sun microsystems

Installing UNIX *on the* Sun Workstation

Sun Microsystems, Inc. • 2550 Garcia Avenue • Mountain View, CA 94043 • 415-960-1300

Part No: 800-1158-01
Revision B of 15 May, 1985

Credits and trademarks

Revision History

Revision	Date	Comments
A α	3 December 1984	First release of this UNIX Installation Manual.
B β	1 February 1985	Added NFS-specific material, general update.
C	27 March, 1985	Added corrections discovered during β test.
D	5 April, 1985	Final Pass for Release 2.0.

Errata Notes for Installing UNIX

Part Number: 800-1242-01

The following pages list errata from *Installing UNIX on the Sun Workstation* (Part Number: 800-1158-01)

Errata in the 2.0 release of Installing UNIX on the Sun Workstation

Page(s)	Comments
4-3	PROM Revision Levels. Insert the following note on page 4-3, directly below the `diag>` prompt, and above Step 7: NOTE: If you selected a Xylogics 450 controller in Step 2, ***diag*** reports the Xylogics PROM revision level in the above display. If that Xylogics board is going to support a **dual-ported Eagle** disk, it MUST have level C.3 or C.5 PROMs.
1-4	***e_interface***. Change the bottom paragraph on page 1-4 to read: Throughout this manual, when we use ***tape, disk*** or ***e_interface***, you should replace it with the appropriate UNIX device name. When we use ***e_interface#***, you should follow the Ethernet device name by its controller number: 0 for controller 1, or 1 for controller 2.
1-10 and 7-1	Files on Release Tape. These pages contain tables which list the order of files on Sun-2 Release Tapes. Files 8 and 9 are reversed on both tables. Change **file 8 on tape 1** to read ***/usr/demo***, and change **file 9 on tape 1** to read ***/usr/games***.
4-7	Formatting a Disk Not Purchased from Sun. Add the following notes to the bottom of page 4-7: NOTE: To format a disk that you did not purchase from Sun Microsystems, you must use the SMD ***format*** command instead of ***fix***. Enter `format` to the `diag>` prompt, and when it displays the prompt `format>`, enter `format`. The format proceeds like the SCSI format described on page 4-5. NOTE: Regardless of whether you use ***fix*** or ***format*** we recommend 5 surface analysis passes.

Page(s)	Comments

4-11	Digression for Servers with Multiple Disks. Add the following lines to page 4-11 (near the top of the page):
	Right after the line:
	`/dev/xy1c /usr2/ 4.2 rw 1 2`
	add the lines:
	Next, run *makedev* for your second disk:
	`makedev xy1`
	and construct a new file system for the partition:
	`newfs /dev/rxy1c`

6-10	Minor Errors. Make the following changes to the text at the bottom of page 6-10: a) Delete the first "master" in the sentance: "If it is to be a master yellow pages master server ..." so that it reads: "If it is to be a yellow pages master server ..." b) In the list of commands to execute, delete: `# mount /usr` c) In the same list, change: `# ifconfig` ***e_interface hostname*** into: `# ifconfig` ***e_interface# hostname***
7-2	Extract Release. On page 7-2, at the bottom of the page, change the command line: **mysys# extract_release tapeless gala suntools graphics pascal** *to read:* **mysys# extract_release st tapeless gala suntools graphics pascal**

Contents

Contents

Tables

Chapter 1

Introduction to UNIX Installation

In this document we describe how to install UNIX† for the very first time on a Sun Workstation or a network of Sun Workstations. We assume that you have a workstation which has been set up according to the procedures in the Sun *Hardware Installation Manual* for the model, and that you have distribution software either on two nine-track half-inch tape reels or on three quarter-inch cassette tapes.

We also provide instructions for upgrading clients, servers, and diskless workstations to use the NFS. For additional upgrade information, please read the *Upgrading System Software* section in the *System Administration Manual.* The instructions for saving and rebuilding your existing root and *usr* file systems may prove especially useful.

Note that to perform the procedures in this manual, you need "rev N" or greater PROMs installed in your system.

1.1. Intended Audience

This document is written specifically for persons who are installing or upgrading Sun Workstations. It supports installation on the entire Sun-2 family of workstations, which includes the following models:

Sun-2/100U	Desktop workstation with a seven-slot Multibus card cage. Peripherals may include 1/2" or 1/4" tape drives, and 64, 130, or 380 MByte (formated) disks.
Sun-2/150U	Rackmountable workstation with a fifteen-slot card cage. Peripherals are identical to those for the 100U.
Sun-2/50	Node with a two-slot backplane, and optional memory expansion or floating point boards.
Sun-2/120	Deskside SunStation with a nine-slot card cage. Peripherals may include 1/2" or 1/4" tape drives, and 42, 71, 130 or 260 MBytes (formated) of disk storage.
Sun-2/120FS	Identical to the Sun-2/120 without monitor, video controller board, keyboard and mouse. This workstation is designed to be a file server, and uses a standard ASCII terminal as a console.
Sun-2/160	Color SunStation with color monitor, and a twelve-slot Eurocard (triple-high VME) card cage. Peripherals may include 1/2" or 1/4" tape drives; and 71, 142, 260, or 380 MBytes (formated) of disk storage.
Sun-2/170	Rackmountable SunStation with a fifteen-slot card cage. Peripherals may include 1/2" or 1/4" tape drives, and 130 or 380 MBytes (formated) of disk storage.

† UNIX is a trademark of Bell Laboratories.

1.2. Terminology and Context

The following subsections describe terms used in the installation procedures, and provide necessary background information. Since the procedures assume that you understand these terms, it's a good idea to read through the subsections even if you feel you already understand the terms.

1.2.1. Machine Types

In addition to differences in basic hardware, workstations differ in their relation to other machines on a local network.

Standalone

A standalone workstation has a complete root file sytem on its disk and does not require another machine to boot UNIX. It must have its own disk; it may or may not be attached to an Ethernet, and it may or may not have a local tape drive. But it does not rely on any other machines for storage of any sort. **A standalone machine without a local tape drive requires special installation procedures.** These are in the chapter ***Installing** UNIX **on Tapeless Workstations.***

Server

A server workstation on a local network provides resources like services and disk storage for other machines, which are called "clients". For installation purposes, the term "server" means "network disk server" (or "nd server" — see ***nd***(4p)), that is, a machine which provides disk storage for its clients. Normally, a server uses both the ***nd*** and the NFS ("Network File System" — see ***nfs***(4p)) protocols to exchange files with its clients. **For installation purposes, a server must have a local tape drive.**

Diskless Client

A client workstation on a local network relies on a server for disk storage. **To 'install' a diskless client workstation, complete installation on its server and then simply power it on.** The only things you need to know about the client for server installation are its name, hardware Ethernet address (see *Networking Terminology*, below), and Internet address (see below).

Diskful Client

A diskful client workstation on a local network relies on a server for resources, such as files, but has its own disk storage. A diskful client can actually boot the UNIX system up single user (whereas a diskless client cannot), but requires a remote machine to run multi-user. In other words, some of its files are local, and others are remote. The remote files can be obtained from any machine running NFS; the machine need not be an nd server.

At this writing, a straightforward installation path for such a machine does not exist; however, the NFS System Administration material, included with your documentation, provides notes. Basically, you can follow the path for a standalone machine until you get to running ***setup***; then you must edit files by hand, and so on. See the NFS material for procedures.

1.2.2. Networking Terminology

If you are installing several machines linked by a local network, part of installation includes basic network configuration.

Hardware configuration must be completed first: each machine must have Ethernet controller hardware, and be 'plugged in' via a transceiver to a common Ethernet cable. For Ethernet hardware configuration instructions see the ***Hardware Installation*** manual for your machine model.

Before doing software installation, you must obtain basic information about the system's place in the network. Some of the information items, like machine name, are arbitrary, and others are determined by hardware.

For additional information on networking and tuning your configuration, see the Sun *System Administration Manual.*

Before proceeding with the installation, obtain, decide on, and write down all the items in the following list:

Hostname
: (Also called "machine name"). Names the workstation. Although hostnames may contain up to 32 alpha/numeric characters, it is best to keep them brief. All alpha characters in a hostname **must be lower case**. Avoid the use of special characters. Assign each machine in your network a hostname.

Ethernet Address
: Refers to the address which is permanently assigned to each workstation, and is used by the Ethernet software to decide which packets to deliver to that machine. The Ethernet address resides in the ID PROM on the Sun-2 CPU Board. This address is a 6-byte hexadecimal value with each byte separated by a colon. A typical Ethernet address is "8:0:20:0:14:76".

  To find a machine's Ethernet address, power it on. You will see the Sun logo, and a message like:

```
Self Test completed successfully.

Sun Workstation, model_type, keyboard_type
ROM Rev N, some_number_MB memory installed
Serial #some_number, Ethernet address xx:xx:xx:xx:xx:xx

Auto-boot in progress . . .

abort using the appropriate abort sequence here

Abort at some address
>
```

Network Number
: The network number is an arbitrary value used to uniquely identify local networks across an interconnected network system. If you expect to be connected to the DARPA Internet, you must contact DARPA to get a network number which is unique on the Internet. If you do not intend to connect to a wider area network, you may use Sun's default network number: **192.9.200**.

Host Number
: The host number of each host uniquely identifies the host machine to the all other machines on the local net. The host number **must be between 1 and 255** (since 8 bits are allocated for host numbers on Class C networks), and it **must be unique**.

  You may elect to have host numbers assigned to all nodes 'automatically', or you may assign them 'manually'. If you elect to have then assigned automatically, the ***setup*** program, which you run later during the installation, handles it. When you run ***setup*** (see the chapter *Using*

Setup to Configure Your System), it asks if you want host numbers assigned automatically. If you answer yes, your server is given host number "1", and your clients are given numbers "2", "3", etc. This works well if you are installing a new network with no previously existing nodes attached to it. **However, if you are adding workstations to an existing network you should assign the host numbers yourself when using *setup*,** to ensure that each host number is unique.

Internet Address
A machine's Internet address consists of two parts: the network number followed by the host number. For example, "192.9.200.45" is a typical Internet address consisting of Sun's default network number, "192.9.200", followed by the host number "45".

Domain Name
The domain name identifies a group of workstations on a single local network that share the same */etc/passwd* and */etc/hosts* files. Note that you only ***need*** a domain name if you want to link two local networks, and hostnames on the two nets are not unique — two hosts with the same machine name are distinguished by their domain names. Usually, one organization will have one */etc/hosts* file for the entire organization; the domain name is then unnecessary. Note also that this domain name has absolutely nothing to do with the domain name used by the *sendmail*(8) program for mail routing on the ARPA Internet.

1.2.3. UNIX Device Naming

UNIX has its own set of names for devices. These names are fairly arbitrary, but are often based on abbreviations for the controllers used to drive the devices. When UNIX boots up, it probes the system for the device and controller configuration and reports what it finds there. Since you will be using these names during installation and in most of your administrative dealings with the system it's a good idea to identify your system's devices at this point, and to remember their UNIX device names:

Table 1-1: UNIX Device Names

Name	*Device*
xy	Xylogics 450/440 disk controller
ip	Interphase SMD-2180 disk controller
sd	SCSI disk controller
mt	Nine-track 1/2 in. magnetic tape
ar	Archive quarter-inch tape
st	SCSI 1/4 in. tape controller
xt	Xylogics 472 tape controller
ec	3COM Ethernet controller
ie	Sun-2 Ethernet controller

In the walkthrough of the installation, when we ask you to type in your ***tape***, ***disk***, or ***e_interface*** name, you should type in the UNIX device name for the appropriate device.

1.2.4. Disk Device Naming and Partitioning

Each physical disk attached to a Sun system is divided into eight ***logical*** disk partitions by the ***diag*** program, which is used during installation. These partitions are accessible as ***/dev/<disk_device_name disk_unit_number partition_name>***, where ***partition_name*** is a letter "a" to "h". For example, the first partition on my first disk attached to a Xylogics SMD Controller is called ***/dev/xy0a***. Each partition may be used for either a raw data area, such as a paging area, or to store a UNIX file system.

The number of these logical partitions actually used by a UNIX system depends on your system configuration. A typical standalone machine uses only three of these logical partitions: the first partition on a disk (the "a" partition) is usually used to store a root file system, from which UNIX may be bootstrapped, the second partition ("b") is used as a paging and swapping area, and the seventh partition, ("g") holds a user file system.

The "c" partition also has a conventional usage: it provides access to the entire physical device. It is occasionally used to store a single large file system or to access the entire pack when copying the disk's contents to another disk.

A disk partitioned by ***diag*** in this way 'looks' something like this:

Table 1-2: Standalone Disk Hard Partitions

Partition	*Contents*
a	Root Filesystem Area
b	Swapping Area
g	***/usr*** Filesystem Area

Servers, on the other hand, normally use a different partitioning scheme better suited to the network file system environment. Like the partitioning for standalone systems described above, the server's first or only disk uses the "a" and "b" partitions for root and swap areas, and uses the "c" partition to access the entire disk; unlike standalone systems, the "d" and "e" partitions on this disk are used to store UNIX utilities (the ***/usr*** directory) and user's home directories (***/usr2***) respectively. The "f" partition is also made available as an alternate "d" plus "e" partition; the "c" partition is sometimes used on an additional disk for a large ***/usr2*** partition. The "g" partition is commonly sub-partitioned for clients — this is discussed briefly below.

A disk partitioned for an NFS server may look something like this:

Table 1-3: Server Disk Hard Partitions

Partition	*Contents*
a	Server's Root Filesystem Area
b	Server's Swapping Area
d	***/usr*** Filesystem Area
e	***/usr2*** Filesystem Area
g	Clients' nd Partitions and ***/pub***

Although the division of the disk into eight hard partitions is adequate for standalone systems using the disk only for their own local storage, servers whose disk is being used for storage by clients need a finer breakdown of the "g" partition into subpartitions. This is handled in a separate phase of installation by the system configuration utility, ***setup***[1]. ***setup*** takes care of subdividing the "g" partition into "soft"[2] partitions: one for each client's paging area, one for each client's filesystem area, and one subpartition for a public, read-only file system (***/pub***) shared by all clients (see ***nd***(4p) for an explanation of how this is done). Contents of the ***/pub*** subpartition vary with system configuration.

