and Sun-4 Server Kit for Sun386i Diskless Systems" later in these notes.

Subnets – Subnets are set up automatically using the YP `netmasks.byaddr` map. If this map does not exist, and you need to set up a system on a subnetted network, see *Sun386i Advanced Administration* for details of what to do.

Installing applications – Keep in mind that the `/usr` partition on Sun386i systems is mounted read-only. When you are installing or reinstalling an application, follow the vendor's instructions about where to store the files on Sun386i systems. If you have developed your own applications, refer to the *Sun386i Developer's Guide* for further information.

Network Administration

This section contains software and documentation notes pertaining to Chapter 2 of *Sun386i Advanced Administration.*

Changing a domain name – The instructions for changing the domain name on a Sun386i network (or standalone system) given on pages 18–19 of *Sun386i Advanced Administration* are incomplete. After step 3, perform the following additional step:

3a.   Delete the file `/var/yp/netid.time`

Without this additional step, all secure RPC-based applications (such as SNAP, Automatic System Installation, and New User Account generation) will fail RPC authentication.

Multiple domains on a network – You cannot have two Sun386i YP domains on the same network, even after setting the policies as described on page 19 of *Sun386i Advanced Administration.* This is because the DRARP daemon cannot be completely disabled by setting the `ip_address_allocation` policy to none .

Disabling root logins – To disable root logins on the console, you must remove the `secure` field *and* the –n option from the `console` entry in the `/etc/ttytab` file. If you remove the `secure` field from the `console` in `/etc/ttytab` and do not disable logintool by removing the –n option, users can still log in as root if they know the password.

Secure NFS – To use secure NFS™, you must reconfigure your SunOS kernel to include DES encryption code. You do this by enabling the `CRYPT` option. (This is omitted from page 27 of *Sun386i Advanced Administration.*) See *System and Network Administration*, included in the Owner's Supplement Documentation Set, for details on how to reconfigure the kernel.

Publickey credential problem – On a Sun386i network, user accounts (and the superuser for a system) may use some special authentication information stored in the `publickey.byname` Yellow Pages map. This is used for the Secure RPC authentication system, which enhances the security of the network.

Sometimes a user will experience problems with some applications using his or her account. In particular, if the network administrator deletes a user's entry from the /etc/publickey file and rebuilds YP, various systems will incorrectly think that the user is using the old entry in this map.

Secure RPC applications, such as secure NFS or SNAP, sometimes generate errors with RPC credentials or verifiers. The problem will often go away if you delete the file /etc/keystore from the system where the user logged in, and then reboot.

Systems may experience a similar problem. After deleting the publickey entry for a system from the /etc/publickey file, you must delete the /etc/.rootkey file on that system, along with /etc/keystore, before you reboot the system.

Locking services on non-Sun386i home directory servers – If Sun386i users have their home directories stored on a Sun-3 or Sun-4 workstation, you should update the locking services (the "lock daemon") for these machines. Without this update, Sun386i users whose home directories are served by a Sun-3 or Sun-4 workstation may see DOS and other applications "hang" when they try to use them. For non-Sun home directory servers, see the note at the end of these instructions.

To perform the update:

1. Instruct users to quit all applications.
2. Locate the Sun386i SunOS 4.0.1 diskette labeled "Sun3/4 Lockd Fix Diskette." This diskette contains the locking service changes to upgrade the servers. Insert this diskette into the Sun386i drive (you'll perform this update over the network).
3. Create a temporary directory on the Sun386i system to hold the new software:

   system:1} **mkdir /tmp/lockd**

   system:2} **cd /tmp/lockd**
4. Copy the lock daemon software from the diskette using the following command:

   system:3} **bar xvZpf /dev/rfd0c .**
5. Remotely log into the server as root (for *server*, substitute the system name of the system you are updating):

   system:4} **rlogin *server* -1 root**
6. Kill the existing rpc.lockd process. Use the following command to see the process ID:

   SUPERUSER} **ps -ax | grep lockd**

   Then use kill -9 followed by the process ID to kill the lock daemon process. (See *Sun386i Advanced Skills* for information on the kill command.)
7. Change to the directory containing the lock daemon software.
   On a server running SunOS 3.x software, enter **cd /etc**
   On a server running SunOS 4.0 software, enter **cd /usr/etc**
8. Rename the original lock daemon software:

   SUPERUSER} **mv rpc.lockd rpc.lockd.old**
9. Copy the new file from the Sun386i system to the server. (For *system*, substitute the name of the Sun386i system on which you loaded the lock daemon software. For *arch*, substitute Sun3 or Sun4 as appropriate.)

   SUPERUSER} **rcp *system*:/tmp/lockd/*arch*/rpc.lockd .**
10. Restart the locking service:

   SUPERUSER} **rpc.lockd**

12. Log off the server by typing **logout**.

13. Remove the temporary lock daemon files you loaded earlier onto the Sun386i system:

```
system:5}  cd /tmp
system:6}  rm -rf lockd
```

Users of DOS Windows™ and other Sun386i applications will need to wait at least 45 seconds before restarting their applications.

Note: No lock daemon update is available for non-Sun systems. Therefore, there is no network-wide locking for users with home directories on non-Sun systems. However, locking between windows on a single Sun386i system works fine, even with no locking services. But you should be sure that a single user doesn't run DOS when logged onto two or more machines at the same time, and that more than one user does not try to access DOS on the same Sun386i system at the same time. Also, be sure that no one uses DOS applications that require file locking.

If people are using a non-Sun system as a home directory server, add the following line to users' .login files to prevent an error message from being displayed on the Sun386i system when a DOS window starts up :

setenv DOS_LOCKING OFF

SunLink DNI driver change – If you're using SunLink™ DNI, you must modify the DNI driver for it to work correctly under SunOS 4.0.1. See the note on this topic in the "System Software" section of the Sun386i Developer's Notes, later in this document.

Shutting down diskless clients – As root, you cannot use the **shutdown** command to shut down a diskless client. You must be a member of the **operator** group to do this. However, you can use the **halt** command (as root) or the **Shut down** option on the SunView™ menu to shut down a diskless client.

User Accounts

This section contains software and documentation notes pertaining to Chapter 3 of *Sun386i Advanced Administration*.

Creating user accounts on standalones and Sun386i networks – The instructions for creating user accounts, documented on pages 32–34 of *Sun386i Advanced Administration*, are incorrect. Steps 9 and 10 should be reversed; also, steps 11 and 12 are incorrect. Perform steps 1–8, and then the following four steps (9–12):

9. Create the directory /export/home/*groupname* if it does not already exist. Then create a symbolic link in /export by entering the following command:

    **ln   -s   /files/home/*groupname*/*username* \
    /export/home/*groupname*/*username***

10. Place a copy of the files in ~*groupname*/defaults in the user's home directory. You can use the following command to do this provided that none of these files have been protected against remote root access. If this is the case, you will have to go to the system containing these files and copy them from there; then return to the user's home directory server.

    ~*groupname*/copy_home ~*groupname* ~*username*

11. Change the ownership of the user's home directory and its files to the new user, using the chown command:

chown −R *username.groupname ~username/* .

12. Export the new user's home directory by entering the following commands:

echo *'/export/home/groupname/username* **−access=domain'** **>> /etc/exports**

exportfs −a

Stop being superuser by entering the **exit** command.

Now complete the instructions on page 34, steps 13 and 14.

Printers, Terminals, and Modems

This section contains software and documentation notes pertaining to Chapter 4 of *Sun386i Advanced Administration*.

Adding an AT bus serial card – The procedure for adding an AT serial card on pages 41 and 42 of *Sun386i Advanced Administration* is incorrect. Follow the procedure documented here.

You can add up to two additional serial ports to your system by installing AT-compatible serial boards, and then activating SunOS software required to operate the boards. Serial boards installed this way are accessible to both DOS and SunOS systems. Standard AT COM1/COM2 boards provide one or two serial ports. Multiport (4, 8, or 16) serial boards are also available from some PC equipment vendors, but you must use a third-party driver or your own custom-written driver to access them.

Follow these steps to complete the installation of a serial card and make it accessible to DOS:

1. Set the switches or jumpers on the serial card or modem card for the appropriate interrupt levels and I/O addresses. You may choose COM1 (interrupt level 4, address 3£8, port name ttym0), COM2 (interrupt level 3, address 2£8, port name ttym1), or both. Then install the board or card in the system unit. Consult Appendix B of *Sun386i System Setup and Maintenance* as well as the note on this in the *Sun386i Owner's Bulletin for SunOS 4.0.1*.

2. After starting up your system, use the su command in a Commands window to become superuser.

3. Enter the following command to edit the system's rc.local file:

   SUPERUSER} **textedit /etc/rc.local**

4. Remove the # symbol from the following line:

   #modload ats.o −exec ats.script −conf ats_conf && chat 'AT serial port driver.'

5. Save the file and quit the Text Editor.

6. Still as superuser, enter the following command:

   SUPERUSER} **cd /etc/dos/defaults**

   If the serial port is to be used as COM2, enter:

   SUPERUSER} **mv com2 com2.orig**

   SUPERUSER} **ln −s /dev/ttym1 com2**

   Note that ttym1 will be created by the modload command in step 9 on the next page (if it does not already exist). The same is true for ttym0, when the serial port is COM1.

If the serial port is to be used as COM1, enter:

```
SUPERUSER} mv com1 com1.orig
SUPERUSER} ln -s /dev/ttym0 com1
```

7. Check your ~/pc/setup.pc file to ensure that the com*n* line you are using is *not* commented out.

8. Save your work and quit any open DOS window on your Desktop.

9. Still as superuser, enter the following commands:

```
SUPERUSER} cd  /etc/modules
SUPERUSER} modload ats.o -exec ats.script -conf ats_conf
```

10. Exit from superuser. The new serial port will be available to DOS and SunOS systems.

11. If you see the message "vd load: none of the specified devices are on line," this means your serial board is not connected or seated properly. In this case, shut down the system and physically reinstall the serial board. Then repeat steps 1–10 above.

The installation is now complete. You can use SNAP at this point to add a printer, terminal, or modem to the newly installed port.