After subpartitioning, a server's disk looks something like the following:

[1] Or may be handled 'manually' by editing the ***/etc/nd.local*** file. See the Sun *System Administration Manual* for instructions.

Table 1-4: Server Disk "g" Subpartitions

Partition	*Contents*
a	Server's Root Filesystem Area
b	Server's Swapping Area
d	*/usr* Filesystem Area
e	*/usr2* Filesystem Area
g	*/pub*
	Client1's Filesystem Area
	Client1's Swapping Area
	Client2's Filesystem Area
	Client2's Swapping Area

Details of using ***setup*** for subpartitioning are given in the chapter ***Using Setup to Configure Your System***.

When you are up and running UNIX, you can use the */etc/dkinfo*(8) command for a report on the physical details of each partition on your disk(s).

1.3. Conventions Used in the Procedures

In the installation walkthrough which follows, we use several conventions to make the directions clear, general enough for all configurations, and consistent. Some of these conventions are UNIX conventions, such as device names; others are simply Sun documentation conventions. Please familiarize yourself with them before continuing.

1.3.1. Use of Fonts in Procedures

We use fonts in this manual to distinguish what you type from what the system types at you, and to identify those items for which either you or the system must substitute a variable. Our conventions are:

- What you type is shown in **boldface text like this** or **`like this`**. Everything shown in boldface is a literal and should be typed exactly as it appears.

[2] In the sense that this partitioning information is not actually written on the disk label, as is the partitioning handled by *diag*.

- When parts of a command are shown in ***italic text like this***, they refer to a variable which you have to substitute from a selection; it is up to you to make the proper substitution.
- Items in `typewriter font like this` show what the system types at you.

Note that in the standard installation examples, the variables ***disk*** and ***tape*** are VERY common and important. You must replace ***disk*** with the UNIX device name for your disk, and ***tape***, with the device name for your tape drive (see the section *UNIX Device Naming*, above, for the correct names for your system's devices).

For example, a common configuration loads from a quarter-inch distribution tape using a drive attached to a SCSI controller, onto an SMD disk driven from a Xylogics 450 disk controller. In this case, replace ***tape*** with **st** (SCSI tape) and ***disk*** with **xy** (Xylogics disk) everywhere.

If you are performing a remote installation (that is, installing UNIX on a machine without a tape drive across the Ethernet from another machine) you will see the a third important variable: ***e_interface***. This is also described in the section *UNIX Device Naming* above. Replace ***e_interface*** with the correct device abbreviation for your Ethernet interface.

1.3.2. Naming Scheme for Reference Manual Pages

References to commands and utilities from the ***Commands Reference Manual for the Sun Workstation*** and the ***System Interface Manual for the Sun Workstation*** use the notation:

> *passwd*(1)

to indicate the ***passwd*** page in Section 1 of the manual pages. There are a total of eight sections: Sections 1, 6, 7, and 8 appear in the Commands Reference Manual; sections 2, 3, 4, and 5 appear in the ***System Interface Manual***. Thus, ***passwd***(5) means refer to the ***passwd*** manual page in the Section 5 pages of the ***System Interface Manual***. The notation *spline*(1G) means that this is a Graphics command in the Section 1 pages — you'll also see the addended letter to indicate other subsections, like *title*(3M) for a page in the Math Library subsection of the Section 3 pages, or *title*(3N) for a page in the Network Library subsection.

1.4. Correcting Typing Mistakes

To correct typing mistakes during the installation procedure, use the DEL key to erase and back up over incorrect characters or the control-U key to kill the entire line.

1.5. Abort Procedure

To return to the PROM monitor at any time during installation — either because you are asked to in the installation procedure or because you have botched things entierly — you can type what we call an **abort sequence** on your keyboard. The abort sequence usually consists of two keys typed in sequence; the first key is **held down** while the second key is typed. The keys vary with Sun Workstation model and keyboard type:

- If your Sun-1 keyboard has a 'SET-UP' key, the abort sequence is 'SET-UP a' (hold down the 'SET-UP' key while typing 'a').
- If your Sun-1 has an 'ERASE-EOF' key, the abort sequence is 'ERASE-EOF a' (hold down the 'ERASE-EOF' key while typing 'a').

- On a Sun-2 keyboard, type 'L1 a'. (hold down the 'L1' key in the uppermost left-hand corner while typing 'a').
- On a standard terminal (if it is the console) the 'BREAK' key generates an abort.

CAUTION: To avoid file system damage, never shut down the system when UNIX is running without preparing it first. Enter the command ***/etc/halt***, ***/etc/fasthalt***, or ***sync*** before aborting.

1.6. What is on the Distribution Tape?

The software needed to load the UNIX system is contained either on two half-inch magnetic tape reels, or on three quarter-inch tape cartridges, distributed with the system. The tapes contain six files which are used to install the basic UNIX system (three of these are installation utility programs, and three are file systems), seventeen files of additional software that you may load if you wish, and six copyright marks.

The additional software files are listed as "Optional" in the table which follows. These files are in *tar*(1) format, and may be copied to your disk(s) with the *extract_release* utility if you have sufficient storage capability. Procedures are given in Chapter 7 of this manual.

The following table describes the distribution tapes' contents, and the order[3] in which the tape files appear.

[3] Note that the boot command follows the convention of numbering the first file on the tape as file #0. If you must load these files directly from the tape, decrement the numbers in the table by 1.

Table 1-5: Contents of Distribution Tapes

1/2" Tape File Number	*1/4" Tape File Number*	*Contents*
Tape 1 1	**Tape 1** 1	A general purpose boot program which knows how to boot from the various devices that can be attached to the Sun Workstation. You boot this program from PROM monitor.
2	2	A copy of the ***diag*** program. ***diag*** is used during installation to format and label disks.
3	3	A stand alone ***copy*** program which can copy from specified sources to specified destinations.
4	4	An image of a miniature version of the root file system, which contains enough information to get you up and running.
5	5	Copyright file.
6	6	The complete root file system for the UNIX operating system. This is on the tape in ***tar***(1) format and is loaded in by the ***xtr*** or ***rxtr*** utility.
7	7	Optional File: Contains the online manual sources, in the directory ***/usr/man***.
8	8	Optional File: Games, the files in directory ***/usr/games***.
9	9	Optional File: Demonstration programs, the files in directory ***/usr/demo***.
10	10	Copyright file.
11	**Tape 2** 1	Copyright file.
12	2	A ***tar*** image of the ***/usr*** file system. The ***setup*** program transfers this file system to the disk.
13	3	Copyright file.
Tape 2 1	**Tape 3** 1	Copyright file.
2	2	Optional File: Various bootable diagnostic programs from directory ***/stand***.
3	3	Optional File: ***/lib*** FORTRAN compiler modules.
4	4	Optional File: FORTRAN libraries and commands.
5	5	Optional File: Diagnostic programs which may be run under UNIX.

1/2" Tape File Number	*1/4" Tape File Number*	*Contents*
6	6	Optional File: Graphics (SunCore and CGI) libraries.
7	7	Optional File: Network News (see, for example, *inews*(1)) software.
8	8	Optional File: Pascal interpreter, compiler, and libraries.
9	9	Optional File: Libraries compiled for profiling.
10	10	Optional File: Various Sun Window System tool sources, and demonstration program sources.
11	11	Optional File: Sun Window System executables and libraries.
12	12	Optional File: *uucp* programs (see *uucp*(1C)).
13	13	Optional File: Software for the Versatec Printer: *vtroff*(1) programs and */usr/lib/vfont* font files.
14	14	Copyright file.

1.7. Overview of the Installation Procedure

The steps required to load the UNIX system from magnetic tape onto your Sun Workstation are, briefly:

0. Read everything in each section before doing anything.
1. If you are setting up a local network of machines, or adding a new machine to such a network, determine network information for use later in installation. If you are installing a standalone workstation, obtain your complete Internet address (network number and host number). If you are installing a cluster of workstations, obtain each client's hardware Ethernet address as well.
2. Load the distribution tape on your tape drive.
3. Use the resident PROM monitor to boot the general purpose bootstrap program from the magnetic tape.
4. Use the bootstrap program to load the *diag* utility from the magnetic tape, and format and label your disk(s) with *diag*.
5. Use the bootstrap program to load the stand alone *copy* utility from the magnetic tape, and use *copy* to copy the mini UNIX root file system from the tape onto the swap area on the disk.
6. Boot the mini UNIX operating system just copied to your disk, and set the correct date.
7. Copy the full root file system from tape to disk using the *xtr* utility.
8. Boot UNIX in single-user state.

9. Run the ***setup*** program to acquire the ***/usr*** file system and to configure your entire system.
10. Boot UNIX again now that you have an entire system.
11. Use the ***/usr/etc/extract_release*** utility to load optional files from the distribution tape if you have sufficient storage for them. These optional files are listed above; they include the software for the Sun Window System, languages, diagnostics, special fonts for the Versatec printer, online manuals, demos, games and so on.
12. Reconfigure the system kernel. This step is mandatory for systems with only 1 MByte of memory, and highly recommended for systems with more.

You are now ready to proceed with the actual installation walkthrough in the following chapters. If you have not done an installation before, we strongly advise that you read it thoroughly before you begin. Good luck!

Chapter 2

Determining Network Information

PLEASE NOTE: The procedures in this chapter apply only to systems with Ethernet. If you do not have Ethernet, skip to the next chapter, ***Loading the Bootstrap Program.***

PLEASE NOTE: If you have an existing network (of older Sun-1 machines or mixed machines) that you do not intend to upgrade, you might need to make them compatible with the new network software. In any case, complete the steps given below for any machines you are installing or upgrading, and when you are finished, read the section ***Making 2.0 Networks Compatible With Existing Networks*** in the Sun *System Administration Manual.*

Before beginning actual installation, you must know:

1) The full Internet address (network number followed by unique host number) for each workstation you are setting up — whether it is a server or a client,

2) The hardware Ethernet address of each client machine if you are installing a server/clients configuration of machines, and

3) The domain name of the workstation or group of workstations you are installing.

These items will be requested later during installation, but must be obtained now.

For definitions of Internet address, domain name, and hardware Ethernet address, and for instructions on how to determine them, see the ***Networking Terminology*** section in the previous chapter.

Please remember that:

- You can use Sun Microsystems' default network number (192.9.200) if you have not been assigned a network number by ARPA, or if you are not connected to a higher level network.
- You only need a domain name if you have two or more distinct networks at your site, and these have duplicated hostnames or user id's (the domain name simply distinguishes network nodes). If you have only one ***/etc/passwd*** file and one ***/etc/hosts*** file for your organization (if there is a single unique user id and machine name space), the domain name is unnecessary.
- Host numbers of all machines must be between 1 and 255, and must be unique on your local network.
- You can allow ***setup*** to assign have host numbers automatically if you are installing a new network, but you should assign host numbers yourself if you are adding workstations to an existing network.

- Obtain each client's machine hardware Ethernet address by powering up the workstation, and checking the six-byte hexadecimal address displayed in the monitor power-up banner. When you turn the machine on, you'll see the Sun logo, and a message like the one in the example below. Abort immediately when the machine begins to auto-boot (If you don't know the proper abort sequence for your machine, see the section in the previous chapter titled *Abort Procedure*):

  ```
  Self Test completed successfully.

  Sun Workstation, model_type, keyboard_type
  ROM Rev N, some_number_MBytes memory installed
  Serial #some_number, Ethernet address xx:xx:xx:xx:xx:xx

  Auto-boot in progress . . .

  abort by typing the appropriate abort sequence here

  Abort at some address
  >
  ```

  You will need the entire six bytes of the displayed Ethernet address later; copy them down.

Chapter 3

Loading the Bootstrap Program

This chapter covers the first steps of actual installation: loading the distribution tape on your tape drive, and using the PROM Monitor to load the bootstrap program from tape. The bootstrap program is used to load other programs from tape into memory.

3.1. Step 1: Loading the Tape

If you have a nine-track half-inch tape, load it onto the tape drive following the diagram on the front of the drive and put the drive on-line.

On all machines except the Sun-2/160, hold the tape cartridge so that the word "SAFE" in it is at the top left hand corner, then push it firmly in the slot in the front of the tape drive. With the 2/160, hold the cartridge so that the word "SAFE" is towards you and to the left. Insert it end-first until it clicks into position, and to remove it, press it in until it clicks and releases.

If you have any questions about your tape drive, see the subsystems chapter in the ***Hardware Installation*** manual for your machine.

3.2. Step 2: Loading the Bootstrap Program

1. Turn on the Sun Workstation which you are installing. Almost immediately, the PROM monitor displays its power-up banner, which looks something like the example below, and then the machine begins to auto-boot. Stop the auto-boot immediately by typing the appropriate abort sequence for your machine (abort sequences are described in Chapter 1). When you abort the auto-boot, you return control to the monitor, and it displays its prompt (>):

```
Self Test completed successfully.

Sun Workstation, model_type, keyboard_type
ROM Rev N, some_number_MBytes memory installed
Serial #some_number, Ethernet address xx:xx:xx:xx:xx:xx

Auto-boot in progress . . .

abort by typing the abort sequence for your machine here

Abort at some_address
>
```

2. Now boot the general purpose bootstrap program from the tape by typing a **b** (for **boot**), followed by the two character device abbreviation for your tape drive type, followed by open and closed parentheses.

   In the following, please remember to substitute the proper device abbreviation for your tape controller for ***tape***: **ar** for Archive 1/4" Tape Controller, **st** for SCSI Tape Controller, **mt** for a 1/2" tape controlled by a Tapemaster Controller, or **xt** for a 1/2" tape controlled by a Xylogics 472 Tape Controller. For more information on device abbreviations or conventions used in these procedures, see Chapter 1.

   When you type the command, the monitor echoes it back to you, with the parameters filled in. A boot command looks like this:

```
> b tape()
Boot: tape(0,0,0)
Boot:
```

   As shown in the example, the bootstrap program displays a **Boot:** prompt at the start of a line when it is ready to accept commands. If you have a **1/4"** tape and the > prompt returns instead of **Boot**, the tape may need retensioning. Try entering:

```
>b tape (0,0,100)
Boot: tape(0,0,0)
Boot:
```

   Note that if for any reason control returns to the monitor instead of the bootstrap program, you will see the monitor prompt (>) instead of the bootstrap prompt (`Boot:`). If this happens, you must reactivate the bootstrap program, either from memory of from the tape.

   First, try to activate it from memory:

```
> g80000
Boot: tape(0,0,0)
Boot:
```

   If this fails, reload it from the tape, as shown above.

   Also note that if you type <**return**> to the `Boot` prompt, control returns to the monitor and you must either reload the bootstrap program or restart it from memory, as described above.

Now the bootstrap program is in control. In the following steps, it will be used to load the ***diag*** and ***copy*** programs. When those programs have completed, they return control to the bootstrap program. When the bootstrap program loads another program, it displays some numbers which are the sizes of the different portions of the program just loaded.

Chapter 4

Using *diag* to Format and Label Disks

This chapter describes the third step of installation: loading the ***diag*** program from the distribution tape and using it to format and label your disk(s).

Using ***diag*** has two phases: in the first phase, ***diag*** prompts for information about the type of disk drive and controller, then 'configures' itself to work with that disk and controller. In the second phase, ***diag*** allows you to format and label your disk by executing a series of protocols as you type the commands ***format*** and ***label.***

If you have multiple drives/controllers, complete both phases for your first drive on your first controller, and then loop back to begin the cycle for your next device. We give directions for this at the appropriate time.

4.1. Phase One: Specifying Hardware Configuration

1. First boot ***diag*** from your distribution tape. Type the following, (replacing ***tape*** with the appropriate device abbreviation for your controller):

```
Boot: tape(0,0,1)
Size: some_number+some_number+some_number bytes     [ varies with release ]
```

2. When ***diag*** starts up, it displays a sign on message, then it begins prompting for hardware information. It asks what sort of disk controller(s) and disk drive(s) you have, then it uses this information to configure itself to work with your hardware.

   First, ***diag*** asks for the disk controller type:

```
Version sccs version_number and date
Disk Initialization and Diagnosis

When asked if you are sure, respond with 'y' or 'Y'

specify controller:
        0 - Interphase SMD-2180
        1 - Xylogics 440 (prom set 926)
        2 - Xylogics 450
        3 - Adaptec ACB 4000 - SCSI
which one? type the number for your controller type
```

3. Next, ***diag*** asks what address this controller occupies on the main system bus. Unless you have an unusual controller configuration[4], use the correct address from this table:

[4] If you have one Xylogics SMD Controller and one Interphase SMD Controller in your system, you cannot use the

Table 4-1: Default Addresses for Disk Controllers

Controller Type	*Address (hex)* 1st Controller	2nd Controller
Interphase SMD-2180	40	48
Xylogics 450/440	ebee40	ebee48
Adaptec ACB 4000-SCSI for Sun-2/120 or 170	80000	84000
Adaptec ACB 4000-SCSI for Sun-2/160	ee2800	

When you give *diag* your controller's address, it echoes the address back to you:

```
Specify controller address on the mainbus (in hex): address
Device address: address you entered
```

4. If your controller interfaces to the SCSI bus (Adaptec, for example), ***diag*** asks for the address of the controller on the SCSI bus as well. The correct address is **0** for the first (or only) SCSI disk controller; **1** for the second:

```
Which target? SCSI bus address
```

5. Next, ***diag*** asks for the disk's unit number on this controller. The correct unit number is **0** for the first (or only) drive on this controller, and **1** for the second:

```
Which unit? unit number for the drive you're working with
```

6. Now ***diag*** asks for the type of disk drive you are working with. It displays a menu of the different disks it knows about, and asks you to specify your disk type. Note that ***diag*** contains a different menu for each controller. It displays the correct menu for the controller you specified earlier.

   If you do not know what type of disk you have, make a guess from the list ***diag*** displays. You will be able to verify and correct your guess in a few more steps; if you do guess, pay special attention to step 7.

   For instance, if you specified an Adeptec ACB 4000 SCSI, it displays the following:

default addresses (the Interphase Controller sees only four bytes). The Interphase must be configured to be at address 48, and the Xylogics at ee40. The Xylogics does not need to be the first controller specified. If both controllers are the same type, use the defaults. See the *Hardware Configuration and Expansion* chapter in the *System Manager's Manual for the 100U/150U* for Interphase Board configuration; board configuration procedures for other controllers are in the *Hardware Installation Manual* for the appropriate machine model.

```
Specify drive:
0 - Micropolis 1304
1 - Micropolis 1325
2 - Maxtor XT-1050
3 - Fujitsu-M2243AS
4 - Vertex V185
5 - Other

which one? type the number for your drive type

ncyl number acyl number nhead number nsect number interleave number
status:
[ ...status information... ]
```

After you select a disk drive from the menu, ***diag*** displays a table of physical data about that disk. This includes the number of cylinders, number of alternate cylinders, number of heads, and number of sectors per track. ***diag*** then displays drive-specific status information.

Now, ***diag*** knows everything it needs to know about the controller and the disk you are using. It displays its prompt:

```
diag>
```

7. If you are not sure you selected the correct disk type in step 6 above, use the ***verify*** command to read the label on the disk:

```
diag> verify
verify label
id: a disk type
    Partition a: starting cyl=0, # of blocks=#
    [ and so on ...]
```

If the drive type matches the disk drive type you specified in step 6, continue to phase 2. If not, type **diag** to re-start the initialization process, then go back to Step 2:

```
diag> diag
specify controller:
[ and so on ...]
```

You are now ready to begin the second phase of the ***diag*** operation: formatting and labeling the disk.

4.2. Phase Two: Formatting and Labeling the Disk

In this second phase, you use ***diag*** format a disk, then to label it.

For servers, use the ***partition*** subcommand to select and possibly modify an alternate label for the disk, and then use the **label** subcommand to write this label on the disk. For standalone systems, simply use the **label** subcommand to write a default label to the disk.

There are two distinct disk formatting and labeling paths: one for disks controlled by SCSI disk controllers (Adaptec, for example), and one for disks controlled by SMD controllers (Xylogics, for example). Follow the appropriate procedures for your system.

4.2.1. Formatting SCSI Disks

Note that all disks shipped by Sun are formated for testing at the factory. Because of the possibility that additional surface damage may occur during transit, it is our policy to have the customer reformat the disk on site.

1. Begin by typing **format** to call up *diag*'s SCSI formatting subprogram (*diag* remembers from its configuration phase). It displays its prompt:

```
diag> format
SCSI format.
format>
```

The SCSI format subprogram has various utilities which help you prepare and format a new disk. To see a list of its capabilities, type:

```
format> ?

SCSI Format Subcommands:
   f: format disk
   r: read defect list from disk
   p: print defect list from disk
   a: add defect to list
   d: delete defect from list
   s: surface analysis
   t: translate block no. to cyl/hd/bfi
   q: quit format
```

2. Check to make sure that the disk defect list written on the disk is the same as the hardcopy list shipped with your drive. Enter the command *r* to read the list from the disk into RAM, then enter the command *p* to display it on the screen:

Note that to avoid erasing valuable bad sector information, you must use the commands *r* and *p* correctly. The command *r* writes the bad sector list from the disk to RAM; the command *p* displays the bad sector list currently in RAM on the screen. To write the list from RAM to disk, you must format the disk using the command *f*. If you do a format without having done an *r*, you will overwrite list that was the disk. If you do an r without having done a format, you will overwrite any corrections made by either you or the surface analysis command (*s*).

```
format> r
format> p
Defect list
Defect  Cylinder  Head   Bytes from Index
  1        34      1        9213
etc.
```

3. Make sure this list matches the hardcopy disk defect list.
   a) If they match, continue with step 4.
   b) If the hardcopy list shows defects that are not displayed on the screen list, use the *a* command to add to the list on the disk:

```
format> a
cylinder? number
head? number
bytes from index? number
```

c) If you cannot read the list from the disk, use the *a* command as shown above to type in the entire hardcopy list.

d) After making any changes to the list on the disk, use the *p* command to display the changes on the screen. Check again to make sure the copy on the screen matches the hardcopy list.

Note that the location of the hardcopy list depends on the workstation type. It is usually taped to the front or top of the pedestal. If you don't see it or have misplaced it during unpacking, look for a second copy taped to the disk drive housing inside the pedestal (remove the two Phillips screws from the top rear of the enclosure, slide the top of the beige metal housing off, and check). **Be sure to replace this second copy when you have used it**. Also note that on a some drives, the hardcopy list groups defects by head; the cylinder and bytes from index numbers appear in the CYL and BI columns. On others, the numbers appear in the CYL, H, and BYTE columns respectively.

4. When you have verified the defect listing, format the disk with the format (**f**) command. After you type this command, the system displays a warning, then asks for confirmation:

```
format> format
format/verify, DESTROYS ALL DISK DATA!
are you sure? y
```

The formatting process takes three minutes or more. At the end, you should see a "Defect list written on disk" message. If you see any other message ("SCSI reset", for example), the formatting process did not succeed, and the defect list was not recorded on the disk. **You must format the disk again.**

5. When you have successfully formated the disk, do a surface analysis using the (**s**) sub-command. This analyzes the entire disk, then displays a list of any defective sectors it found. For a new disk, you should do five surface analysis passes:

```
format> s
# of surface analysis passes (5 is usual)? 5
```

Five passes may take an hour or more to complete. When it's done, it displays a message to announce that fact.

6. If it finds any bad sectors, it displays a report describing what it found, then adds the sector to the bad sector list in RAM. When its done, it tells you to re-format the disk:

```
Surface analysis complete
some_number bad sectors found
Use the 'f' command to format the disk.
format>
```

Before continuing, you **must**:

a. Reformat the disk (go back to step 4).

b. Continue this loop until the surface analysis reports that it found no bad sectors:

Once the surface analysis completes without finding any bad sectors, the format is successful. You may now continue to the next step, labeling the disk.

4.2.2. Labeling SCSI Disks

A disk must be labeled after it has been formated. The disk label records information about how the disk is divided into partitions for such things as paging space and file systems.

If you intend to run standalone, proceed to label your SCSI disk with the **label** command. This command provides a default partition table for SCSI disks; however, the table only works for standalone systems.

If you must use your SCSI disk as the only disk on a file server machine, then you must hand craft an alternate label for your disk; this must include "d" and "e" partitions. You can do this (with some difficulty) by using the **partition** command at this point. This is unusual and a bit complex; the **partition** command prompts for starting cylinder and number of blocks per partition, so you must know exactly how big you want to make each division. For an example of the **partition** command, see the "*Alternate Partitioning:* **partition**" section below. Otherwise, continue:

1. To label your disk, type the **label** command to the *diag* prompt. (If it still displays `format>`, type `q` to exit the format subsystem.) *diag* then asks if you want to use the logical partition map that is 'built in' to the program, and then asks for confirmation. Default partitioning for Sun-supplied SCSI disks is shown in the following table[5]:

Table 4-2: Default Partition Sizes for SCSI Disk Subsystems

SCSI Disk	*Raw*	*Partition Sizes (MBytes)*							
		"a"	"b"	"c"	"d"	"e"	"f"	"g"	"h"
Micropolis 1304	50	8.1	8.4	43.1	*unused*	*unused*	*unused*	26.5	*unused*
Micropolis 1325	85	8.1	17.1	70.9	*unused*	*unused*	*unused*	45.6	*unused*
Maxtor XT-1050	50	8.1	8.4	44.4	*unused*	*unused*	*unused*	27.9	*unused*
Fujitsu M2243AS	86	8.1	17.1	70.8	*unused*	*unused*	*unused*	45.6	*unused*
Vertex V185	85	8.1	17.1	70.9	*unused*	*unused*	*unused*	45.5	*unused*

If you confirm, *diag* will partition your disk according to these default maps:

```
diag> label
label this disk...
OK to use logical partition map 'your disk type'? y
Are you sure you want to write? y
```

2. After labeling the disk, *diag* automatically verifies the label it has just written. For example, the verify for a Fujitsu M2322 might look like this:

[5] Note that in all discussions of disk partitioning, numbers are approximate, since formated capacity depends on the type of controller being used with the drive. Also, note that a 'Megabyte', as far as numbers given in discussions of disk capacity, is defined as one million bytes.

```
[ This is an example only; do not enter this information. ]

verify label
id: <Fujitsu-M2322 cyl 821 alt 2 hd 10 sec 32 interleave 1>
        Partition a: starting cyl=0, # blocks=15884 [ #'s vary with disk ]
        Partition b: starting cyl=50, # blocks=33440
        Partition c: starting cyl=0, # blocks=262720
        Partition g: starting cyl=155, # blocks=213120
diag>
```

3. This completes the formatting and labeling process for a single SCSI disk. If you have a second disk drive on your controller, or two controllers and two or more drives, you may now return to the beginning of *diag*'s first phase by responding to the *diag* prompt with the command **diag**:

   ```
   diag> diag
   ```

   Note that you must complete both phases of *diag* for each of your disks. Be careful each time in the first phase of *diag* to respond with correct values when you are prompted for: controller mainbus address, controller type, disk type, and disk unit number.

4. If you have only one disk, or if you are done formatting and labeling all of your disks, you are ready to continue with the next phase of installation. Get back to the bootstrap program by typing the **q** (quit) command to *diag*, and you'll see the boot program's prompt again:

   ```
   diag> q

   Boot:
   ```

Now continue with the procedures in the next chapter.