Adding a printer, terminal, or modem – To add one of these peripherals so that it can be later maintained using SNAP, you should perform an additional step. (This step was omitted from the documentation in Chapter 4.) The device should be added to the /etc/ext_ports file. The format of this file allots one line per peripheral, with eight fields. The first two fields are separated by a colon and the rest of the fields are separated by a single tab. The fields and values are:

Field	Value
System name	oak, for example
Port name	ttya is the system serial port; pp0 is the system parallel port
Peripheral type	printer, terminal, or modem
Status	on or off for printers and terminals
	in, out, or in_out for modems
Baud rate	the baud rate of the printer, terminal, or modem; for a printer connected to a parallel port, insert the baud rate of 9600
Model	hayes, for example
Printer name	lp, for example; there should be one printer named lp on the system or network (this is the default printer); leave this field blank when adding a terminal or a modem
Location	this field is optional; if included, a number sign (#) should precede the location, making this field a comment

The following example adds an Epson™ printer to the parallel port on the system named oak. The printer is enabled, named lp, and located in room 39.

```
oak:pp0  printer  on  9600  epson  lp  #room 39
```

Add the printer to the /etc/ext_ports file *before* you issue the make command (step 3 on page 42 of *Sun386i Advanced Administration*).

For modems and terminals, you need to also issue the `make` command. After entering the modem or terminal in the `/etc/ext_ports` file, enter the following command as superuser:

SUPERUSER} **cd /var/yp;make**

Adding a printer to a non-Sun386i system – When you add a printer to a non-Sun386i system, you should perform the following steps to make the printer available from the Sun386i systems.

If the Yellow Pages master is a Sun386i system:

1. Log in to the YP master and become superuser.
2. Add an entry to the file `/etc/ypprintcap`, in the following format:

   name: lp=: rm=*systemname*: rp=*name*: sd=/var/spool/*name*

   where *name* is the name of the printer, for example, `lp3`, and *systemname* is the name of the system that the printer is connected to.
3. Enter the following command:

   SUPERUSER} **cd /var/yp; make**

If the YP master is *not* a Sun386i system:

On each system from which you want to access the printer, perform the following:

1. Log in and become superuser.
2. Add an entry to the file `/etc/printcap`, in the following format:

   name: lp=: rm=*systemname*: rp=*name*: sd=/var/spool/*name*

   where *name* is the name of the printer, for example, `lp3`, and *systemname* is the name of the system that the printer is connected to.

Adding a Hayes 2400 modem – The instructions for adding a modem on pages 49–52 of *Sun386i Advanced Administration* are incomplete for the Hayes® 2400 modem. See the "Hayes 2400 modem" note in the *Sun386i SNAP Administration* section of the *Sun386i Owner's Bulletin for SunOS 4.0.1* for details on installing this modem.

File System Structure

This section contains software and documentation notes pertaining to Chapter 5 of *Sun386i Advanced Administration*.

The root (/) file system – The root file system contains the directory `stand`. This information was omitted from pages 59 and 60 of *Sun386i Advanced Administration*. The `/stand` directory is the directory for hardware diagnostics. It contains a standalone copy program, tape boot program, and a copy of the boot program.

The /files file system – The `/files` file system no longer contains the directory `vol.local`. The contents of this directory, as described on page 66 of *Sun386i Advanced Administration*, are now contained in the directory `/files/vol/local`.

The /export file system – The `/export/vol/local` subdirectory is no longer a symbolic link to `/files/vol.local`. It is a symbolic link to `/files/vol/local`.

UUCP files – Some files contained in `/usr/lib/uucp` are now contained in `/etc/uucp`.

Installing Third-Party Software

This section contains software and documentation notes pertaining to Chapter 6 of *Sun386i Advanced Administration*.

Loading third-party applications – Step 2 of the "Pre-Installation Steps" section (page 74 of *Sun386i Advanced Administration*) has an error. The line

`/vol/local/bin.`*arch*

should read:

`cd /vol/local/bin.`*arch*

Exporting an application or directory – The command to ensure that the directory `/export/vol` exists (step 2, page 76 of *Sun386i Advanced Administration*) is no longer needed. You do not need to perform this step with SunOS 4.0.1.

Registering an application – If you performed the additional steps for registering an application with an icon (page 78 of *Sun386i Advanced Administration*) using the 4.0 `.orgrc` files, these files must be replaced with the new ones. The 4.0 `.orgrc` files are incompatible with the 4.0.1 Sun Organizer™. See *Installing Sun386i SunOS 4.0.1* for more details.

Automounter – The automounter has an option that allows it to select one of many servers to satisfy a particular automount request. For example, if `/vol/local` is replicated on more than one server, the following `auto.vol` map entry allows the automounter to mount from any one of the hosts listed in the map entry:

`local   host1:/usr/local   host2:/usr/local   host3:/usr/local`

This has the advantage that the automounter can satisfy the mount request if one or more of the servers is unavailable. However, this option does not work on the 4.0.1 automounter. A patch will be made available through customer support for customers requiring this feature.

Repartitioning and Restoring Disks

This section contains software and documentation notes pertaining to Chapter 7 of *Sun386i Advanced Administration*.

Increasing swap space – In the instructions for increasing swap space, steps 4 and 5 on page 88 of *Sun386i Advanced Administration* are reversed; perform step 5 first, then step 4.

Yellow Pages

This section contains software and documentation notes pertaining to Chapter 8 of *Sun386i Advanced Administration*.

The /etc/bootservers file – There is an error in the description of the file /etc/bootservers on page 104 of *Sun386i Advanced Administration*. The last three fields are in the following order, with corrected values:

Field	Value
Minimum free Kbytes in tmp	8 Mbytes
Minimum free Kbytes in root	40 Mybtes
Minimum free Kbytes in swap	0

Mail

This section contains software and documentation notes pertaining to Chapter 9 of *Sun386i Advanced Administration*.

Administering mail – Mail is delivered to users' home directories. If users have any home directories on non-Sun386i systems, they will have to log in on their own systems to read their mail.

Creating mailing lists – SNAP does not maintain mailing lists for groups in the format shown on page 115 of *Sun386i Advanced Administration*. Mailing lists for groups are contained in the file /etc/ypaliases, in the following format:

groupname: username, username, username

Undelivered mail – The Sun386i system is set up to have users' mail delivered to their home directories. If users are not receiving their mail, it might be because the mail is not going to their home directories. Be sure that:

♦ /usr/ucb comes before /bin or /usr/bin in the users' search paths (in the .login, .cshrc, or .profile files)

♦ The MAIL environment variable is set in the users' .login files

After correcting either of these two problems, you need to log out, then log in again, to effect these changes.

Back Up and Restore

When you specify the Z option with bar, bar uses the /tmp directory to compress and uncompress files. If there is not enough room in the /tmp directory for bar to copy a file and compress or uncompress it, the file will be added to the bar archive uncompressed, or extracted from the bar archive and left in its compressed format.

Sun-3 and Sun-4 Server Kit for Sun386i Diskless Systems

About the Server Kit

The Sun386i SunOS 4.0.1 tape media includes all the files required to allow a Sun-3 or Sun-4 system to act as a server for one or more diskless Sun386i machines. The server kit components are:

◆ These notes

◆ Sun386i Application SunOS distribution tape

◆ Sun386i Developer's Toolkit distribution tape (optional)

The server kit is intended for use by persons who have some understanding of Sun networks. Do not attempt to use it if you do not understand the concepts of IP addresses, Ethernet addresses, or Yellow Pages.

The server kit makes the following assumptions:

1. That the Sun-3 or Sun-4 server is running SunOS version 4.0 or greater.

2. That the Sun-3 or Sun-4 server is in a Yellow Pages domain.

   If you have a diskful Sun386i system already installed on the network, then a Yellow Pages domain is present. If no Yellow Pages domain is present, one must be set up (see the *System and Network Administration* manual).

3. That the server system and the diskless Sun386i clients reside in the same Yellow Pages domain.

   If the server system and the Sun386i clients are in different domains you must perform several manual steps in addition to running the server kit scripts.

4. That the server system is the Yellow Pages master server for the domain.

   If the server system is not the Yellow Pages master server for the domain in which the server and clients reside, you must perform several manual steps in addition to running the server kit scripts.

5. That you do not need to use SNAP and New User Accounts facilities on the diskless Sun386i systems.

   If you have a diskful Sun386i system installed on the network, it is preferable to use this system as the Yellow Pages master server for the domain. This will allow limited use of SNAP and New User Accounts on the diskless Sun386i systems.

Note: It is highly recommended that the server be a Yellow Pages master or slave server. Administration of multiple Yellow Pages domains on a network is complicated and error-prone.

There are three steps to installing the server kit, covered in the following sections:

1. Loading the Server Kit Scripts

2. Running the `sun386server` Script

3. Setting Up the Diskless Sun386i Clients

The final three sections discuss:

◆ Manual procedures for loading optional clusters (if you do not load them when you run the server kit scripts)

◆ How to remove a diskless Sun386i client from a non-Sun386i server, and

◆ What files are modified by the server kit scripts

Loading the Server Kit Scripts

The server kit scripts are in `tar` format on file 4 of the Application SunOS tape. The tapes are available in two sizes:

♦ 1/2-inch reel tape (6250 bpi)

♦ 1/4-inch cartridge tape (QIC-24)

The server kit scripts should be loaded in the Sun-3 or Sun-4 directory `/usr/etc/install`. To load the scripts:

1. Become superuser on the server by typing **su** and then entering the root password.

2. Change directories to `/usr/etc/install` by typing the following:

   cd /usr/etc/install

3. Load the Application SunOS tape into the tape drive.

4. Load the scripts from tape by typing the appropriate commands from the table below. The server kit is available only on 1/4-inch tape as of 4.0.1.

	Local	**Remote**

From 1/2″ Tape Drive	`mt -f /dev/nrmt8 rew`
	`mt -f /dev/nrmt8 fsf 3`
	`tar xvpbf 64 /dev/nrmt8`
		`rsh host mt -f /dev/nrmt8 rew`
		`rsh host mt -f /dev/nrmt8 fsf 3`
		`rsh host dd if=/dev/nrmt8 bs=64b`
		`
			tar xvpBbf 64 -`

From 1/4″ Tape Drive	`mt -f /dev/nrst8 rew`
	`mt -f /dev/nrst8 fsf 3`
	`tar xvpbf 64 /dev/nrst8`
		`rsh host mt -f /dev/nrst8 rew`
		`rsh host mt -f /dev/nrst8 fsf 3`
		`rsh host dd if=/dev/nrst8 bs=64b`
		`
			tar xvpBbf 64 -`

In the above table, *host* is the name of the remote system. Terminate each of the three command lines listed in the table with a carriage return. A list of server kit files will appear on your screen.