4.2.3. Using fix to Format SMD Disks

You can use the command *fix* to format the SMD disk without re-doing existing mappings and slippings. This preserves the work done at the factory by Sun, and still repairs any new problems.

1. First, you must tell *diag* to *fix* the entire disk. Look at *diag*'s status display above, and add the number of data cylinders to the number of alternate cylinders. Then, enter the *fix* command, using the number obtained above in place of *nnn*:

   ```
   diag> fix
   fix -- DESTROYS SOME DISK DATA
   formats a range of tracks
   enter track number as 'cyl/track'
   starting track? 0/0
   ending track? nnn/0/0
   # of surface analysis passes (5 recommended)? 5
   OK to format from 0/0/0 to nnn-1/X/X ? y
   NNN
   diag>
   ```

   This process takes a while.

As *fix* formats each sector, it displays the cylinder and track number (represented by NNN in the above display). Before *fix* exits, it updates the bad sector map. Note that if it is interrupted, it does not update the map, and the process must be repeated.

Now, you may continue to the next step, labeling the disk.

4.2.4. Labeling SMD Disks

A disk must be labeled after it has been formated. The disk label records information about how the disk is divided into partitions for such things as paging space and file systems.

If you intend to run standalone, label your SMD disk with the **label** command. The **label** command uses a default partitioning table which works for standalone systems only. If you are installing an NFS server, use the **partition** command at this point to prepare an alternate label more suitable for running the Network File System, and then write this label to disk. Choose the path which suits your system.

4.2.4.1. Default Partitioning: **label**

Default partitioning is adequate for standalone systems. The default tables include the "a" (root), "b" (swap), "c" (entire disk), and "g" (for */usr*) partitions described in the first chapter (see *Disk Device Naming and Partitioning*). Default values[6] for these partitions (on standard SMD disk subsystems shipped by Sun) are:

Table 4-3: Default Partition Sizes for SMD Disk Subsystems

SMD Disk	*Raw*	*Partition Sizes (MBytes)*							
		"a"	"b"	"c"	"d"	"e"	"f"	"g"	"h"
Fujitsu 2312K (8")	84	8.1	17.1	67.3	*unused*	*unused*	*unused*	42.0	*unused*
Fujitsu 2284 (14")	169	8.1	17.1	134.5	*unused*	*unused*	*unused*	109.1	*unused*
Fujitsu 2322 (8")	168	8.1	17.1	134.5	*unused*	*unused*	*unused*	109.1	*unused*
Fujitsu 2351 Eagle	474	8.1	17.1	395.7	*unused*	*unused*	*unused*	369.8	*unused*

If this partitioning is suitable for your system, proceed as follows.

1. Type the **label** command in response to *diag*'s prompt. When you give the command to *diag*, it asks if you want to use the logical partition map that is 'built in' to the program (the default map), and then asks for confirmation before proceeding:

```
diag> label
label this disk...
OK to use logical partition map 'your disk type'? y
Are you sure you want to write? y
```

[6] Note that in all discussions of disk partitioning, numbers are approximate, since formated capacity depends on the type of controller being used with the drive. Also, note that a 'Megabyte', as far as numbers given in discussions of disk capacity, is defined as one million bytes.

2. After labeling the disk, *diag* automatically verifies the label it has just written. For example, the verify for a Fujitsu M2322 might look like this:

   [*This is an example only; do not enter this information.*]

```
verify label
id: <Fujitsu-M2322 cyl 821 alt 2 hd 10 sec 32 interleave 1>
        Partition a: starting cyl=0, # blocks=15884 [ #'s vary with disk ]
        Partition b: starting cyl=50, # blocks=33440
        Partition c: starting cyl=0, # blocks=262720
        Partition g: starting cyl=155, # blocks=213120
diag>
```

3. This completes the formatting and labeling process for a single disk. If you have a second disk drive on your controller, or two controllers and two or more drives, you may now return to the beginning of *diag*'s first phase by responding to the *diag* prompt with the command **diag**:

```
diag> diag
```

   Note that you must complete both phases of *diag* for each of your disks. Be careful every time you configure *diag* to respond with correct values for the controller mainbus address, controller type, disk type, and disk unit number.

4. If you have only one disk, or if you are done formatting and labeling all of your disks, you are ready to continue with the next phase of installation. Return to the bootstrap program by typing the **q** (quit) to *diag*. The bootstrap displays its prompt:

```
diag> q

Boot:
```

Now continue to the next chapter.

4.2.4.2. Alternate Partitioning: **partition**

If you intend to run the Network File System, use the **partition** command to set up a partitioning map better suited to the NFS environment.

Like the partitioning for standalone systems described above, the server's first or only disk uses the "a" and "b" partitions for root and swap areas, and uses the "c" partition to access the entire disk; unlike standalone systems, it uses the "d" and "e" partitions on this disk to store UNIX utilities (the */usr* directory) and user's home directories (*/usr2*) respectively. The "f" partition serves as a "d" plus "e" partition; the "f" partition is usually used on an additional disk for a large */usr2* partition. The "g" partition is commonly sub-partitioned for clients. Values for these partitions are:

Table 4-4: Alternate Partition Sizes for Systems Running NFS

NFS Disk	*Raw*	*Partition Sizes (MBytes)* "a"	"b"	"c"	"d"	"e"	"f"	"g"	"h"
Fujitsu 2284 SMD	169	8.1	17.1	134.5	30.0	25.6	55.5	53.6	***unused***
Fujitsu 2322 SMD	168	8.1	17.1	134.5	30.0	25.6	55.5	53.6	***unused***
Fujitsu 2351 SMD	474	8.1	17.1	395.7	30.1	130.0	160.2	209.6	***unused***

These alternate labels are designed for four users on a 2284/2322 or twelve users on an Eagle. If you would rather make your own label, you can modify one of these maps to suit. Use the following guidelines:

- Figure out the size of your "g" partition[7] by allowing approximately 6 MBytes for ***/pub***, plus 4-6 MBytes for each client's filesystem area, plus 10-12 MBytes of swap space for each client. Thus, if you have 7 users on an Eagle, each with 12 MBytes of swap space, then allow 6+7*(5+12)=125MBytes for "g".
- The "d" partition, for ***/usr***, should be at least 30 MBytes for the current basic distribution. If you want to load any of the optional tape files, increase the "d" partition by their amounts (given in Chapter 7).
- Put any remaining blocks into the "e" partition, where clients' home directories are normally mounted (***/usr2***).
- In this scheme, the "f" and "h" partitions are not used.

If you want to use the alternate maps, with or without modification, **and your server has only one disk**, jump over the ***digression for servers with multiple disks*** below and continue with the subsection, *Using* **partition** *for Alternate Partitioning*. If you have more than one disk on your server, please read the brief digression which follows.

4.2.4.3. Digression for Servers with Multiple Disks

If you have more than a single disk on your server(s), you may want to make an adjustment we have found works well with our configurations at Sun. Instead of having a small ***/usr*** and ***/usr2*** area on each disk, put a larger ***/usr*** on your first disk and a larger ***/usr2*** on your second. This uses disk space more efficiently on both disks.

If you choose to use this scheme, continue with the **partition** procedures as follows. When you use **partition**, request the "alternate" map for your first disk, then modify the "alternate" map for your first disk so that partition "g" uses all the remaining disk space following partition "d" (eliminate block"e" and give the space to "g"). We provide an example of this below.

Use an "alternate" map without modifications for your second disk; you will mount ***/usr2*** on the "c" partition of this disk.

[7] The "g" partition in this scheme contains one root filesystem area and one swapping partition for each client, plus a */pub* read-only filesystem area for (generally) */pub/vmunix*, */pub/boot*, */pub/stand*, and */pub/bin*.

When you subpartition the first disk (later, in Chapter 6) using ***setup***, you will be asked whether you want ***/usr2*** on this first disk. Answer "no". The first disk will then be loaded properly. Do not use ***setup*** at all on your second disk. Instead, run ***/etc/newfs*** on partition "c" of the second disk, then and place all clients' home directories in partition "c" of your second disk.

Finally, after you have run ***setup***, add the following line to ***/etc/fstab*** to tell the system that ***/usr2*** is mounted on the "c" partition of your second disk:

```
/dev/xy1c /usr2 4.2 rw 1 2
```

4.2.4.4. Using **partition** *for Alternate Partitioning*

If you want to use the alternate maps, with or without modification, proceed as follows:

1. Type the **partition** command in response to the ***diag*** prompt. The program then displays a menu of partition tables, and asks you to choose the one you wish to use:

```
diag> partition
Select partition table:
        0 - Fujitsu-M2312K
        1 - Fujitsu-M2284/M2322
        2 - Fujitsu-M2284/M2322 alternate
        3 - Fujitsu-M2351 Eagle
        4 - Fujitsu-M2351 Eagle alternate
        5 - Fujitsu-M2351 Eagle standalone
        6 - Other
Which one? 2 or 4
```

2. When you have selected a table, you are asked whether you want to modify it. If you have a single disk on this server, the answer is generally "n". In this case, you are asked to verify the partition sizes, confirm that you want to use the map you have chosen, and then asked to use the label command to write out the label for this map (you can follow the procedures in the ***label*** section just above for this).

   However, if you choose to make modifications to the alternate map, either to improve the performance of your configuration, or because you have more than one disk on a server, then answer "y" to the query. You are then asked to confirm or define the size of each disk partition. The following example, demonstrates how the dialogue goes on in this case. We assume the 'two-disks-per-server' scenario described above: the example shows how to effectively eliminate your "e" partition, and how to give the space to the "g" partition. We assume you have designated an Eagle alternate map:

```
[ CAUTION: This Is An Example Only . . . Do Not Enter This Information ]

Do you wish to modify this table? y
Partition a: starting cyl=0, # blocks=15884
Change this partition? n
Partition b: starting cyl=18, # blocks=33440
Change this partition? n
Partition c: starting cyl=0, # blocks=772800
Change this partition? n
Partition d: starting cyl=55, # blocks=58880
Change this partition? n
Partition e: starting cyl=119, # blocks=253920
Change this partition? n
Partition f: starting cyl=55, # blocks=312800
Change this partition? n
Partition g: starting cyl=395 # blocks=409400
Change this partition? y
Starting cylinder? 119   [ same as start of "e" partition ]
# of blocks? 663320   [ all the remaining blocks on the disk. Varies per disk ]
Partition h: starting cyl=0, # blocks=0
Change this partition? n
```

3. When you are done modifying the map, you are asked to verify the new table you have set up:

```
Verify partition table 'Fujitsu-M2351 Eagle alternate':
        Partition a: starting cyl=0, # blocks=15884
        [ and so on . . . ]
OK to use this partition table? y or n
```

4. When you have created a map you like and have verified it, you are reminded to "Use the label command to write out the partition table", then you see the *diag* prompt again. Use the **label** command as described in the preceding section (you will be given the alternate label you have just made).

5. Exit *diag* and then continue to the next chapter.

```
diag>q
Boot:
```

Chapter 5

Loading the Root File System

After a disk is formated and labeled, you can store information on it. In this chapter, you load the root file system to your disk. To do this, you must first load and boot a minimal subset of the UNIX system called mini UNIX. This contains the programs you will need to extract the real root file system properly.

5.1. Loading the Mini UNIX System

Boot the standalone *copy* program from tape, and use it to copy the mini UNIX system from the distribution tape to your disk.

Remember to substitute the correct device abbreviation for ***disk*** and ***tape*** (device abbreviations are given in Chapter 1). Also, note that the ***copy*** program prompts for the source (`From:`) and destination (`To:`) of the copy; there is very brief (approximately ten seconds) delay between the prompts as the ***copy*** program reads from the tape:

```
Boot: tape(0,0,2)
Boot: tape(0,0,2)
Size: 22528+4096+182836 bytes    [ numbers vary with system level ]
Standalone Copy
From: tape(0,0,3)
To: disk(0,0,1)
```

Copying in the mini UNIX system takes about four minutes using a half-inch tape, and about seven minutes using a quarter-inch cartridge. At the end of the copy, the ***copy*** program returns control to the bootstrap program:

```
Copy completed
Boot:
```

5.2. Booting the Mini UNIX System

1. Now that you have an operable mini UNIX system on disk, you can tell the the bootstrap program to boot mini UNIX from the disk. Because this boot is an unusual one, you must specify the **−a** (for ask me) option on the boot command, and also the **−s** (come up single user) option, as follows:

```
Boot: disk(0,0,1)vmunix -as
Size: 366592+61440+98828 bytes   [ numbers vary with system level ]
Sun UNIX 4.2 (GENERIC) #145: Tue Sept 11 20:35:13 PDT 1984
Copyright (c) 1984 by Sun Microsystems, Inc.

[ ...about a dozen lines of configuration messages... ]

root device?
```

2. As the mini UNIX system comes up, it displays some messages about the configuration of the system on which it is running, and finally queries you, asking for its root file system. The root file system at this stage is "*disk*0*", which has a special meaning to the mini UNIX system. Since this notation looks ambiguous, let me specify: if you have a Xylogics disk controller, your root device is **xy0*** ; if you have an Interphase controller, it is **ip0*** ; and if you have a SCSI disk controller, it is **sd0*** — the asterisk is part of the device name:

    ```
    root device? disk0*
    ```

Depending upon your hardware, you may be asked to set the date at this point:

```
using number buffers containing number bytes of main memory
WARNING: no tod clock -- CHECK AND RESET THE DATE!
#
```

If so, continue with the following section.

5.3. Setting the Date

Systems that have time of day clocks should set the date and time at this point, so that when the real root and */usr* file systems are loaded, all the log files and such will start with the right dates.

1. When the mini UNIX system starts up it displays a # prompt. You can now set the correct system date using the *date*(1) command:

    ```
    # date yymmddhhmm[.ss]
    ```

    where *yy* is the last two digits of the year; *mm* specifies month; *dd* designates day of the month; *hh* is hour (on a 24-hour clock); the next *mm* is minutes elapsed; and the optional *.ss* specifies seconds.

For example, to set the date at 4:43 PM and 30 seconds on July 30, 1984 you would type:

```
[ This is only an example; don't enter this information. ]
# date 8407301643.30
```

The system would then echo back to you:

```
Mon Jul 30 16:43:31 PDT 1984
#
```

Note: the "WARNING" message shown above is repeated when you boot the full UNIX system. Once you have set the date, you may ignore the message.

5.4. Loading the Root File System

Next, use the *xtr* shell script to extract the root file system from the distribution tape and write it to your disk at the proper location. This process takes a while, so be patient. During the first part of the extraction, *xtr* displays your disk super-block backup numbers. **These are crucial if you must repair a corrupted disk; copy them down.**

1. Extract the root file system from tape and place it on your disk by typing the following *xtr* command. Remember to substitute the appropriate device abbreviations for ***disk*** and ***tape***:

   ```
   # disk=disk tape=tape xtr
   ```

   Note that the *xtr* script assumes that the root file system goes on partition 'a' of the disk. If you wish to put the root file system on any other partition (very rare!), use the form:

   ```
   # disk=disk tape=tape root=X xtr
   ```

   where *X* is the partition where you want *xtr* place the root file system.

2. During the extraction, the *xtr* shell script displays a lot of messages which look something like the following. When your disk super-block back-up numbers are reported, copy them down:

   ```
   + cd /dev
   + ./MAKEDEV disk0 tape0
   [ ...Possible messages from MAKEDEV.
     You may ignore any Warning messages here... ]
   [ When the superblock backup numbers are displayed copy them down: ]
   super-block backups (for fsck  -b#) at:
           32,  3048, 6064, 9080, 12096, 15112 [ numbers vary with disk type ]
   + sync
   + /etc/fsck  /dev/rdisk0a
   [ ...Informative messages from fsck...
     If fsck halts or queries you for information DO NOT PROCEED;
     call Sun Microsystems Customer Support. ]
   + /etc/mount  /dev/disk0a /a
   + mt  -f  /dev/rtape0  rew
   + mt -f /dev/nrtape0 fsf 5
   + cd /a
   + tar xpfb /dev/rntape0 20
   [ ...Restoring the file system may take about ten minutes.
     You may ignore Warning messages during this time... ]
   [ ...More informative messages and another fsck.
     Same cautions apply as noted for fsck above.... ]
   Root filesystem extracted
   #
   ```

At this point, a complete UNIX root file system has been copied onto your disk.

5.5. Booting the UNIX System

Now you can boot UNIX:

1. Get back to the monitor by typing an abort sequence (see Chapter 1 for the abort sequence appropriate to your model). The monitor responds by displaying a message and a prompt when ready:

```
# abort using the appropriate abort sequence here
Abort at some address
>
```

2. When you see the monitor prompt, boot the UNIX system, using the **-s** option to make it come up single user. (There is no */usr* file system as yet.) The monitor responds by filling in the full details of the boot command. Then the system displays a progress report as it goes through the initialization sequence, and finally it displays the super-user prompt, #:

```
> b vmunix -s
Boot: disk(0,0,0)vmunix -s
Load: disk(0,0,0)boot
Boot: disk(0,0,0)vmunix
Size: 366592+61440+98828 bytes   [ numbers vary with system level ]
Sun UNIX 4.2 (GENERIC) #145: Tue Sept 11 20:35:13 PDT 1984
Copyright (c) 1984 by Sun Microsystems, Inc.

[ ...Several lines of configuration messages... ]

#
```

You are now up and running UNIX as the single, super-user. You have a full root file system, but no */usr* file system as yet. During the next phase of installation, you specify your system configuration, and acquire a */usr* file system on your disk.

Note that if you are converting a pre-NFS standalone workstation to use NFS (as per Section 10.2), do not continue with Chapter 6. Instead, continue with Step 4 in Section 10.2.

Chapter 6

Using Setup to Configure Your System

For the next installation task, you use the ***setup*** program to specify your system configuration and to acquire a ***/usr*** file system. This chapter describes both of these jobs. When you're done using ***setup***, you can boot up and run the full UNIX system.

setup works in two phases. The first phase is an interactive front-end which queries for the information it needs; the second phase is a non-interactive back-end which uses this information to do the actual configuration.

During the first phase, ***setup*** does consistency and error checking to ensure that the configuration will work. If it detects errors , it reports them to you, and asks you to enter corrected information.

The following three system types each require a different ***setup*** path:

- A standalone system,
- A system with equal-sized client disk partitions, or
- A server with different-sized client disk partitions.

6.1. Standalone Systems

If you are installing a standalone system, the dialogue with ***setup*** is fairly straightforward. As you glance through the walkthrough of the dialogue in the next section, you will see that you will be queried for your hostname (machine name), network and host numbers (if the machine is on an Ethernet), and if you want to assign a domain name to your system. If you are not networked at all, the answer is "no"; if you are networked with several systems sharing the same ***/etc/hosts*** and ***/etc/passwd*** files, and no domain name has yet been assigned, you may choose one.

Finally, ***setup*** asks you about your tape controller. Then, when it's completed its queries, ***setup*** begins its backend phase.

6.2. Servers

If you are installing a server, remember from the discussion in the first chapter of this manual (***Disk Device Naming and Partitioning***) that ***setup*** handles the "soft" sub-partitioning of the "g" partition of your disk: ***setup*** allocates a root filesystem area and a paging area for each client, and allocates an area for the read-only ***/pub*** directory shared by all clients.

There are two distinct ***setup*** paths available for this sub-partitioning. These are:

- Equal root filesystem and paging partitions for each of your clients, or
- Sub-partitions of varying sizes.

With the 'equal partitions' path, ***setup*** allows you to select the size of the public partition (which must be placed on the first disk you partition), and the size for a standard paging (swap) partition which will be given to each client. It divides the remaining disk space equally among clients for their root file system area.

With the 'unequal partitions' path, ***setup*** allows you to select the size of the public partition (which need not be on the first disk you partition), the size of each client's paging area, **and** the size of each client's filesystem area.

Choosing your path is mainly a matter of allocating your resources — for guidelines, read on.

6.2.1. Subpartitioning for NFS Servers

To choose your path, figure out whether equal partitioning allows you to match the number of clients you need to support with your available disk space.

At this point in installation, a disk set up for an NFS server looks something like this:

Table 6-1: NFS Server Disk Hard Partitions

Partition	*Contents*	*Size (MBytes)*
a	Server's Root Filesystem Area	8.1
b	Server's Swapping Area	17.1 or 8.4[9]
d		***varies with disk***
e		***varies with disk***
g		***varies with disk***

setup takes care of loading the ***/usr*** filesystem into your "d" partition; creating the ***/usr2*** filesystem on your "e" partition if you want it on your first disk[10], and subpartitioning your "g" partition for client's root filesystem areas, swap areas, and ***/pub*** (containing the clients' kernel image, ***/pub/vmunix***, a boot block, and the ***/bin*** directory). After running ***setup***, such a disk looks something like this:

[9] Disk-dependent: the Micropolis 1304 and Maxtor XT-1050 currently have 8.4 MByte swap areas (default); all other disks shipped by Sun have 17.1 MByte swap areas.

[10] If you have two disks on your server, and wish to use the "f" partition on your second disk for ***/usr2***, as suggested in Chapter 4, you can tell ***setup*** not to create ***/usr2*** on your first disk (see the instructions later in the walk-through), and place the filesystem 'manually' on your second disk (see the instructions in Chapter 4).

Table 6-2: NFS Server Disk Hard and Soft Partitions

	Contents	*Size (MBytes)* Hard	Minimum	Default	Suggested
a	Server's Root Filesystem Area	8.1			
b	Server's Swapping Area	17.1 or 8.4			
d	*/usr* Filesystem Area	30+ optional software			
e	*/usr2* Filesystem Area	*varies with disk*			
g	*/pub*		4.0	6.0	4-6
	Client1's Root Filesystem Area		4.0	6.0	4-6
	Client1's Swapping Area		4.0	6.0	8-12
	Client2's Root Filesystem		4.0	6.0	4-6
	Client2's Swapping Area		4.0	6.0	8-12

To determine whether you can use the 'equal' path, add your */pub* size to all your clients' swap area sizes, subtract this number from your total "g" partition size, and divide by the number of clients. If you used an "alternate" label for your disk, you can check the following table for your "g" partition size; use *dkinfo*(8) to remember your disk geography if you made your own label. In the following table, note that numbers are approximate, since formated capacity depends on the type of controller being used with the drive:

Table 6-3: Alternate Partition Sizes for NFS Servers

NFS Disk	*Raw*	*Partition Sizes (MBytes)* "a"	"b"	"c"	"d"	"e"	"f"	"g"	"h"
Fujitsu 2284 SMD	169	8.1	17.1	134.5	30.0	25.6	55.5	53.6	*unused*
Fujitsu 2322 SMD	168	8.1	17.1	134.5	30.0	25.6	55.5	53.6	*unused*
Fujitsu 2351 SMD	474	8.1	17.1	395.7	30.1	130.0	160.2	209.6	*unused*

If you come up with a reasonable root filesystem area after calculating, you can use the 'equal' path. In this case, proceed with the *Setting up Equal Client Partitions* subsection of the disk partitioning phase in the walkthrough.

If you wish instead to allocate varying amounts of disk storage space to clients, use the *Setting up Unequal Client Partitions* disk partitioning path; glance at it in the walkthrough below to get an idea of how it works. Since setting up custom partitioning is more complex than the other two paths, we urge you to work out your numbers before starting the walkthrough.

6.3. *setup* Walkthrough

This section presents a walkthrough of the ***setup*** dialogue for standalone machines and for servers. Please read through it before beginning your actual ***setup*** dialogue; it will help you use the program to your best advantage. Also, note that since the section treats three separate paths, it forks and merges at times; be careful to follow the path that applies to your system configuration.

Start the ***setup*** program by typing the ***setup*** command. ***Setup*** begins by requesting global information:

```
# setup

Sun Microsystems Configuration System

Global Information

     1) network disk server
     2) standalone
     3) standalone with remote tape

Enter the number for your environment: 1 or 2

You have a envi_type system; is this correct? (y/n): y
```

Note that ***setup*** asks you to verify configuration information after each 'phase' of configuration. It is extremely difficult to undo system configuration, so be careful with your responses: 'y' casts things in concrete, and 'n' allows you to start the phase over again. You can also type 'q' to any prompt to quit the ***setup*** program and return to the shell — this allows you to annihilate what you have done and start over.

After querying for global information, ***setup*** begins queries about disk configuration for servers only. If you are installing a standalone system with a disk, ***setup*** automatically sets up the proper disk configuration. Standalone systems can skip to the section ***Network Information***, below.

6.3.1. Disk Partitioning Information

This section and its subsections are relevant only to server machines.

This phase of the dialogue determines how many disks are being used in your system configuration, queries you about where you want ***/pub*** and ***/usr2***, and sets up partitioning for client root file system, client paging, and ***/pub*** areas.

Note: if you think you'll be adding clients in the future, it's an excellent idea to reserve partitions for them during the partitioning phase of the dialogue. You can allocate partitions of appropriate size to "user1" "user2" etc. Also, allocate dummy ethernet addresses to these dummy users later in the dialogue. The storage space you allocate for these future clients does not have to remain unused: you can make such partitions available as extra mounted filesystems for the initial set of clients. **If you do not reserve partitions in this way, you must re-partition your disk when you add new clients.**

For servers setting up equal client partitions, this phase of the dialogue allows you to choose which disk you want ***/usr2*** on, sets up a ***/pub*** partition of whatever size you select (if this is the first device being partitioned), allocates a standard paging area (referred to in the example below as the swap partition) for each of your specified clients (again, you determine the size), and carves the remaining free disk space into equally sized root file system area sub-partitions for each of them.

For servers setting up unequal client partitions, this phase allows you to select the disk placement of ***/usr2***, the size and disk placement of the public partition (it need not be on the first disk you partition, as it must in the standard path described just above), the size of each user's paging partition, and the size of each user's root file system area.

If you have more than one disk on a single controller, ***setup*** will completely finish partitioning the first disk and then turn to the second. If you have multiple controllers, you will be asked to repeat this entire phase for each controller. ***setup*** will validate you configuration after all disks have been partitioned. If it finds errors at that time — for example, if all clients have not been allocated both file system and paging areas — it will ask you to edit your specified configuration until it is correct.

6.3.1.1. Setting up Equal Client Partitions

Note that if you follow this path, you must put ***/pub*** on the first disk you configure.

setup begins by probing for your controller and determining how many disks will be configured into the system:

```
# setup
You have booted off of a controller type

Enter the number of disks attached to the controller type: (1-2): 1 or 2

The controller type has number disk(s):
      disk0 [ device name for your first or only disk ]
      disk1 [ device name for your second disk, if you have one ]

Is this configuration correct? (y/n): y
```

For the remainder of the dialogue, ***setup*** refers to your disk(s) by their UNIX device abbreviations as shown above.

Next, ***setup*** asks whether you want ***/usr2*** on this disk. There are two cases in which one might respond "n" to this question: first, if you are upgrading, and don't want ***setup*** to annihilate the ***/usr2*** already on your "e" partition; second, if you have more than one disk, and intend to put your ***/usr2*** on the "c" partition of your second disk (instructions for this are given in Chapter 4).