Running the sun386server Script

Once you have loaded the server scripts from the Application SunOS tape, you must run the server kit script `sun386server`. This script, run on your Sun-3 or Sun-4 system, allows it to serve Sun386i systems. The script requires you to enter the following information:

♦ Tape size (1/4-inch) and location of drive (local or remote)

♦ Where to load Sun386i software components (described below)

♦ What Sun386i Application SunOS and Developer's Toolkit optional clusters to load

The `sun386server` script loads software from the Sun386i distribution tapes into six different directories on the server system. You are allowed to specify where these six paths will be. Where you choose to locate these six directories depends on your particular file system layout and disk space availability. The first three directories are similar to those required by any diskless Sun system:

♦ The executable path—where the Sun386i `/usr` file system will be loaded. The `/usr` file system requires about 20 Mbytes of disk space. The default executable path is `/export/exec`.

♦ The root path—where the roots for Sun386i clients will be created. The default root path is `/export/root`. Each diskless root requires about 2.2 Mbytes of disk space.

♦ The swap path—where the swap files for Sun386i clients will be created. The default swap path is `/export/swap`. Each Sun386i diskless client requires a 16 Mbyte swap file.

The final three paths are specific to the Sun386i diskless systems:

♦ The cluster path—where the Sun386i optional clusters are loaded. The entire set of Application SunOS clusters requires about 13.5 Mbytes of disk space and the entire set of Developer's Toolkit clusters requires about 17.5 Mbytes of disk space. The default cluster path is `/export/cluster`.

♦ The local path—the `/usr/local` directory for the Sun386i clients. The default local path is `/export/local`. No software is initially loaded into this directory.

♦ The help path—where the help files will reside. The default help path is `/export/help`. This directory is automounted as `/vol/help` on the Sun386i clients. By default it contains symbolic links to the default help files in `/usr/lib/help`.

Note: The Application SunOS tape must be loaded in the tape drive before you run the `sun386server` script.

The following instructions show how to run the `sun386server` script to set up a Sun-3 system named `fred` as a server for diskless Sun386i clients. Files are being loaded remotely from a Sun-4 system named `barney`. The default paths (defaults appear in square brackets) are accepted by pressing the ⌐Return⌐ key. What you should enter is shown in **boldface** type.

1. Change directories by typing the following:

    cd /usr/etc/install/sun386

2. Execute the script:

    sun386server

3. The system responds with:

    ```
    What size are your distribution tapes?
          1 - 1/4"
          2 - 1/2"
    Enter [1-2]: 1
    ```

    As of 4.0.1, the server kit is available only on quarter-inch tape.

4. Tell whether you are loading from a local or a remote tape drive.

    ```
    Are you loading from a remote tape drive? (y/n): y
    ```

    If you are loading from a remote tape drive, you will be prompted to provide the name of the remote host to which the tape drive is attached.

    ```
    Enter the remote host name: barney
    ```

5. Type in the executable path (where the Sun386i /usr file system will be loaded).

    ```
    Enter the executable path [/export/exec]: Return
    ```

    Executables will be loaded in a subdirectory of the path you specify in this step. The subdirectory will be labeled sun386.sunos4.0.1. In this example, the full executable path is /export/exec/sun386.sunos4.0.1.

6. Type in the root path (where the root directories for Sun386i clients will be created).

    ```
    Enter the root path [/export/root]: Return
    ```

7. Type in the swap path (where the swap files for Sun386i clients will be created).

    ```
    Enter the swap path [/export/swap]: Return
    ```

8. Type in the cluster path (where the Sun386i optional clusters are loaded).

    ```
    Enter the cluster path [/export/cluster]: Return
    ```

    Clusters will be loaded in a subdirectory of the path you specify in this step. The subdirectory will be labeled sun386.sunos4.0.1. In this example, the full cluster path is /export/cluster/sun386.sunos4.0.1.

9. Type in the local path (the /usr/local directory for the Sun386i clients).

    ```
    Enter the local path [/export/local]: Return
    ```

    The actual path used will be a subdirectory of the local path you specify. The subdirectory will be labeled sun386.sunos4.0.1. In this example, the full local path is /export/local/sun386.sunos4.0.1.

10. Type in the help path (where the help files will reside).

    ```
    Enter the help path [/export/help]: Return
    ```

11. The system then asks whether you want to load all the Application SunOS Optional Clusters.

Note: The load and loadc commands that allow you to load Application SunOS and Developer's Toolkit clusters are not available on Sun systems other than the Sun386i system. Once you run the sun386server script, clusters can only be loaded manually using the procedure outlined in the "Loading Clusters Manually" section. Therefore, you should load all the clusters you think that you will need at the time you run the sun386server script.

    ```
    Do you want to load all the Application SunOS optional
    clusters? (y/n): y
    ```

    If you respond by typing **y**, all the clusters will eventually be loaded (see step 13 of this procedure). If you respond by typing **n,** you will be prompted for each cluster individually. It's a good idea to load all clusters.

12. The system then asks whether you want to load all the Developer's Toolkit optional clusters.

Note: You should respond to the following questions by typing **n** unless you have a Developer's Toolkit tape and plan on loading clusters from it at this time.

    ```
    Do you want to load all the Developer's Toolkit optional
    clusters? (y/n): n
    ```

If you respond by typing **y**, all the clusters will be loaded. If you respond by typing **n**, you will be prompted for each cluster individually (see below).

```
Do you want to load the base_devel cluster? (y/n): y

Do you want to load the sysV_devel cluster? (y/n): n

   .

   .

Do you want to load the config cluster? (y/n): n
```

13. The clusters you specified in the above steps are now ready to be loaded. Insert Application SunOS tape1 into the tape drive and confirm by typing **y** in response to the following prompt:

```
Insert Application SunOS tape 1, confirm (y/n): y
```

You will see the following messages as files are extracted:

```
Positioning tape ...

Extracting contents of stand and sbin ...

   .

   .

Extracting contents of /usr ...

   .

   .

Extracting help files ...

Extracting the extended_commands cluster ...

   .

   .
```

If you chose to load any of the Developer's Toolkit optional clusters in step 12, you will be prompted to place the Developer's Toolkit tape into the tape drive, and the clusters will be loaded. Otherwise, the server configuration is complete.

Setting Up the Diskless Sun386i Clients

Once your machine is set up as a Sun386i server, you can install Sun386i clients at any time by running the `sun386client` script (as root on the server). For each Sun386i system you want to install, you must run the `sun386client` script once, providing the following information:

♦ Client's Yellow Pages domain (default is domain of server)

♦ Client's host name

♦ Client's IP address

♦ Client's Ethernet address

♦ All the systems on which home directories of users of the client machine will reside (see "A Note About Home Directories" on the next page)

The `sun386client` script performs all functions necessary to install the client, except in cases where:

♦ The server is not the Yellow Pages (YP) master server for the client's YP domain

♦ The server is not the YP master server for its own YP domain

In these cases the script will instruct you to perform one or two simple manual steps (including running another script on the Yellow Pages master server) to complete the installation of the diskless client. If the YP master server is a Sun386i system, after performing all necessary manual steps, log on as superuser to the YP master server and type:

cd /var/yp; make

A Note About Home Directories

In order to retain compatibility with Sun-3 and Sun-4 conventions, access of home directories from diskless Sun386i clients on Sun-3 or Sun-4 servers requires explicit specification of the server machine in the path.

For example, users of the client machine may have their home directories on the server machine, but users may also have home directories on any diskful machine in the network.

The case illustrated earlier in the server kit section assumes a network configuration like that shown below, in which some of the home directories for users of Sun386i client pebbles are stored on the server fred, but others are stored on a machine called barney.
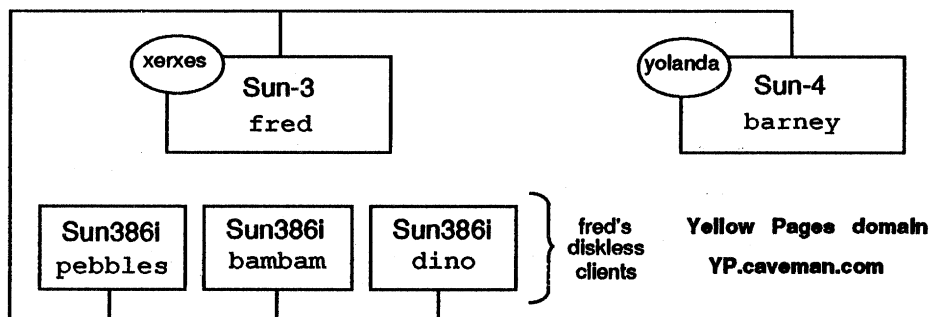
After you install diskless Sun386i client `pebbles` as shown in "Running the `sun386client` Script" (below), the home directories for users of `pebbles` can be accessed by the paths `/home/fred` and `/home/barney`. This is because, while running the `sun386client` script, both `fred` and `barney` are specified as home directory servers for `pebbles`. (See step 7 on the next page.)

In this case, the home directory for user `xerxes` resides on system `fred` and is accessed by the path `/home/fred/xerxes`. The home directory for user `yolanda`, on the other hand, resides on system `barney` and is accessed by the path `/home/barney/yolanda`.

Running the sun386client Script

The following instructions show how to run the `sun386client` script to install the diskless Sun386i client `pebbles` in Yellow Pages domain `YP.caveman.com`. The system `pebbles` has IP address `121.2.121.212` and Ethernet address `9:8:76:5:4:3c`.

1. Running as root on system `fred`, change directories by typing:

    cd /usr/etc/install/sun386

2. Execute the script:

    sun386client

3. Enter the YP domain in which you are installing the diskless Sun386i client.

   `Enter the YP domain for this client [YP.caveman.com]:` [Return]

4. Enter the host name of the diskless Sun386i system.

   `Enter the host name of this client:` **pebbles**

5. Enter the IP address of the diskless Sun386i system.

   `Enter the IP address of this client:` **121.2.121.212**

6. Enter the Ethernet address of the diskless Sun386i system.

   `Enter the ethernet address of this client:` **9:8:76:5:4:3C**

   You can find out the Ethernet address of the system by powering it on and observing the power-on banner.