Then, ***setup*** handles subpartitioning. Don't forget to reserve partitions for future clients here, if you need to:

```
Disk Partition Information

Would you like setup to make a /usr2 partition on disk0? y or n

Would you like to partition device disk0 into "n" equal sized clients? (y/n): y

Enter the pub partition size (nnnM or nnnK)
              - minimum is 4M
              - default is 6M: size of /pub
     Partition size rounded to cylinder boundary K.

Enter the swap partition size (nnnM or nnnK)
              - minimum is 4096K
              - default is 6144K: size of clients' paging partitions
     Partition size rounded to cylinder boundary K.

Enter the number of clients on device disk0: (1-3): 1, 2, or 3

Enter a client's name: client 1

Enter a client's name: client 2

Enter a client's name: client 3

Do you want to add another controller? (y/n):
```

setup asks you to declare partition sizes in K Bytes or M Bytes, then rounds partitions to cylinder boundaries to improve performance. To get default partitioning, you can always type 'RETURN' in response to a sizing prompt.

This completes partitioning for the first disk. If you have a second disk attached to this controller (and you want to use ***setup*** on this disk), it queries again for disk partition information, and the same dialogue takes place — this time you supply information for the second disk. The same kind of 'looping' takes place if you have multiple controllers.

When all disks have been partitioned, ***setup*** asks for network information for each of your specified clients, and your network server. Skip to the ***Network Information*** section to continue.

6.3.1.2. Setting up Unequal Client Partitions

setup begins by probing for your controller and determining how many disks will be configured into the system:

```
You have booted off of a controller type

Enter the number of disks attached to the controller type: (1-2): 1 or 2

The controller type has number disk(s):
     disk0 [ device name for your first or only disk ]
     disk1 [ device name for your second disk, if you have one ]

Is this configuration correct? (y/n): y
```

For the remainder of the dialogue, ***setup*** refers to your disk(s) by their UNIX device abbreviations as shown above.

Next, ***setup*** asks whether you want ***/usr2*** on this disk. There are two cases in which one might respond "n" to this question: first, if you are upgrading, and don't want ***setup*** to annihilate the ***/usr2*** already on your "e" partition; secondly, if you have more than one disk, and intend to put your ***/usr2*** on the "c" partition of your second disk (instructions for this are given in Chapter 4).

Then, ***setup*** does partitioning. Note:

- that you need not put the ***/pub*** partition on the first disk you partition — you may put it on another disk **if you subpartition this disk using *setup***. If you do not intend to use ***setup*** on additional disk(s) (for example, if you intend to partition it 'manually' by editing */etc/nd.local* as suggested in the earlier chapter on ***diag***), you must put ***/pub*** on this one.
- that client file system and paging areas need not be on the same disk (**if you run both disks through *setup***), though each client must of course have both. Don't forget to reserve partitions at this point, if you need to, for future clients.

```
Disk Partition Information

Would you like setup to make a /usr2 partition on disk0? y or n

Would you like to partition device disk0 into "n" equal sized clients? (y/n): n

Partitions for disk disk0 - available capacity bytes free

Do you want the public partition on this disk unit? (y/n): y or n

[ If y, continue. If n see NO below.]

Enter the pub partition size (nnnM or nnnK)
               - minimum is 4096K
               - default is 6240K: /pub size
     Partition size rounded to cylinder boundary.

Partitions for disk disk0 - available — rounded /pub bytes free
1)      public:              rounded /pub size

[ If NO ]

Partitions for disk disk - available capacity bytes free

Do you want to add or edit a partition? (y/n/1): y

Enter the partition type (user/page/other): type

Enter the user partition size (nnnM or nnnK)
               - minimum is 4096K
               - default is 6144K: partition size
     Partition size rounded to cylinder boundary

[ If partition type is user or swap ]
Enter the name of the client: client name
```

Throughout this phase, ***setup*** declares the amount of free space on your disk and lists any allocated partitions before asking you to alter the partitioning in any way. It asks you to declare partition sizes in K Bytes or M Bytes, and it rounds partitions to cylinder boundaries to improve performance. To get default partitioning, you can always type 'RETURN' as a response to a sizing prompt.

Continue the 'add partition' process until you have finished partitioning your first disk. ***Setup*** then allocates any remaining disk space to "other", and asks you to verify your work. If something has gone wrong, you can edit a partition by responding with its number when you are prompted for adding or editing. Both size and type fields can be changed. Our final screen might look something like this:

```
Partitions for disk disk0 - 0K bytes free
1)      public:                         rounded /pub size
2)      client 1's    user:             rounded file system area
3)      client 1's    swap:             rounded paging
4)      client 2's    user:             rounded file system area
5)      client 2's    swap:             rounded paging
Is this partitioning correct? (y/n): y
Do you want to add another controller? (y/n):
```

This completes partitioning for the first disk. For each additional disk and controller, loop through the appropriate portion of the dialogue again.

When all disks have been partitioned, and the configuration has been validated, ***setup*** begins to gather network information. Continue with this below.

6.3.2. Network Information

Next, ***setup*** queries for network information. For a standalone without Ethernet, it only needs the hostname; for a standalone with Ethernet, it needs the Internet address (network number and host number); for a server, it needs clients' names, Internet addresses, and hardware Ethernet addresses. Finally, to identify different local groups of machines running NFS (each group shares its own ***/etc/hosts*** and ***/etc/passwd*** files), ***setup*** prompts for a domain name[11]. Only one machine in these groups (the server) needs to give such a name. Also, non-networked systems don't need the domain name. If you are confused about the networking terminology used here, please see the ***Network Terminology*** section in Chapter 1.

Remember that in this phase of the dialogue you may assign host numbers yourself, or have ***setup*** assign them automatically. If you are adding machines to an existing network, you must assign numbers yourself.

```
Network Information
Enter your hostname: hostname
Is this workstation on a network? (y/n): y or n
[ If y; else skip to "For standalones ...", below ]
Network numbers may be either class A, B or C
Their formats are:

    Class A:  nnn          (  0 <= nnn <= 127)
    Class B:  nnn.b1       (128 <= nnn <= 191)
    Class C:  nnn.b1.b2    (192 <= nnn <= 255)

b1 and b2 are one byte (0-255) quantities
Enter your network number (default is 192.9.200): network number
Your network number is number entered; is this correct? (y/n): y
[ For servers, the dialogue continues: ]
Enter the 6-byte hexadecimal ethernet address for each of the clients
The correct form is xx:xx:xx:xx:xx:xx
```

11 Remember that this domain name is completely different from the domain name used by the *sendmail*(8) program to route mail messages to different machines on the ARPA Internet. For more information on the domain name used by *sendmail*(8), and how to assign it, see the *System Administration Manual*, the section entitled *Setting Up The Mail Routing System.*

```
Client client 1: xx:xx:xx:xx:xx:xx

Client client 2: xx:xx:xx:xx:xx:xx

Client client 3: xx:xx:xx:xx:xx:xx

The clients and their addresses are:

     1) client 1:          xx:xx:xx:xx:xx:xx
     2) client 2:          xx:xx:xx:xx:xx:xx
     3) client 3:          xx:xx:xx:xx:xx:xx

Are the ethernet addresses correct? (y/n): y

Do you want host numbers automatically assigned? (y/n): y or n

[ If y ]
Server Information

Enter the name of the server: server name

The server's hostname is:

        server name

Is this correct? (y/n): y

[ For standalones, the dialogue continues: ]
Enter your host number

This is a number between 1 and 255: host number

Your hostname and host number are:

        hostname:          host number

Is this correct? (y/n): y

Do you want to pick a domain name? (y/n): y or n

[ If y ]
Enter the domain name: domain_name

[ If you chose to enter your own hostnumbers above ]
Each host number must be unique and between 1 and 255.

Enter the host number for each client

Client client_name NN

Client client_name NN

[and so on for all clients ...]

The clients and their host numbers are:

    1) client_name host number
    2) client_name host number

[and so on for all clients ...]

Are the host numbers correct? [y/n] y

[ loop back to "for standalones, the dialogue continues:"]
```

6.3.3. Tape Subsystem Information

The last phase of ***setup*** configuration requests information about your tape subsystem:

```
Tape Information

     1) 1/4'' SCSI tape (st)
     2) 1/2'' magnetic tape (mt)
     3) 1/4'' archive tape (ar)

Enter the number for the type of tape: (1-3): number

You have specified a tape type; is this correct? (y/n): y
```

When this last phase has been completed, ***setup*** asks you whether you want to institute the configuration you have designed and, if you confirm, proceeds to edit several of the database files:

```
You have completed the configuration questions.
Continuing will destroy any existing files under /usr
and any client partitions if you are a server.

Do you want to begin configuration? (y/n): y

Updating /etc/hosts

Updating /etc/nd.local

Making the public file system

   [ lots of messages ]

Extracting the /usr files

Making a file system for client_name

Initializing client_name's user partition

[ ...A few lines of configuration messages... ]
```

If your distribution is on 1/4-inch tape cartridges, ***setup*** prompts you to change to the second cartridge about two minutes into its back-end routine. Insert the second tape and type 'RETURN' to continue the routine; it takes approximately 25 minutes to complete.

When ***setup*** completes its back-end work, your shell prompt returns. **If you are installing a yellow pages master or a yellow pages slave server, continue.** If you are not installing a yellow pages master, continue your installation by booting the full UNIX system, as described in the next section.

If the machine you are installing is going to be a yellow pages master server or a yellow pages slave server, you must prepare it for that role now.

If it is to be a master yellow pages master server, perform the following:

Move to the */etc* directory and execute the following:

```
# cd /etc
# mount /usr
# /bin/csh
# /bin/hostname hostname
# /bin/domainname domainname
# ifconfig e_interface hostname
# cd /etc/yp
# ypinit -m
```

If the machine is to be a yellow pages slave server, perform the above steps with the following variations:

a) Ensure that its yellow pages master is up and running.

b) Edit the file */etc/hosts* and add the following line:

   internet_address ***master_name***

   where the internet address is that of the slave server, and where the ***internet_address*** and the ***master name*** are separated by a tab character.

c) Replace the line `ypinit -m` with the line:

   `ypinit -s` *master_name*

The program ***ypinit*** leads you through an interactive session. The following example shows the dialogue for a yellow pages master; the case for a yellow pages slave server is slightly different:

```
Installing ypdata will require that you answer a few questions.
Questions will all be asked at the beginning of the procedure.

Do you want this procedure to quit on non-fatal errors [y/n: n] n

At this point, we have to construct a list of the hosts which will run
yp servers.  hostname is the list of yp server hosts.  Please
continue to add the names for other hosts, one per line.  When you are
done with the list, type a <ctl D>.

   Next host to add: some host name
   [ and so on... ]
   Next host to add: <ctl d>

The current list of servers looks like this:


hostname
some host name
[ and so on... ]

Is this correct? [y/n: y] y

There will be no more questions.  The remainder of this procedure
should take 5 to 10 minutes.

Building...

[ list... takes a while ]

hostname has been setup as a yp master server without any errors.

If there are running slave yp servers, run yppush for any data bases
which have been changed.  If there are no running slaves, run ypinit on
those hosts which are to be slave servers.

#
```

Note that you need not worry about the action indicated in the last paragraph printed by *ypinit*; it occurs automatically when you reboot.

For additional information on yellow pages, see the "System Administration Manual".

6.4. Booting the Full UNIX System

When *setup* finishes its back-end routine, you have a complete */usr* file system on disk, and all the machine-specific files have been initialized, so that you can use the normal UNIX boot procedure to bring up the full UNIX system.

1. First, halt the system, using the */etc/halt* command. This shuts the system down in an orderly manner, and returns control to the monitor:

```
tutorial# /etc/halt
Syncing disks . . . .  done
UNIX halted

>
```

2. Now you can simply boot the UNIX system:

```
> b
Boot: disk(0,0,0)vmunix
Load: disk(0,0,0)boot
Boot: disk(0,0,0)vmunix
Size: 366592+61440+98828 bytes
Sun UNIX 4.2 (GENERIC) #145: Tue Feb 21 20:35:13 PDT 1984
Copyright (c) 1984 by Sun Microsystems, Inc.

[ . . . Many lines of configuration messages . . . ]

tutorial login: root
tutorial#
```

You can now use the full UNIX system on this machine.

If you are installing a server, proceed with the next final step in this chapter. If not, continue with the next chapter (if you wish to load any of the optional files/directories included on the distribution tape, to take advantage of Sun Microsystems' enhancements to 4.2BSD), or go on to configure your system kernel.

6.5. Copying an Unused Client Partition for Future Use

There are several ways to add new clients to an existing server's disk — some more difficult than others. The simplest way is to reserve 'other' partitions for these clients-to-be, as suggested in the ***setup*** walkthrough above; then adding the client is a matter of editing a few files (directions for this are given in your Sun ***System Administration*** Manual). Another way is to give clients 'vacated' partitions; that is, put them on partitions that have previously been used by other clients. **To do this, you must have a 'copy' of a brand new client partition available,** so it's a good idea to dump such a copy to tape either right now, if you're done using your drive, or after loading the final optional software files from the distribution tapes (next chapter). In any case, you must dump the partition **before booting up your current clients**, to be able to use it in the future.

To do this, proceed as follows:

1. Mount a good, blank tape on your half-inch tape drive, and put the drive on-line; or insert a new quarter-inch cartridge into your cartridge tape drive.
2. Choose an existing client partition, typically ***/dev/ndl0***, and dump it to tape (no need to mount it):
   - For 1/2" tape:

     ```
     anysys% dump 0uf /dev/rmt8 /dev/ndl0
     ```

   - For 1/4" tape:

```
anysys% dump 0cuf /dev/rtape8 /dev/nd10
```

where *tape* is replaced by the appropriate device abbreviation for your tape drive.

3. Label the tape clearly as a template copy of a brand new client partition and store it in a known, public place. Make sure the place is cool and clean.

When you want to add a new client, follow the directions given in the Sun *System Administration Manual*.

Now continue with the next chapter (if you wish to load any of the optional tape files included on the distribution tapes, to take advantage of Sun Microsystems' enhancements to 4.2BSD), or go on to configure your system kernel.

Chapter 7

Loading Optional Software

Several files on the distribution contain optional software. You may load any or all of these if you have sufficient storage resources. This chapter describes the contents of these optional tape files and gives instructions for loading them to your disk(s).