7. Enter the host names of any and all machines on which the client's home directories reside (as explained in "A Note About Home Directories" on the previous page).

   `Enter the host name(s) of the home directory server(s):`
   fred barney

   You will see the following messages:

   `Updating Yellow Pages hosts map for domain software ...`

   `Updating Yellow Pages ethers map for domain software ...`

   `Updating Yellow Pages bootparams map for domain software ...`

   `Updating Yellow Pages auto.master map for domain software`
   `...`

   `Updating Yellow Pages auto.vol map for domain software ...`

   `Updating Yellow Pages auto.home map for domain software ...`

   `Creating 16M bytes of swap for client pebbles ...`

   `Creating root for client pebbles ...`

Repeat this procedure for every diskless Sun386i client you want to install on the Sun-3 or Sun-4 server. For example, to install bambam and dino as clients of `fred`, you need to run the `sun386client` script two more times.

When you have finished running the script, you should boot the diskless Sun386i system by turning on the power switch.

Note: There is a bug in the `sun386client` script which can initialize diskless Sun386i clients with an incorrect time zone.

The problem appears only in the Central, Mountain, and Pacific time zones of the U.S. The time zone of the client is set to GMT (Greenwich Mean Time) instead of CST, MST, or PST (Central, Mountain, or Pacific Standard time zones).

If this happens you must reset the time zone on the client. To do this, on the client system log on as root and type the following commands:

rm /etc/localtime
ln −s /usr/share/lib/zoneinfo/*timezone* /etc/localtime
tzsetup

where *timezone* is one of the following:

♦ CST6CDT— Central Standard time zone

♦ MST7MDT—Mountain Standard time zone

♦ PST8PDT—Pacific Standard time zone

Loading Clusters Manually

If you need to load Application SunOS or Developer's Toolkit clusters after running the `sun386server` script, you must follow this manual procedure for each cluster.

1. Become superuser on the Sun-3 or Sun-4 server machine by typing **su** and then entering the root password.

2. Change your current working directory to the directory where the clusters are loaded.

   For Application SunOS clusters, this directory is:
   clusterpath/`sun386.sunos4.0.1/appl`

   For Developer's Toolkit clusters, this directory is:
   clusterpath/`sun386.sunos4.0.1/devel`

   In these paths, *clusterpath* represents the path where optional clusters are loaded. This is the same path you specified in step 8 of the "Running the `sun386server` Script" section, earlier in these notes.

3. Load the appropriate tape (Application SunOS or Developer's Toolkit) in the tape drive.

4. Wind the tape to the point that contains the cluster you want to load (use the commands in the table below):

	Local	**Remote**

From 1/2" Tape Drive	`mt -f /dev/nrmt8 rew`
	`mt -f /dev/nrmt8`
	`        fsf` *pos*
		`rsh` *host* `mt -f /dev/nrmt8 rew`
		`rsh` *host* `mt -f /dev/nrmt8`
		`        fsf` *pos*

From 1/4" Tape Drive	`mt -f /dev/nrst8 rew`
	`mt -f /dev/nrst8`
	`        fsf` *pos*
		`rsh` *host* `mt -f /dev/nrst8 rew`
		`rsh` *host* `mt -f /dev/nrst8`
		`        fsf` *pos*

In the above table, *host* is the name of the remote system and *pos* is the position in the tape of the cluster in question. Check the list on the next page to find out the position of the clusters you want to load.

Application SunOS:

Cluster	Position
extended_commands	5
spellcheck	6
doc_prep	7
networking_plus	8
comm	9
name_server	10
accounting	11
disk_quotas	12
audit	13
advanced_admin	14
mail_plus	15
old_commands	16
plot	17
sysV_commands	18
man_pages	19
games	20

Developer's Toolkit:

Cluster	Position
base_devel	2
sysV_devel	3
proflibs	4
plot_devel	5
sccs	6
sunview_devel	7
help_guide	8
dos_net_toolkit	9
config	10

5. Extract the cluster from the tape by typing the appropriate command from the table below. Only 1/4-inch tape is available at this time.

	Local	Remote
From 1/2″ Tape Drive	`tar xvbf 64 /dev/nrmt8`	`rsh host dd if=/dev/nrmt8` `ibs=32k obs=20b` `| tar xvpBbf 20 -`
From 1/4″ Tape Drive	`tar xvbf 64 /dev/nrst8`	`rsh host dd if=/dev/nrst8` `ibs=32k obs=20b` `| tar xvpBbf 20 -`

where *host* is the name of the remote system. The cluster is now loaded. Repeat steps 2 through 5 for each cluster you want to load.

Removing a Sun386i Client

At the present time, there is no command to remove a diskless Sun386i client from a server that is not a Sun386i workstation. However, you can follow manual steps to remove a Sun386i diskless client. The following example shows the removal of the Sun386i client named `pebbles` that was added previously.

1. Halt the client. Log in to `pebbles` and select the **Exits⇒Shutdown** option from the main menu.

2. Become superuser on the server (in this case, Sun-3 system `fred` is the server) by typing **su** followed by the root password.

3. Remove the client's link in `/tftpboot`.

   Each diskless system's link in `/tftpboot` has the format *inetaddr*.`S386` where *inetaddr* is the hexadecimal (base 16) representation of the client's IP address. The IP address for `pebbles` is `121.2.121.212`. The hexadecimal representation is derived as follows:

121	2	121	212	decimal IP address for pebbles
↓	↓	↓	↓	
79	02	79	D4	hexadecimal representation

   You can remove the link for `pebbles` in `/tftpboot` with the following command:

   rm -f /tftpboot/790279D4.S386

4. Unexport the client's swap and root files and remove their entries from `/etc/exports`.

   The root and swap file entries for `pebbles` in `/etc/export` will look like:

   ```
   /export/root/pebbles  -root=pebbles,access=pebbles
   /export/swap/pebbles  -root=pebbles,access=pebbles
   ```

   Unexport these files by typing the following commands:

   exportfs -u /export/root/pebbles
   exportfs -u /export/swap/pebbles

   Edit the `/etc/export` file and delete the root and swap entries for `pebbles`.

   Note: The files `/export/root` and `/export/swap` are the root and swap paths specified in steps 6 and 7 of the "Running the `sun386server` Script" section (earlier in these notes). You may have specified different root and swap paths.

5. Remove the client's root and swap areas. Type the following commands:

   rm -rf /export/root/pebbles
   rm -f /export/swap/pebbles

6. Log in to the Yellow Pages master server and become superuser.

   Type **su** followed by the root password. (In this case, Sun-3 system `fred` is the Yellow Pages server—superuser is already logged on.)

7. Remove the client's entry in `/etc/hosts`.

   The entry for `pebbles` in `/etc/hosts` on the Yellow Pages master server will look like:

   ```
   121.2.121.212 pebbles
   ```

   Remove this line.

8. Remove the client's entry in /etc/ethers.

    The entry for pebbles in /etc/ethers on the Yellow Pages master server will look like:

    ```
    7:8:76:5:4:3c pebbles
    ```

    Remove this line.

9. Remove the client's entries in /etc/bootparams.

    The entry for pebbles in /etc/bootparams on the Yellow Pages master server will look like:

    ```
    pebbles       root=fred:/export/root/pebbles \
                  swap=fred:/export/root/pebbles \
                  usr=fred:/export/exec/sun386.sunos4.0.1
    ```

    Remove these three lines.

10. Update the Yellow Pages maps for hosts, ethers, and bootparams.

    The Yellow Pages maps on the server are in /var/yp. Update them by typing the following commands:

    ```
    cd /var/yp
    make hosts ethers bootparams
    ```

This concludes the manual procedure to remove a Sun386i client.

Files Modified by the Server Kit ,

Several files are modified by the server kit scripts when you run those scripts on the server system. These files are listed in this section.

Files modified by the sun386server script:

♦ The /etc/exports file is modified to export the executable path, the cluster path, the local path, and the help path. See the "Running the sun386server Script" section earlier in these notes for more information on these paths.

♦ The /etc/inetd.conf file is modified to run tftp (only if the system isn't already a boot server)

♦ The /tftpboot directory—Sun386i boot programs are copied to this directory.

Files modified by the sun386client script (on the server):

♦ In the swap path a swap file is created for the client: *swappath/client*

♦ In the root path a root file system is created for the client: *rootpath/client*

    where *rootpath* and *swappath* are the root and swap paths you supply to the sun386server script, and *client* is host name of the client being added. In the example in the "Running the sun386server Script" section, the files created would be /export/swap/pebbles and /export/root/pebbles.

♦ The /etc/exports file is modified to export the swap file and root file system to the client.

♦ A link file is added to the /tftpboot directory. The link file, whose name is the IP address of the client (in hexadecimal representation) provides a link to the Sun386i boot program.

Files modified by the sun386client script (on the Yellow Pages master server):

Note: In most cases the server and the Yellow Pages master server are the same system.

♦ The /etc/hosts file—The client's name and IP address are added.

♦ The /etc/ethers file—The client's name and Ethernet address are added.

♦ The /etc/bootparams file—Entries to specify the client's root, swap, and /usr directories are added.

♦ /etc/auto.master, /etc/auto.home, and /etc/auto.vol—These files are automount maps. They are created unless your Yellow Pages master server is a Sun386i workstation (in which case they already exist).

♦ /var/yp/Makefile—If your Yellow Pages master server is a Sun-3 or Sun-4 system, this file is modified to include entries for the automount map files above.

Sun386i Developer's Notes

The notes in this section apply to the Developer's Toolkit software and the Sun386i Developer's Toolkit Documentation Set. Other information intended for developers includes the *Sun386i 4.0.1 man Page Supplement* (see your Sun sales representative if you didn't get a copy and want one) and *Sun386i Developer's Guide Replacement Pages* (these are the same replacement pages included with the 4.0 version of the *Administrator's & Developer's Notes*; you may ignore them if you've updated your *Sun386i Developer's Guide* with the earlier set). You should also review the *Sun386i Owner's Bulletin for SunOS 4.0.1* for a look at the user-level release notes shipped with each Sun386i system. If you're upgrading your software from the 4.0 version, see *Installing Sun386i SunOS 4.0.1*. In particular, check the "Summary of Changes" section and the "Compatibility" section of the installation notes.

Hardware

Using a mounted file system on the diskette drive – The diskette drive is a slow device. It is inadvisable to use scripts that mount the diskette, copy a file to it, and then unmount it. If you do use a script to copy files, run `sleep(1)` for about 20 seconds before unmounting the diskette. This does not apply to cases where you copy files with the `tar(1)` or `bar(1)` commands, but applies only to instances when you use a mounted file system on the diskette.

System Software

Capabilities not included by default – The default kernel (the one that resides in the root file system for each system) is configured for the system on which it runs. For most customers, this kernel includes all the functionality that they will need. Several SunOS features not included in the default kernel are:

Diskful Systems:

♦ disk quotas (`quota`)

♦ system accounting (`sysacct`)

♦ C2 security auditing (`sysaudit`)

♦ Secure NFS support (`crypt`)

♦ TCP debugging code (`tcpdebug`)

Diskless Systems:

♦ disk quotas (`quota`)

♦ system accounting (`sysacct`)

♦ C2 security auditing (`sysaudit`)

♦ TCP debugging code (`tcpdebug`)

♦ Secure NFS support (`crypt`)

♦ file system code for local disks (`ufs`)

♦ NFS server code (`nfsserver`)

♦ SCSI driver code (`wds` and `sd0`)

If you need any of these capabilities, you can build a new kernel by loading the `base_devel` and `config` subset clusters, which are part of the Developer's Toolkit

software. The configuration files for the kernel are found in /sys/sun386/conf. You can find directions for rebuilding kernels in *System and Network Administration* (included in the Sun386i Owner's Supplement Documentation Set).

Security enhancements – Several security flaws in the SunOS system were exposed by a widely-publicized virus (technically a worm) that attacked some systems on the DoD Internet. Though the Sun386i system was not affected by this program, the security flaws that it exploited in the background administration programs sendmail and fingerd have been eliminated in this release.

Changing the block size of a disk – When you change the block size of a disk, the file system block size must be greater than or equal to the system page size (4 Kbytes). A file system with a block size of 2 Kbytes is unsupported because the Sun386i page size is 4 Kbytes. A file system with a block size equal to or greater than the system page size is acceptable.

Core files appear to be very large – Core files of dynamically linked programs produced by SunOS 4.0 or 4.0.1 appear to be very large, having a length over 2 Mbytes in many cases (as shown by an ls -l command). However, the actual size on disk (as shown by an ls -s) is often much less, usually about the same as it would be with previous versions of the system. Such unusually large files contain one or more "holes" of unused space.

If such files (or for that matter, any file containing a hole) are copied using a command such as cp or tar, the holes will be filled in and as a consequence the file will actually occupy the amount of disk space indicated by its length. This is not a problem if the files are renamed (via mv) or dumped with dump.

Avoid write-only mmap segments – Programs that map segments with PROT_WRITE but do not contain a corresponding PROT_READ will dump core. Always use both protections together.

Signal handler for divide by zero – The program counter does not advance when a program that has a signal-handler for divide by zero encounters a divide-by-zero error. The program loops, repeatedly calling the signal-handler and returning to the divide by zero.

filec requires you to disable scrolling – The file completion mechanism, filec, does not work in a Command Tool window unless you select the Disable Scrolling option from the Command Tool window menu. This turns the Command Tool window into a shelltool and permits you to use filec.

quotactl does not work on /dev/root* – The disk quota utilities (quotaon, quotaoff, edquota, repquota) do not work in conjunction with the /dev/root pseudo-device. If you wish to use disk quotas, you must modify the entries in your /etc/fstab file to use the actual block device name. Your /etc/fstab file probably looks something like this:

```
/dev/roota / 4.2 rw 1 1
/dev/rootg /usr 4.2 ro 1 2
/dev/rooth /files 4.2 rw 1 3
```

If sd2 (the disk in your system unit) is your main disk, then you must change the /dev/root entries to /dev/sd2 to use quotas, like this:

```
/dev/sd2a / 4.2 rw, quota 1 1
/dev/sd2g /usr 4.2 ro 1 2
/dev/sd2h /files 4.2 rw, quota 1 3
```

After you have modified /etc/fstab in this manner, you must reboot your system for these changes to take effect.

SunLink DNI driver change – If you're using version 6.0 of SunLink DNI, you must use adb to modify the DNI driver, due to a change in the 4.0.1 Ethernet driver internals. As superuser, do the following steps:

1. **cd /usr/sunlink/dni/sys/sun386**
2. **adb -w dni_driver.o**
3. **dni_swap_type?W 1**
   dni_swap_type: 0x0   =   0x1
4. Press ⌊Control-D⌋ to exit from adb.
5. Rebuild the kernel according to the directions in *System and Network Administration* (included in the Sun386i Owner's Supplement Documentation Set). You must have the base_devel cluster of the Developer's Toolkit software loaded to rebuild the kernel.

Building kernels on diskless Sun386i systems – Because the diskless client mounts the server's /export/cluster/sun386.sunos4.0.1 directory read-only, you cannot configure and build a kernel on a diskless Sun386i system. However, if your diskless client's file server is a Sun386i system, then you can build your kernels on the file server for the diskless client. If your diskless client's file server *is not* a Sun386i system, or you still wish to build a kernel on your diskless client, you can use the following procedure.

Warning: Performing this procedure enables the diskless client to operate as root in the file server's /files partition. This procedure must be performed by root on both the diskless client and file server.

On the file server:

1. Edit the /etc/exports file to change access for the /export/cluster/sun386.sunos4.0.1 line to read:

   -root=*client*, access=*otherclients*

   where *client* is the diskless client you want to build the kernel on, and *otherclients* are other diskless clients using the same server.

2. Type **exportfs -av**.

On the diskless client:

1. Edit the /etc/fstab file to change mount permissions for /usr/cluster from ro to rw.
2. Type **umount /usr/cluster; mount /usr/cluster**
3. Follow the steps for building kernels in the *System and Network Administration* manual—see Chapter 9, Reconfiguring the System Kernel. (This book is in the Sun386i Owner's Supplement Documentation Set.) You must have the

base_devel cluster of the Developer's Toolkit software loaded to rebuild the kernel.

5.25-inch diskette support – The kernel now supports 5.25-inch diskettes, enabling the use of such commands as tar(1), bar(1), and fdformat(1) on these diskettes. The Sun386i system includes software driver support only; you must contact your Sun sales representative to find out about purchasing a 5.25-inch drive that will work on this system.

Utilities, Libraries, and Include Files

Program exit value – Some SunOS commands and utilities incorrectly exit with a nonzero value on successful completion. Of these programs, virtually all those that are commonly used in shell scripts have been fixed, but you may still encounter some. Their random-valued exit codes could cause problems for shell scripts that rely on the exit value of a program in a conditional statement.

Accessing shared libraries – By default, programs are built to access shared libraries. This is the standard behavior and can only be changed through the use of the –Bstatic flag. See the ld(1) man page for more information.

Shared memory segment change – The definition for the largest shared memory segment has been changed from SHMPOOL in Release 3.x to SHMSIZE in the SunOS 4.0 system.

sscanf function – The sscanf function does not properly handle 0. as a valid floating point string. Use 0.0 instead.

monitor() function – The monitor() function is in the profiling version of the C runtime startup module, /usr/lib/mcrt0.o, and is called during startup in all programs that link with mcrt0.o. Prior to SunOS 4.0 this function was also part of the C library, but this is no longer the case. This change is permanent.

The time command – When your system is heavily loaded, the way the system keeps track of program time can become distorted. As system activity gets heavier, particularly with increased DMA traffic, the distortion increases. This affects the output of both /bin/time and the C-shell's built-in time command, which report the time taken by system processes. However, the time of day kept by the system remains correct.

tip can hang commandtool – If you try to transfer a large file using tip in a Command Tool window, the transfer stops and the window hangs, forcing you to quit the window and reboot the system to use tip again. To avoid this problem, use tip only in a shelltool.

New 80386-format fonts – System-supplied fonts in /usr/lib/fonts/fixedwidthfonts are now available for both 80386 and 680x0 architectures. The fonts with the .i386 extension are in the new 80386 format. As of 4.0.1, dynamically linked programs that specify one or more fonts in this directory will automatically use the new 80386-specific fonts. Statically linked programs built under 4.0.1 will also use

80386 fonts. The 80386-style fonts provide enhanced performance, while the 680x0-style fonts are provided for backward compatibility.

Do not add the .i386 extension to any font file name in any existing or new program that you write. The 4.0.1 version of the libpixrect.a library automatically appends .i386 to a font file name before trying to open it so that if the specified font exists in the 80386 format, that will be the one used. If libpixrect.a cannot open the file, it attempts to open the originally specified font file name (minus the .i386 extension). In this way existing static programs continue to run, while new static programs or dynamically linked programs gain the performance benefits of using the 80386 fonts.

Statically linked programs that use 680x0-format fonts on the Sun386i system will still run as of 4.0.1. However, you should relink 4.0 statically linked programs that use these fonts for two reasons:

♦ The 80386-format fonts run faster, and so increase the performance of your programs.

♦ The 680x0-format fonts will be phased out sometime in the future.

If you are an applications supplier, it is particularly important for you to relink statically linked programs that are part of your application before your next release. As with dynamically linked programs, do not change the font file names in your code before relinking statically linked programs. When you relink, libpixrect.a automatically reads in 80386-format fonts.

fontedit(1) changes – As of 4.0.1, all fonts created with fontedit(1) will be in 80386 format, and fontedit will automatically append the extension .i386 to font file names as it creates the new font. If you use fontedit to modify any of the system-supplied fonts in /usr/lib/fonts/fixedwidthfonts, all of which are available in both 680x0 and 80386 formats, fontedit changes the 80386 version. Do *not* use the .i386 extension when specifying a font. fontedit uses libpixrect.a to open fonts, and libpixrect.a appends .i386 to the font name you specify and tries to open that file. If a font file with the .i386 extension does not exist, libpixrect.a tries to open the font file name that you supplied. fontedit then creates an 80386-format font as output and appends .i386 to the file name.