7.1. Contents and Sizes of Optional Distribution Tape Files

The software is stored in *tar* format in tape files #7 though #9 on the first reel/cartridge, and files #2 through #13 on the final reel/cartridge. The following table describes the contents the files and the ***approximate*** amount of disk space necessary to store them. When estimating space demands, calculate ± 5%.

Table 7-1: Contents and Sizes of Optional Distribution Tape Files

1/2" Tape File #	*1/4" Tape File #*	*Keyword*	*File Contents*	*Size*
Tape 1 7	**Tape 1** 7	**man**	Online reference manual pages, */usr/man*	2.45 MBytes
8	8	**games**	Games, */usr/games*	2.25 MBytes
9	9	**demo**	Demonstration programs, */usr/demo*	5.35 MBytes
Tape 2 2	**Tape 3** 2	**stand_diag**	Various bootable diagnostic programs	350 KBytes
3	3	**fortran**[13]	FORTRAN compiler modules from */lib*	100 KBytes
4	4		FORTRAN libraries and commands	475 KBytes
5	5	**usr_diag**	Diagnostic programs which may be run under UNIX	450 KBytes
6	6	**graphics**	SunCore and CGI libraries	1.48 MBytes
7	7	**news**	Network news software	875 KBytes

[13] Note that using the fortran keyword with *extract_release* loads both files 3 and 4. Assume 575 KBytes.

1/2" Tape File #	*1/4" Tape File #*	*Keyword*	*File Contents*	*Size*
8	8	**pascal**	Pascal interpreter and compiler	450 KBytes
9	9	**profiled**	Profiled libraries	625 KBytes
10	10	**src**	SunWindows tool source and demonstration program source	400 KBytes
11	11	**suntools**	SunWindows executables and libraries	1.75 MBytes
12	12	**uucp**	*uucp*(1) programs	550 KBytes
13	13	**vtroff**	Versatec Printer software: *vtroff* programs and font files, */usr/lib/vfont*	6.05 MBytes

7.2. The *extract_release* Utility

The *extract_release* utility extracts the desired distribution tape files from a local or remote tape drive, and places them on your disk(s). It prompts for the next tape when required, and uses the following command syntax:

extract_release *tape type keywords* . . .

where

tape specifies the tape controller (local or remote) being used for the extraction (**ar**, **xt**, **mt**, or **st**);

type is either

tapefull
if you are using a local tape, or

tapeless *servername*
if you are using a remote tape. Replace *servername* with the name of the machine with the tape drive.

keywords names one or more tape files you wish to extract. Keywords are given in the table just above: **man**, **games**, **demo**, **stand_diag**, **fortran**[14], **usr_diag**, **graphics**, **news**, **pascal**, **profiled**, **src**, **suntools**, **uucp**, or **vtroff**.

Thus, for example, to load the online manual pages to a SCSI disk from a SCSI tape drive, would load the first tape reel/cartridge and, as superuser, type:

```
mysys# extract_release st tapefull man
```

To load the SunWindows, graphics, and Pascal software to a disk via a remote tape drive attached to a machine named "gaia", load the third 1/4" tape cartridge and type:

```
mysys# extract_release tapeless gaia suntools graphics pascal
```

[14] Note that using the fortran keyword with *extract_release* loads both files 3 and 4. Assume 575 KBytes.

There is only one exception to this procedure, which is otherwise identical for installing on servers, diskfull, or diskless machines: to install either **news** or **uucp** on a diskless client, you must run the *extract_release* utility using either **news** or **uucp** as keywords on both the server and each client. For example:

```
server# extract_release tape tapefull news
[ and ]
client# extract_release tape tapeless server news
```

7.3. Using *extract_release* to Extract Optional Tape Files

You may run the *extract_release* utility from any directory; the utility will put files in their proper locations. To load optional software, proceed as follows:

1. Check your available disk space with *df*(1) to make sure you have adequate storage space for the software you wish to extract. For example:

```
anysys% df
Filesystem              kbytes    used    avail  capacity   Mounted on
/dev/xy0a                 7437    6254      439     93%     /
/dev/xy0g                10027    2846     6178     32%     /pub
/dev/xy0d                28351   22786     2729     89%     /usr
/dev/xy0e               122119   47130    62777     43%     /usr2
[and so on]
```

   Use the table above to calculate how much the space required to store each tape file on disk.

2. Load the cartridge/reel of the distribution tape which contains the files you wish to extract. You can use a local or remote tape drive for this. If *extract_release* requires you to change tapes while it is running it will prompt you for the new tape.

3. Become superuser on your machine and run the *extract_release* script:

```
mysys% su
Password:   type the superuser password here
mysys# /usr/etc/extract_release tape type keyword(s)
[  messages from extract_release . . .  ]
```

   See the above section if you have questions about the *extract_release* syntax.

This completes the portion of installation in which you need the distribution tape.

Chapter 8

Configuring the System Kernel

Sun Microsystems' implementation of UNIX provides for a ***configurable kernel.*** Certain system parameters that were hardwired in previous implementations can now be changed. We strongly advise that you configure the system kernel on all new UNIX systems to meet your particular needs. This reduces the kernel size, giving a larger effective memory size to programs and improving system performance substantially. This is especially important if you intend to run the Sun Window System.

PLEASE NOTE: all 1-MByte systems **must** and all other systems should perform this reconfiguration.

This chapter presents a very terse explanation and walkthrough of kernel configuration. W begin with a summary of the configuration process, explain the format of the configuration file used by the ***config*** utility (see ***config***(8)) to build your system configuration, and the present a walkthrough of the procedure.

This material should be adequate for users who have some experience with UNIX kernel configuration, and for those who have a standard system configuration. However, if there is anything you do not adequately understand, please read the sections on kernel configuration in the Sun *System Administration Manual.* Those sections go into more detail on the kernel configuration process, describe the layout of the kernel code, and explain the format of the configuration file in much greater detail.

Your configurable system also allows for adding new device drivers; all the kernel object files required to build a new system are present. For procedures, see the ***Device Driver Tutorial*** in the Sun ***System Internals Manual.***

8.1. Kernel Configuration Introduction

Building a new system is a semi-automatic process; most of the fine detail is handled by a configuration-build utility called ***/etc/config. /etc/config*** uses five files as input; these files are in the ***/usr/sys/conf*** directory as the system is shipped:

makefile.sun	generic makefile for Sun systems,
files	lists files required to build the basic kernel,

files.sun	files for a Sun-specific kernel,
devices.sun	name to major device mapping for Sun kernels, and
SYSTEM_NAME	describes characteristics of a specific system named *SYSTEM_NAME.* This is the only file you have to 'worry' about in this entire process; you must create it and tailor it to your system specifications.

When you run */etc/config*, it uses these files to generate the files needed to compile and link your kernel:

../SYSTEM_NAME/mbglue.s	contains short assembly language routines used for vectored interrupts,
../SYSTEM_NAME/ioconf.c	contains a description of I/O devices attached to the system,
../SYSTEM_NAME/makefile	for building the system, and
../SYSTEM_NAME/device_name.h	a set of header files (various *device_name*'s) containing devices which can be compiled into the system.

Note that */etc/config* places all its output files in a directory called *../SYSTEM_NAME*, which it assumes exists; you must create this directory before running */etc/config*.

Next, you use the generated makefile to create the dependency graph for the new system, build the system, and, finally, install the new kernel.

If you are installing a server, you should also make and install a kernel for your clients.

8.2. Producing a System Configuration File

Your contribution to this process is to create a file SYSTEM_NAME which contains a description of the kernal you want */etc/config* to produce. It uses this information to create a kernal named SYSTEM_NAME. We have tried to make this phase as painless as possible in a few ways.

First, rather than creating the file from scratch, you can copy and edit one of the several template files provided with the distribution. These templates are located in the */usr/sys/conf* directory, and their names are, by convention, written in upper case:

GENERIC	Contains a line for every Sun-supported device. Your system is shipped with a GENERIC kernel.
ND100	Diskless Sun-2/100U
ND120	Diskless Sun-2/120
ND50	Sun-2/50
SDST120	Sun-2/120 with one SCSI disk and tape
SDST160	Sun-2/160 with one or two SCSI disks and one SCSI tape
XY100	Sun-2/100U with one Xylogics disk
XYAR100	Sun-2/100U with one Xylogics disk and Archive tape
XYMT150	Sun-2/150U fileserver with two Xylogics disks and one 1/2" tape
XYMT160	Sun-2/160 fileserver with SMD and SCSI disks, and 1/2" tape

Copies of these files are included as Appendix A of this manual. Rather than starting from *GENERIC*, we suggest you choose one of the tailored files which most closely approximates your system configuration for editing.

To understand the format of the configuration file, begin by taking a look at the 'mother' template, *GENERIC*. The beginning of *GENERIC* looks something like this:

```
#
# GENERIC SUN
#
machine         sun
cpu             "SUN2"
ident           GENERIC
timezone        8 dst
maxusers        4
options         INET
options         SYSACCT
options         RPC
options         NFS

config          vmunix          swap generic

pseudo-device   nfs
pseudo-device   rpc
[ and so on ]
```

You can see by the line groupings that the configuration file has three different 'types' of entries:

- Lines which describe general things about the system (parameters global to the kernel image which this configuration file generates),
- A line which describes things specific to each kernel image generated, and
- Lines which describe the devices on the system, and what those devices are connected to.

The next three subsections cover these three types of lines.

8.2.1. General System Description Lines

The first six general description lines in the configuration file are mandatory for every Sun system. They are:

machine *type*
This system is to run on the machine type specified. Only **one** machine type can appear in the configuration file. The legal ***type*** for a Sun system is **sun**.

cpu *type*
This system is to run on the cpu type specified. For a Sun system, legal ***type*** is **"SUN2"** (enclose in double quotes).

ident *name*
Gives the system identifier — a name for the machine or machines that run this kernel. ***name*** must be enclosed in double quotes if it contains both letters and numbers (for example, "SDST120"), or you will get a syntax error when you run */etc/config*. Also, note that if ***name*** is **GENERIC**, you can specify the unique ***config_clause*** **swap generic** in the **config** line described below (see the next section, ***Specific System Description Lines***). If you use any other string for ***name***, and you also include an **options GENERIC** line, you can still use the **swap generic** line. However, if you use any other string for ***name*** and omit the **options GENERIC** line, you may **NOT** use the line **config vmunix swap generic** to specify your kernel image.

timezone *number* [**dst**]
Specifies the timezone you are in, measured in the number of hours west of GMT. 5 is EST, 8 is PST. Negative numbers indicate hours east of GMT. If you specify **dst**, the system will convert to and from daylight savings time when appropriate. An optional integer or floating point number may be used to specify a particular daylight savings time correction algorithm; the default value is 1, indicating the United States. Other values are: 2 (Australian style), 3 (Western European), 4 (Middle European), and 5 (Eastern European). See *gettimeofday*(2) and *ctime*(3) for more information.

maxusers *number*
The maximum expected number of simultaneously active users on this system is ***number***. This number is used to size several system data structures. In particular, it controls the size of the process table which sets the upper bound on the number of user processes. Since this table is statically allocated, it cuts into your buffer space; it is thus important to set ***number*** no higher than necessary for your system. If you are configuring a kernel for a single-user Sun Workstation, for a single-user networking node, or — especially — a 1 MByte system, set ***number*** to "2". Use "4" only if you anticipate running an unusually high number of user processes per machine — for example, if you plan to run 7+ windows at a time.

options *optlist*
Compile the listed options into the system. Options in this list are separated by commas. There is a list of options that you may specify in the generic makefile. A line of the form 'options FUNNY,HAHA' yields −DFUNNY −DHAHA to the C compiler. An option may be given a value, by following its name with '=' then the value enclosed in (double) quotes. None of the standard options use such a value. Note that the **options INET** line must be included for all Sun systems; other options are up to you.

8.2.2. Specific System Description Lines

The next 'type' of line in the system configuration file is a single line specifying the name and location of a bootable kernel image. Multiple bootable images may be specified using multiple lines of this type. The line has the syntax:

> **config** ***kernelname config_clauses***

where

kernelname
: Is the name of the loaded kernel image. ***kernelname*** is usually **vmunix**.

config_clauses
: are one or more specifications indicating where the root file system is located, how many paging (or swap) devices there are and where they go. A ***config_clause*** may be one or more of the following:

  root [on] ***root_device***
  : Specifies location of the root file system.

  swap [on] ***swap_device*** [and ***swap-device***]
  : Specifies location of swapping and paging areas.

  dumps [on] ***dump_device***
  : Specifies device where you would like crash dumps to be taken.

  args [on] ***arg_device***
  : Specifies device where argument list processing for the ***execve***(2) system call should be done.

  The "on" in the syntax of each clause is optional, and the "and" clause in the **swap** clause may be repeated zero or more times. Multiple ***config_clauses*** are separated by white space. For example, the ***config*** line for a system with root on its first SMD disk (partition 'a'), and paging on the 'b' partition of this disk might be:

```
config vmunix root on xy0 swap on xy0b
```

  Note also that the device names supplied in the clauses may be fully specified — as a device, unit, and file system partition — or underspecified. If underspecified, the ***config*** program uses built-in rules to select default unit numbers and file system partitions. For example, the swap area need not be specified at all if the root device is specified, because the default is to place swap in the "b" partition of the same disk where the root file system is located. Thus, our example line could have been simply:

```
config vmunix root xy0
```

A final note: please remember that you cannot use the line **config vmunix swap generic** unless you have named your system image "GENERIC" or have included an "options GENERIC" line. See the notes for the "ident" system description line in the preceding section.

8.2.3. Device Description Lines

Each device attached to a machine must be specified so that the system generated will know to probe for it during the autoconfiguration process carried out at boot time. The final type of entry in the configuration file tells the system what devices to look for and use, and how these devices are connected together.