To specifically edit the 680x0-format version of a font that is also available in 80386-format, you must rename the 680x0-format file to a temporary file name and then use the new file name as input to fontedit. fontedit will produce an 80386-format font as output. You can use the fontflip_to_68k(8) utility (see below) to create a 680x0-format version of the font.

fontflip_to_68k and fontflip_to_i386 – Two new utilities let you maintain font formats for both the 80386 and 680x0 architectures. fontflip_to_68k(8) converts any 80386-format font to 680x0 format. Similarly, to convert any 680x0-format font to 80386 format, use the fontflip_to_i386(8) utility. Both utilities share the same man page, which provides details on their use.

If you use fontedit(1) to change any font, the output produced will be in 80386 format. To apply the changes made to an existing 680x0-format font, use fontflip_to_68k.

By convention, 80386-style fonts have a .i386 extension, while 680x0-style fonts have no extension. fontflip_to_68k tries to strip .i386 from the end of the font file name. If the input font file does not have a .i386 extension, then you must include the −o switch in the command line and specify an output file name.

Determining font format – To determine whether a font is in 680x0 or 80386 format, use the `file(1)` command with the font file name as an argument. The `file(1)` command indicates either:

```
vfont — 68k style

sun386i vfont — 386 style
```

Language Tools

Passing assembler files through C preprocessor (cpp) – If you pass an assembly language source file through cpp, make sure that the `.file` line is the first non-empty line of the file to ensure that it assembles correctly. The assembler must see the `.file` line before any other nonblank line in the file. You may also use the −P switch with cpp. If you use the −P switch, preprocessor macros may precede the `.file` line.

Debugging Tools

enum, struct, and union arguments to whatis command in dbx – If you enter the commands whatis enum, whatis struct, or whatis union in dbx, it refuses to accept any further commands, including quit. The only way out is to kill the process.

Debugging running processes with dbx – The option that enables dbx to debug a process that is running, such as a daemon, does not work in 4.0.1. If you try to issue the command dbx *objectfile process-id* at the system prompt or the debug *objectfile process-id* command from within dbx, dbx displays an error message stating that the process isn't running.

adb: single stepping over floating point instructions – Single stepping over floating point instructions with adb sometimes doesn't work because the PC does not advance correctly. Instead of single stepping, use breakpoints to get around the floating point section.

kadb: debugging module entry points – You can use kadb to debug the module entry point routine. You must set a breakpoint on the kernel routine vd_entry. This is the routine which calls the module entry point routine. When kadb hits the breakpoint, the symbols for the module are usable and you can set a breakpoint in the module itself.

Care must be taken to remove kadb breakpoints before unloading modules. Since kadb inserts bpt instructions in the module itself, unloading and loading new modules while breakpoints are set can cause kadb to insert bpt instructions at incorrect places. This may cause the system to crash.

Window System

After the General window system notes presented below, the remaining notes in this section are organized by SunView application. See also the "Utilities, Libraries, and Include Files" section earlier in these notes for information about new 80386-format fonts and related utilities.

General

SunView background color – The default SunView Desktop background color has been set to blue. This color can be changed on a per-session basis using Color Editor. It can be changed permanently by specifying a different argument to the sunview −color command line in a user's `.login` file. Removing this command line argument altogether causes the screen to revert to medium gray, as in 4.0. The superuser's default desktop color remains gray.

Keyboard software changes – The Sun386i system previously supported 8-bit character codes only. As of 4.0.1, only 16-bit character codes are permitted. This change should only affect you if you recompile any program that uses the KIOCSETKEY or KIOCGETKEY ioctl calls; if you recompile, the binary may not run correctly on 4.0 or 4.0.1 systems.

If you need to recompile a program that uses KIOCSETKEY or KIOCGETKEY ioctl calls, change the ioctl types in your program to KIOCSKEY and KIOCGKEY, respectively. You must also change your structure declaration from kiockey to kiockeymap. The difference between the structures is that the kio_entry field in kiockeymap is larger to accommodate 16-bit values.

Num Lock key enabled in all SunView windows – When you press the Num Lock key and subsequently press any other key on the right keypad, the upper character shown on the keycap is generated instead of a function key sequence. This is true for all SunView windows, not just DOS Windows, as was the case for 4.0. If you included a workaround in a 4.0 program to enable Num Lock to work properly in all windows, you must remove the workaround code for Num Lock to work correctly under 4.0.1. You can also issue the disablenumlock command in any window to get Num Lock functionality for 4.0 programs that have a workaround.

Interposition on open/close events for subwindows within frames – When a frame contains a subwindow such as a canvas or a panel, correct interposition on open/close events requires more than the example described in the *SunView 1 Programmer's Guide* (pages 297–298). In particular, the sample interposition function described there will not catch the case where the user presses the open/close key over an open canvas (or other subwindow) contained by the frame. You can correct this by looking for events of type ACTION_OPEN in the subwindow or by manually storing the frame's state and checking it each time the interposition function is entered, for example:

```
Notify_value
my_frame_interposer(frame, ie, arg, type)
    Frame               frame;
    Event               *ie;
    Notify_arg          arg;
    Notify_event_type   type;
{
    static int          closed_before = UNKNOWN;
    int                 closed_now;
    Notify_value        value;

    /* let frame operate on the event */
    value = notify_next_event_func(frame, ie, arg, type);
    /* get frame's current state */
    closed_now = (int)window_get(frame, FRAME_CLOSED);
    if (closed_before == UNKNOWN) {
```

```
                    /* if this is the first call, just record the */
                    /* frame's state */

                    closed_before = closed_now;
              } else if (closed_now != closed_before) {

                    /* else if the window state has changed, record */
                    /* the new state */

                    closed_before = closed_now;

                    /* do something, e.g.: */

                    printf("window just %s\n", closed_now? "closed" : \
                    "opened");

                    fflush(stdout);

              }

              return(value);

        }
```

Modifying keyboard maps – You can modify workstation keyboard mappings from that of the default U.S. keyboard via User Defaults (or by editing .defaults yourself), or by issuing the loadkeys(1) command. You can change key mappings to suit your preference, for example, to swap the functionality of the Escape and Caps Lock keys. Probably a more common modification will be to enable generation and display of non-U.S. characters, in scrollable windows only. For this reason, the specifics of changing key mappings are covered in the "Internationalization" section later in these notes.

Sun Organizer

4.0 .orgrc files incompatible – For performance reasons, changes were made in the 4.0.1 Organizer that are incompatible with the 4.0 .orgrc file. These changes relate to the color palette section. Previously, this section came at the end of the file and used color names; in 4.0.1 it appears at the beginning of the file and uses RGB values. This change affects your application if it adds colors to the .orgrc file. Your application must now add colors to the beginning of the file and specify RGB values, not color names. See the .orgrc man page (included in the *Sun386i 4.0.1 man Page Supplement*) for detailed file format information.

DOS Windows

New developer's toolkit cluster for PC-NFS programmers – PC-NFS™ Programmer's Toolkit (dos_net_toolkit) is a new cluster in the Sun386i Developer's Toolkit. It is a collection of network libraries and facilities that you can use to write distributed open network applications for PCs and Sun386i DOS Windows.

A PC-NFS Toolkit application running in a DOS Windows session can interact with another application that uses either Remote Procedure Calls (RPC) or the 4.2BSD socket interface. The application must be built on either the TCP or UDP protocol and run on a system that supports RPCs or 4.2BSD, such as:

◆   A DOS Windows session (on the same Sun386i workstation or a remote one)

◆   A SunOS session (on the same workstation or a remote one)

◆   A PC (or 100% compatible PC)

◆   Other operating systems that support 4.2BSD or RPCs

The PC-NFS Toolkit libraries include 4.2BSD socket calls and RPC client calls. If you already have a PC-NFS Toolkit application for a PC, you can run it in DOS Windows after recompiling the source file under DOS Windows.

To write toolkit applications you will need a copy of the *PC-NFS Programmer's Toolkit Manual* . Call the Sun Telemarketing Department to order a manual or to find out more about writing distributed applications for DOS. The toll-free number is 1-800-334-SUNM. The order number for the *PC-NFS Programmer's Toolkit Manual* is PC-NFS-PTK-09.

5.25-inch drive support – The DOS monitor and DOS BIOS now support both 3.5-inch and an external 5.25-inch drive. However, the Sun386i system comes only with a 3.5-inch drive. To get an external 5.25-inch drive that will work on this system, contact your Sun sales representative.

User-related DOS changes and additions – You should also review the DOS section of the *Sun386i Owner's Bulletin for SunOS 4.0.1* for a description of issues that affect both users and developers. This section includes information on topics such as LIM 4.0 expanded memory support, per-drive file sharing, faster DOS keyboard and mouse response, 9600-baud serial support, and faster DOS interrupt response.

Help Viewer

New help_open utility – 4.0.1 includes a new utility, /usr/bin/help_open, for sending commands to the Help Viewer via RPC. See the help_open man page for details.

-file command line argument – The -file command line argument, while still acceptable, is no longer necessary. Both of the following commands cause help_viewer to start with the document /home/ahinkle/somefile.doc:

```
help_viewer /home/ahinkle/somefile.doc

help_viewer -file /home/ahinkle/somefile.doc
```

/tmp/help_viewer.lock – 4.0.1 prevents users from starting more than one help_viewer process per machine. This is accomplished by locking and unlocking the file /tmp/help_viewer.lock .

admin_policies keyword eliminated – Because the *SNAP Handbook* topic Implementing Administrative Policies has been eliminated in 4.0.1, the keyword referencing this topic, admin_policies, has also been removed from the sun_external.info file.

Size of bitmap memory pool reduced – For performance reasons, the size of the bitmap memory pool was reduced from 200 Kbytes in 4.0 to 65 Kbytes in 4.0.1. The bitmap memory pool is the amount of memory Help Viewer allocates for bitmaps (images) in Frame Maker™ documents. This value is set in the file ../../format/Frame/init/bitmaps (relative to the default help directory). Setting it higher allows more or larger bitmaps to appear in documents; setting it lower reduces Help Viewer size and thus improves performance. If you see a message like this:

```
help_viewer: (FrameMaker) no room in memory pool for raster
image filename
```

```
help_viewer: (FrameMaker) you may need to change this file:
'/vol/help/language/USA-English/../../format/Frame/init/
bitmaps'
```

it means the memory pool is too small. The `bitmaps` file can be edited to increase the memory pool size.

Help Viewer font changes – The 4.0.1 Help Viewer font hierarchy has been modified slightly, allowing it to take advantage of pre-processed, bit-flipped fonts for better performance. Also for performance reasons, fewer Frame Maker fonts are used by the standard set of Help Viewer handbooks in 4.0.1; therefore, fewer fonts are shipped on disk and loaded by Help Viewer. These changes affect you if you have developed on-line documentation using one of the fonts no longer shipped. The section "More About Help Fonts" later in these notes presents a detailed description of the font changes and new hierarchy, plus instructions for adding fonts not shipped with the standard system.

Internationalization

You can change key mappings from the default setting of a U.S. keyboard for current and future sessions on a per-workstation basis. The `loadkeys`(1) command and a User Defaults option each let you change the mappings for the workstation (not just for the user who issued the command) until the system is rebooted. You can also change and retain the nondefault mappings between reboots. This section describes both methods.

The key tables and commands provided are part of a phased-in approach to international support. Please note the following:

ISO characters – You can only generate ISO characters in a scrollable window. That is, if you press a key that is mapped to an ISO character, you will only see the ISO character in a scrollable window. However, you can view ISO characters that are in key tables in either a Shell Tool or Command Tool window.

DOS Windows – If you use any key table other than the U.S. default, you will probably be unable to use DOS Windows because it does not interpret the mappings correctly. For this reason, you might want to use the Compose key instead to generate international characters.

The key tables provided with the Sun386i system in the `/usr/share/lib/keytables` directory are:

◆ `belgium_france`

◆ `canada`

◆ `denmark`

◆ `germany`

◆ `italy`

◆ `netherlands`

◆ `norway`

◆ `portugal`

◆ `spain_latin_america`

◆ `sweden_finland`

◆ `swiss_french`

◆ `swiss_german`

◆ `us`

◆ `uk`

You can also edit these files or create your own. The `keytables(5)` man page describes the table layout. Before editing or creating a file in the directory `/usr/share/lib/keytables`, you must first remount the `/usr` partition as superuser with the commands:

`mount -o remount,rw /usr`

If you load the key table file for the United States, United Kingdom, Belgium/France, Germany, Switzerland/France, or Switzerland/Germany, you can use the keyboard illustrations on pages 122–124 of *System Setup and Maintenance* (one of the manuals included in the Sun386i Owner's Documentation Set). Keyboard illustrations for the other key tables provided are not yet available, so you should take careful note of key locations *before* you specify another table. This is important because when you change key mappings, the new mappings automatically remain in affect until you reboot the system. When you or someone else using the workstation log out, it could be difficult to log in again and get the password correct if you can't see what you are typing and if the keys pressed are different from their key caps. Therefore, you might want to:

1. Print the table file you plan to use.

2. Compare the hard copy to the on-line version of the table and mark any ISO characters that didn't print on the hard copy.

3. Look at Figure 10-1 on page 153 of the *Sun386i Developer's Guide* which shows the key station codes for the default keyboard. (This manual is part of the Sun386i Developer's Toolkit Documentation Set.)

4. Type **`/etc/fastboot`** to restore read-only permission to the `/usr` partition.

If you have trouble determining current key mappings, see the "Specifying the Default Key Table" section later in this description.

Changing the Table for Current and Future Sessions

The `loadkeys(1)` command lets you change key mappings immediately. Entering `loadkeys` *filename* takes the table specified by *filename* and changes the key mappings accordingly. If you type `loadkeys` without an argument, the system checks the dip switch settings on the bottom of the keyboard and reads in the appropriate `layout_`*nn* file from `/usr/share/lib/keytables`. If you create your own key table in a directory other than `/usr/share/lib/keytables`, then you must specify the full pathname as an argument. The workstation uses the most recently specified table until you have to reboot the system. When you or someone else log out and log back in, you must use the correct characters according to the current key table.

The `loadkeys(1)` command replaces the `setkeys(1)` command used in the SunOS 4.0 system. To display the currently loaded key mappings, use the `dumpkeys(1)` command.

You can also use User Defaults to reset the key mappings until the system is rebooted:

1. Display the Desktop menu by pressing the right mouse button on the background portion of the screen.

2. Drag right on **Services** and select **User Defaults**.

3. Go to the Input category either by cycling through the selections or selecting it from the User Defaults menu.

4. Enter the file name of the key table as a value for the **Keyboard_Type** parameter.

5. Select **Save** at the top of the window.

Specifying the Default Key Table

If after loading a different key table you cannot tell which characters are mapped to which key stations, you can recover by typing **loadkeys us**. This resets the keyboard to the default (U.S.).

If you cannot even locate the keys required to enter the **loadkeys us** command, try changing the setting via User Defaults as described in the previous section. In step 4, you can either enter **us** or simply delete the value for the **Keyboard_Type** parameter. When no file is specified and you press **Save**, the system uses the default U.S. key mappings.

If for some reason you are unable to do either of these procedures, for instance if you can't even log in, you can reboot the system. During a reboot the system automatically remaps the keys to the default U.S. values, provided you have not edited the `/etc/rc.local` file to specify the loading of a different key table. The next section provides more information on editing `/etc/rc.local`.

Retaining a Nondefault Table Between Reboots

If you always want to use a nondefault key table on a Sun386i system, as superuser include one of the following two lines in the `/etc/rc.local` file:

loadkeys   (to always use the keyboard dip switch settings)

loadkeys *filename*   (to always read the file specified)

Documentation Changes

Sun386i Developer's Guide **replacement pages** – Replacement pages for the *Sun386i Developer's Guide* are included with the Sun386i Developer's Toolkit Documentation Set, which you receive if you purchase Sun386i Developer's Toolkit software. (These are the same replacement pages that were packaged with the *Administrator's & Developer's Notes* for SunOS 4.0.) The pages are three-hole punched so that you can easily insert them into the manual if you haven't already. There are several errors in these pages; see the following notes.

/files/vol.local directory – Page 204 of the replacement pages for the *Sun386i Developer's Guide* incorrectly shows the directory `/files/vol.local`; it should be `/files/vol/local`.

/export/vol.local directory – Page 206 of the replacement pages for the *Sun386i Developer's Guide* incorrectly shows `/export/vol.local` as a link to `/files/vol.local`. Instead, `/export/vol/local` is a link to `/files/vol/local`.

/stand directory – Page 199 of the *Sun386i Developer's Guide* should include a description of the `/stand` directory. `/stand` is a directory that exists only on diskful systems and contains the standalone diagnostics files `boot.S386`, `copy`, and `tpboot.S386`.

loadkeys(1) replaces setkeys(1) – The `loadkeys(1)` command replaces the `setkeys(1)` command on page 152 of the *Sun386i Developer's Guide*. The path to key map tables shown on this page has also changed. Key map tables are now located in the `/usr/share/lib/keytables` directory.

mmap routine for Sun386i drivers – Page 91 of *Writing Device Drivers* shows an example of a driver mmap routine that is incorrect for the Sun386i system. The following code works on all Sun architectures:

```
#include <machine/pte.h>

fbmmap(dev, off, prot, numdevs, mb_devs, size)
        dev_t dev;
        off_t off;
        int prot, numdevs;
        struct mb_device **mb_devs;
        u_int size;
{

        struct pte pte;

        if ((u_int) off >= size)
                return (-1);

        mmu_getpte(mb_devs[minor(dev)]->md_addr + off, &pte);

        return (*(u_int *)&pte & PG_PFNUM);

}
```

***SunOS Reference Manual* inserts** – A package of insert pages for man pages accompanies the Sun386i Owner's Supplement Documentation Set. These include information on the following commands, new or updated for the Sun386i SunOS 4.0.1 release:

bar(1), bar(5)	load(1)
cc(1v)	loadc(1)
dos(1)	loadkeys(1)
dumpkeys(1)	modload(8)
fdformat(1)	modstat(8)
fontedit(1)	mount(8)
fontflip_to_68k(8)	organizer(1)
fontflip_to_i386(8)	orgrc(5)
getmntent(3)	rarpd(8c)
help(5)	rwhod(8)
help_open(1)	start_applic(8)
help_viewer(1), help_viewer(5)	strip(1)
input_from_defaults(1)	textedit(1)
ipallocd(8C)	unconfigure(8)
kadb(8s)	uucp(1c)
kb(4m)	vfont(5)
keytables(5)	ypsync(8)
ld(1)	

init – The printed version of the init(8) page incorrectly states that for a secure system, when the console is marked secure in /etc/ttytab, a root password is required before the system comes up in single-user mode. The on-line page correctly states that when the console is *not* marked secure, the root password is required.

cc(1) man page – The −a option (used to insert code to count how many times each basic block is executed) is not supported on Sun386i systems, only on Sun-2™, Sun-3, and Sun-4 systems. The on-line cc(1) man page does not explicitly state this.

screenblank(1) man page – An additional undocumented option to the `screenblank(1)` command is the `-l` option, which causes `screenblank` to display the Sun Microsystems logo when it closes the screen.

uucp(1) man page – The on-line man page mentions `/usr/lib/uucp/ADMIN`. This file does not exist. All the configuration files go into `/var/spool/uucp/sys`.

***C Programmer's Guide*: comparing a `long` to a `float`** – When comparing an unsigned `long` to a `float`, the Sun386i C compiler does not preserve the unsignedness in the comparison; the Sun-3 C compiler does.

Performance Tips

The Sun386i system has a large, virtual memory system. However, whenever an application touches a page in memory and brings it into its working set, this puts a load on the system that creates the potential for thrashing. There are a number of processes like sync and daemons running in the background for mail, mounting, routing, and other network services that constantly compete for memory. Also, many users run more than one application at a time. All processes must compete for the same physical memory.

One of the major goals of the Sun386i 4.0.1 release was to increase the performance of the system, particularly at the lower end of the product line. You can improve the performance of the programs you develop on the Sun386i system by following the suggestions below. (For specific C language guidelines, see the "Optimizing Code" section that starts on page 48 of the *Sun386i Developer's Guide*.)

♦ Always look for the lowest-level way to perform a task. For instance, use system calls instead of library calls, and library calls instead of SunView calls whenever possible (note that this could make porting more difficult). Avoid creating multiple processes, using pipes, and so on, to do things that you could accomplish with lower system overhead.

♦ Avoid tying up large amounts of memory that a program isn't currently using. For instance, allocate only the memory that is needed for a given task instead of allocating enough memory for several separate tasks.

♦ Where possible, use one system call and save commonly used data in a global variable. For instance, use the `getdtablesize(2)` system call to determine the maximum number of open files (used as a parameter for the `select` system call), and then store that data in a global variable for use by other routines. This will enhance your program's performance, but could make porting more difficult.

♦ If an application frequently searches linked lists, create a header table or a linked list of headers that contain pointers to data instead of having data reside in each element of a linked list. This is particularly important if each item in the list is large. Using a header table can result in many less pages being read into memory.

♦ Push temporary data onto the stack. This enables many different variables to share the same physical memory.

♦ Repainting a window in SunView requires that the process that owns the window be awakened and its pages brought into memory. Therefore, applications should automatically close after a specified time of inactivity. Closed applications take up less screen space, and therefore are repainted less frequently. The default Sun Organizer and Help Viewer applications automatically close to their iconic state after 10 minutes of inactivity.

♦ If your application creates more than one window, try to ensure that windows don't overlap by default.

♦ Avoid using clocks and other similar features that use up CPU time and physical memory with your application.

More About Help Fonts

This section lists the Frame Maker fonts that are shipped with the Sun386i system as of 4.0.1, and explains how you can add more fonts if you have purchased Frame Maker. The same Interleaf™ TPS fonts that shipped with 4.0 are still available as of 4.0.1. See the *Sun386i Developer's Guide* for details.

Standard Help Viewer Fonts

Unless otherwise indicated, all file names in this section are relative to:

　default_help_directory/../../`format/`

where the default help directory, as defined in `.defaults`, is typically:

　`/vol/help/language/USA-English/`

The Frame Maker fonts available with 4.0 in the directory `/vol/help/format/Frame/init/fontdir` have been reduced in number, moved to the directory `/vol/help/format/Frame/init/fontdir.sun386`, and bitflipped (using `/usr/bin/batchfont_create`, explained in the next section). The 4.0.1 `help_viewer` reads fonts from the `fontdir.sun386` directory instead of from the `fontdir` directory. Fonts shipped with 4.0.1 are:

　`Courier-Bold14.bfont`

　`Courier12.bfont`

　`Courier14.bfont`

　`Frame-Roman12.bfont`

　`Helvetica-Bold12.bfont`

　`Helvetica-Bold14.bfont`

　`Helvetica-Bold18.bfont`

　`Helvetica-Bold24.bfont`

　`Helvetica-Oblique14.bfont`

　`Helvetica12.bfont`

　`Helvetica14.bfont`

　`Symbol10.bfont`

　`Symbol12.bfont`

　`Symbol14.bfont`

　`Times-Roman12.bfont`

If you attempt to view a Frame Maker document (in `help_viewer`) containing fonts that `help_viewer` does not have loaded, `help_viewer` replaces these fonts with the default font, Times-Roman 12 point.

The batchfont File

4.0.1 `help_viewer` has been enhanced to enable it to read screen fonts from a specially constructed `batchfont` file. The `batchfont` file consists of Frame Maker screen fonts that have been flipped ahead of time and stored together in a single file. This allows `help_viewer` to map all its fonts at once and, since fonts are not altered,

to place them in a read-only portion of memory. This enhancement improves both start-up time and virtual memory paging performance.

If you look in the `fontlist` file in `fontdir.sun386`, you will see that all or most screen fonts have a new `.batchfont` extension. For example, 12-point Helvetica was specified as:

```
<ScreenFont      Helvetica12.bfont                2    0    12>
```

in 4.0. As of 4.0.1 it is specified as:

```
<ScreenFont      Helvetica12.bfont.batchfont  2    0    12>
```

To ensure backward compatibility, both formats are legal entries in the `fontlist` file. But when `help_viewer` sees a `.batchfont` extension, it looks for a file called `batchfont` and reads the font from it instead of from the file `Helvetica12.bfont`, as was the case in 4.0. This improves performance.

To see what fonts are contained in the `batchfont` file, type **more batchfont**. This displays a list of the font names, followed by the actual font binary data. Once you hit the binary data, quit out of `more`. (You could also use `head batchfont`—see the head(1) man page for details.)

You can create your own `batchfont` file by using the `batchfont_create` program in the `/usr/bin` directory. The syntax is:

```
batchfont_create batchfont fonts
```

where *fonts* is a list of all the fonts you want to be in the `batchfont` file, separated by spaces. You must include the names of the 15 default Frame Maker fonts shipped with the system (see the preceding page) to be able to view Sun386i on-line handbooks. The `batchfont` file (or a link to it) must be in the directory `../../Frame/init/fontdir.sun386` (relative to the default help directory as specifed in `.defaults`). Also, you must include entries in the `fontlist` file (in the same directory) for each font added. The next section explains how to add fonts.

Adding Frame Maker Fonts

If you have Frame Maker and your help documentation requires additional Frame Maker fonts:

1. Locate the fonts. You can probably find them in your `$FMHOME/.makerinit/fontdir` directory (see Frame Maker documentation for details).
2. Copy (or make symbolic links to) whatever screen fonts you want from `fontdir.sun386`.
3. Create a `batchfont` file as described in the previous section. This step is optional, but recommended for better `help_viewer` performance.
4. Tell `help_viewer` which fonts you want it to load by editing the `fontlist` file.
5. Quit and restart `help_viewer`.

The following sections provide details.

Locating Fonts

The Sun386i `help_viewer` Frame font directory is `format/Frame/init/fontdir.sun386`. This directory is relative to the default help directory, as designated in your defaults database. Typically, this is `/vol/help/format/Frame/init/fontdir.sun386`, which resolves to `/usr/lib/help/format/Frame/init/fontdir.sun386`.

Copying Fonts

If you aren't adding many fonts, copy the fonts that you want from `$FMHOME/.makerinit/fontdir` to the destination directory, probably to `/usr/lib/help/format/Frame/init/fontdir.sun386`. To add anything to `/usr` you must become superuser and remount the `/usr` partition to be writable (as shown in the example later in this section). You may also need to change the permissions in `fontdir.sun386` to make the files there writable as well. If there is not enough room in `/usr`, then you must put the additional fonts somewhere else (for example, leave them in `$FMHOME/.makerinit/fontdir`) and create symbolic links to them instead.

For each family of screen fonts, copy (or make a link to) the corresponding font-metric (`.bfm`) file if it is not already present in the destination directory. Each family of fonts has one `.bfm` file. For example:

<u>These screen fonts</u>	<u>Require this .bfm file</u>
`Helvetica7.bfont`	`Helvetica.bfm`
`Helvetica8.bfont`	
`Helvetica9.bfont`	
`Helvetica10.bfont`	
`Helvetica12.bfont`	
`Helvetica14.bfont`	
`Helvetica18.bfont`	
`Helvetica24.bfont`	

Editing the fontlist File

`help_viewer` loads the fonts specified in the `fontlist` file in the `fontdir.sun386` directory. In this file, you must tell `help_viewer` to load both the screen font and the `.bfm` font metric for each font you add. You can look at `fontlist` as a guide, either in the directory `fontdir.sun386`, `fontdir`, or in the Frame Maker directory `$FMHOME/.makerinit/fontdir`.

For instance, to add the screen font `Helvetica7.bfont`, which is not provided with the `help_viewer`, you would make these two entries in `fontlist`:

`<ScreenFont`	`Helvetica7.bfont`	`2`	`0`	`7>`
`<FontMetric`	`Helvetica.bfm`	`2`	`0`	`7>`

In the `fontlist` file, font families are kept together and organized in increasing order by size. The `fontlist` file in `fontdir.sun386` contains several screen fonts with a `.batchfont` extension, which means that `help_viewer` gets all of these fonts from the `batchfont` file. Note that there are no font file names that contain the `.batchfont` extension; this extension is used only in the `fontlist` file, to tell `help_viewer` to get the font from the `batchfont` file.

Font Addition Example

This section shows the steps to add Times-Roman 24 point to the current set of fonts. The Times-Roman font is `$FMHOME/.makerinit/fontdir/ Times-Roman24.bfont`. This example assumes that the rest of the fonts are physically located in `/usr/lib/help/format/Frame/init/fontdir`, the standard place for both 4.0 and 4.0.1, and that links to these fonts exist in `fontdir.sun386` (new as of 4.0.1).

1. **`cd /usr/lib/help/format/Frame/init/fontdir.sun386`**
2. **`su`**
3. **`mount -o remount /usr`**

4. `chmod a+rw . batchfont fontlist`
5. `exit`
6. `cp batchfont batchfont.old` (optional)
7. `rm batchfont`
8. `batchfont_create batchfont \`
   `../fontdir/Courier12.bfont \`
   `../fontdir/Courier14.bfont \`
   `../fontdir/Frame-Roman12.bfont \`
   `../fontdir/Helvetica-Bold12.bfont \`
   `../fontdir/Helvetica-Bold14.bfont \`
   `../fontdir/Helvetica-Bold18.bfont \`
   `../fontdir/Helvetica-Bold24.bfont \`
   `../fontdir/Helvetica-Oblique14.bfont \`
   `../fontdir/Helvetica12.bfont \`
   `../fontdir/Helvetica14.bfont \`
   `../fontdir/Symbol10.bfont \`
   `../fontdir/Symbol12.bfont \`
   `../fontdir/Symbol14.bfont \`
   `../fontdir/Times-Roman12.bfont \`
   `$FMHOME/.makerinit/fontdir/Times-Roman24.bfont`

   If there is no room for the new `batchfont`, create it somewhere else and make a link to it, for example:

   `batchfont_create ~/batchfont` *fontfile1, fontfile2, ...*

   `ln -s ~/batchfont ./batchfont`

9. If `Times-Roman.bfm` doesn't already exist in `fontdir.sun386`, enter the command:

   `ln -s $FMHOME/.makerinit/fontdir/Times-Roman.bfm`

10. `cp fontlist fontlist.old` (optional)
11. `textedit fontlist &`
12. Following the format of `fontlist`, add these lines to the appropriate sections of `fontlist` (see Frame Maker documentation for file format details):

    `<ScreenFont  Times-Roman24.bfont.batchfont  1      0      24>`

    `<FontMetric  Times-Roman.bfm                 1      0      24>`

13. Save `fontlist`.
14. Quit and restart `help_viewer` to view the new fonts.
15. Type **/etc/fastboot** to restore read-only permission to the /usr partition.

Index