Note that the device description line for each supported device is given in the "synopsis" portion of the Section 4 reference manual page for that device. These pages are in the Sun ***System Interface Manual.***

Each device description line has the following format:

> ***dev_type dev_name*** **at** ***connect_dev more_info***

where

dev_type
Specifies the device type. ***dev_type*** may be one of the following:

controller
In general, a disk or tape controller. For all Sun systems, the line:

```
controller     mb0 at nexus ?
```

must be included in every configuration file, in addition to any the line for each tape and disk controller you have. This line configures in the main system bus.

disk or **tape**
Devices connected to a controller.

device
Something 'attached' to the main system bus, like an Ethernet controller.

pseudo-device
A software subsystem or driver treated like a device driver, but without any associated hardware. Current examples are the pseudo-tty driver and various network subsystems.

dev_name
The standard UNIX device name and unit number (if the device is not a **pseudo-device**) of the device you are specifying. For example, the lines for my Xylogics SMD controller and the two drives on it would be:

```
controller     xyc0 at mb0 csr all virt 0xebee40 priority 2 vector xyintr 72
disk           xy0 at xyc0 drive 0
disk           xy1 at xyc0 drive 1
```

connect_dev
is what the thing you are specifying is connected to. For instance, in the example above, disk xy1 is connected to controller xyc0.

more_info
is a sequence of the following:

csr [***bus_spec space_spec***] ***addr***
Specifies the address of the csr (command and status registers) for a device. The optional bus specification (***bus_spec***) and space specification (***space_spec***) pair may be included in the address; otherwise, ***config*** will use a heuristic using ***addr*** to determine the bus and space. Typically, ***addr*** is given as a hexadecimal value.

bus_spec may be:

all	The device is on any main system bus.
mb	The device is on the Multibus.
vme	The device is on the VMEbus.

space_spec may be:

virt	Virtual address (preset by monitor) follows.
obmem	The following address is in on-board memory.
obio	The following address is in on-board I/O.
busmem	The following address is in main bus memory.
busio	The following address is in main bus I/O.

The csr address must be specified for all controllers, and for all devices connected to the main system bus (whether it is a Multibus or a VMEbus).

drive *number*
For a disk or tape, specifies which drive this is.

flags *number*
These flags are passed to the device driver at system initialization time.

priority *level*
For devices which interrupt, specifies the interrupt level.

vector *intr number* [*intr number* . . .]
For devices which use vectored interrupts on VMEbus systems, *intr* specifies the vectored interrupt routine and *number* the corresponding vector to be used (64-255).

A **?** may be substituted for a number in two places and the system will figure out what to fill in for the **?** when it boots. You can put question marks on a *con_dev* (for example, at xyc?), or on a drive number (for example, drive ?). This allows redundancy, as a single system can be built which will boot on different hardware configurations.

8.2.4. An Annotated GENERIC File

The next few pages, contain an annotated copy of the *GENERIC* file to help you identify the lines you need to include in your own system configuration file. The comments explain the device and pseudo-device lines, and may also refer you to the reference manual entry which covers the device in question. If the comments say the line is **mandatory**, then the line **must** be included in every system configuration file, either exactly as it stands, or, if commentary indicates variables, with the variables adjusted to fit your system.

```
#
# GENERIC SUN
#
machine    sun
```
[mandatory.]

```
cpu        "SUN2"
```
[mandatory.]

```
ident      GENERIC
```
[mandatory. If you use "GENERIC" as your system identifier, you may use the "swap generic" clause in the "config" line below. If you customize the identifier to *SYS_NAME*, you must either include an "options GENERIC" line, or specify at least the device where your root file system lives in place of "swap generic". For example, the "config" line for a standard Sun-2 might read: "config vmunix root on xy". See *General* and *Specific System Description Lines*, above, for information. Finally, if *SYS_NAME* contains both alpha and numeric characters (as in, for example, SDST120), you must enclose the name in double quotes ("SDST120") or you will get a syntax error when you run */etc/config*.]

```
timezone   8 dst
```
[mandatory. Specifies your timezone. Adjust value accordingly.]

```
maxusers   4
```
[mandatory. Number may vary. For most systems, "2" is the proper value for maxusers. See the section *General System Description Lines*, above, for information.]

```
options    INET
```
[mandatory. Controls inclusion of Internet code -- see *inet*(4). You must also include the "pseudo-device inet" and "pseudo-device loop" lines below.]

```
options    SYSACCT
```
[Controls inclusion of code to do process accounting — see *acct*(2) and *acct*(5). If you include this line, you must also include the "pseudo-device sysacct" line below.]

```
options    RPC
```
[Controls inclusion of Network File System code. If you include this line, you must also include the "options NFS" line, and pseudo-devices "nfs" and "rpc".]

```
options    NFS
```
[See "options RPC" line above.]

```
config     vmunix  swap generic
```
[mandatory. Specify kernelname and configuration clauses. Please see *Specific System Description Lines*, above, for information.]

```
pseudo-device    nfs
```
[See "options RPC" line above.]

```
pseudo-device    rpc
```
[See "options RPC" line above.]

```
pseudo-device    pty
```
[Pseudo-tty's. Needed for network or window system.]

```
pseudo-device    bk
```
[Berknet line discipline for high speed tty input — see *bk*(4).]

```
pseudo-device    sysacct
```
[See "options SYSACCT" line above.]

```
pseudo-device    inet
```
[mandatory. See "options INET" line above.]

```
pseudo-device    ether
```
[ARP code. Must include if using Ethernet — see *arp*(4).]

```
pseudo-device     loop
```
[mandatory. Software loop back network device driver — see *lo*(4). Must include with 'options INET'.]

```
pseudo-device     nd
```
[Network disk. Necessary for servers and diskless clients, and for machines serving as remote hosts for remote installation — see *nd*(4).]

```
pseudo-device     win128
```
[Window system. Number indicates maximum windows. If you include this line, you must also include the "pseudo-device dtop", "ms", and "kb" lines just below.]

```
pseudo-device     dtop4
```
[Maximum number of screens ('desktops'). Required for window system.]

```
pseudo-device     ms3
```
[Maximum number of mice. Required for window system — see *ms*(4).]

```
pseudo-device     kb3
```
[Maximum number of Sun keyboards. Required if using any Sun keyboard, and for the window system.]

```
pseudo-device     ingres
```
[Sun MicroINGRES lock device.]

```
controller        mb0 at nexus ?
```
[mandatory. Main bus code.]

```
controller        ipc0 at mb0 csr all virt 0xeb0040 priority 2
```
[1st Interphase SMD disk controller — see *ip*(4).]

```
controller        ipc1 at mb0 csr all virt 0xeb0044 priority 2
```
[2nd Interphase controller.]

```
disk        ip0 at ipc0 drive 0
```
[1st disk on 1st Interphase controller.]

```
disk        ip1 at ipc0 drive 1
```
[2nd disk on 1st Interphase controller.]

```
disk        ip2 at ipc1 drive 0
```
[1st disk on 2nd Interphase controller.]

```
disk        ip3 at ipc1 drive 1
```
[2nd disk on 2nd Interphase controller.]

```
controller        xyc0 at mb0 csr all virt 0xebee40 priority 2 vector xyintr 72
```
[1st Xylogics SMD disk controller — see *xy*(4).]

```
controller        xyc1 at mb0 csr all virt 0xebee48 priority 2 vector xyintr 73
```
[2nd Xylogics controller.]

```
disk        xy0 at xyc0 drive 0
```
[1st disk on 1st Xylogics controller.]

```
disk        xy1 at xyc0 drive 1
```
[2nd disk on 1st Xylogics controller.]

```
disk        xy2 at xyc1 drive 0
```
[1st disk on 2nd Xylogics controller.]

```
disk        xy3 at xyc1 drive 1
```
[2nd disk on 2nd Xylogics controller.]

```
controller        sc0 at mb0 csr 0x80000 priority 2
```
[1st SCSI controller on a Sun-2/120 or Sun-2/170.]

```
controller        sc0 at mb0 csr vme busmem 0x200000 priority 2 vector scintr 64
```
[1st SCSI controller on a Sun-2/160.]

```
disk        sd0 at sc0 drive 0 flags 0
```
[1st disk on 1st SCSI controller.]

```
disk        sd1 at sc0 drive 1 flags 0
```
[2nd disk on 1st SCSI controller.]

```
tape        st0 at sc0 drive 32 flags 1
```
[SCSI tape.]

```
controller        sc1 at mb0 csr 0x84000 priority 2
```
[2nd SCSI controller.]

```
disk        sd2 at sc1 drive 0 flags 0
```
[1st disk on 2nd SCSI controller.]

```
disk        sd3 at sc1 drive 1 flags 0
```
[2nd disk on 2nd SCSI controller.]

```
tape        st1 at sc1 drive 32 flags 1
```
[SCSI tape.]

```
device      ropc0 at mb0 csr 0xee0800
```
[mandatory. Raster Op chip — see *ropc*(4).]

```
device      sky0 at mb0 csr 0x2000 priority 2
```
[Sky Floating Point board in any Sun-1, Sun-2/120, or Sun-2/170.]

```
device      sky0 at mb0 csr vme busio 0x8000 priority 2 vector skyintr 176
```
[Sky Floating Point board in a Sun-2/50 or Sun-2/160.]

```
device      zs0 at mb0 csr all virt 0xeec800 flags 3 priority 3
```
[CPU serial I/O ports — see *zs*(4).]

```
device      zs1 at mb0 csr all virt 0xeec000 flags 0x103 priority 3
```
[Sun-2 Video Board ports. Required for Sun-2 keyboard and mouse.]

```
device      zs2 at mb0 csr 0x80800 flags 3 priority 3
```
[1st two serial I/O ports on 1st SCSI Board.]

```
device      zs3 at mb0 csr 0x81000 flags 3 priority 3
```
[2nd two serial I/O ports on 1st SCSI Board.]

```
device      zs4 at mb0 csr 0x84800 flags 3 priority 3
```
[1st two serial I/O ports on 2nd SCSI Board.]

```
device      zs5 at mb0 csr 0x85000 flags 3 priority 3
```
[2nd two serial I/O ports on 2nd SCSI Board.]

```
device      mti0 at mb0 csr all virt 0xeb0620 flags 0xffff priority 4 vector mtiintr 136
```
[Systech terminal MUX — see *mti*(4).]

```
device      ie0 at mb0 csr 0x88000 priority 3
```
[1st Sun-2 Ethernet Controller on a Sun-2/120 or Sun-2/170.]

```
device      ie0 at mb0 csr vme virt 0x0ee3000 priority 3
```
[1st Sun-2 Ethernet Controller on a Sun-2/50 or Sun-2/160.]

```
device      ie0 at mb0 csr 0x8c000 flags 2 priority 3
```
[2nd Sun-2 Ethernet Controller on a Sun-2/120 or Sun-2/170.]

```
device      ec0 at mb0 csr 0xe0000 priority 3
```
[1st 3COM Ethernet Controller — see *ec*(4).]

```
device      ec1 at mb0 csr 0xe2000 priority 3
```
[2nd 3COM Ethernet Controller — see *ec*(4).]

```
controller        tm0 at mb0 csr all virt 0xeb00a0 priority 3 vector tmintr 96
```
[1st TAPEMASTER tape controller — see *tm*(4).]

```
controller        tm1 at mb0 csr all virt 0xeb00a2 priority 3 vector tmintr 97
```
[2nd TAPEMASTER tape controller — see *tm*(4).]

```
tape        mt0 at tm0 drive 0 flags 1
```

```
        [ 1st 1/2" tape drive on 1st TAPEMASTER controller. ]
tape        mt1 at tm1 drive 0 flags 1
        [ 1st 1/2" tape drive on 2nd TAPEMASTER controller. ]
controller          xtc0 at mb0 csr all virt 0xebee60 priority 3 vector xtintr 100
        [ 1st Xylogics 472 Tape Controller. ]
controller          xtc1 at mb0 csr all virt 0xebee68 priority 3 vector xtintr 101
        [ 2nd Xylogics 472 Tape Controller. ]
tape        xt0 at xtc0 drive 0 flags 1
        [ 1st tape drive on 1st Xylogics 472 controller. ]
tape        xt1 at xtc1 drive 0 flags 2
        [ 1st tape drive on 2nd Xylogics 472 controller. ]
device      ar0 at mb0 csr 0x200 priority 3
        [ 1st 1/4" tape drive — see ar(4). ]
device      ar1 at mb0 csr 0x208 priority 3
        [ 2nd 1/4" tape drive. ]
device      cgtwo0 at mb0 csr vme busmem 0x400000 priority 3
        [ Sun-2 color graphics interface — see cgtwo(4s).
device      cgone0 at mb0 csr 0xec000 priority 3
        [ Sun-1 Color Board — see cgone(4s). ]
device      bwtwo0 at mb0 csr 0x700000 priority 4
        [ 1st monochrome monitor on a Sun-2/120 or Sun-2/170 — see bwtwo(4s). ]
device      bwtwo0 at mb0 csr vme obio 0x0 priority 4
        [ 1st monochrome monitor on a Sun-2/50 or Sun-2/160. ]
device      bwone0 at mb0 csr 0xc0000 priority 3
        [ 1st monochrome Sun-1 monitor — see bwone(4s). ]
device      vp0 at mb0 csr 0x400 priority 2
        [ Ikon Versatec Board — see vp(4). ]
device      vpc0 at mb0 csr 0x480 priority 2
        [ 1st Systech Centronics/Versatec Board — see vpc(4s).  ]
device      vpc1 at mb0 csr 0x500 priority 2
        [ 2nd Systech Centronics/Versatec Board. ]
device      pi0 at mb0 csr 0xee2000
        [ Parallel input.  Only used on Sun Models 100U and 150U, for keyboard and mouse. ]
device      des0 at mb0 csr all virt 0xee1800
        [ DES chip — see des(4s). ]
device      tod0 at mb0 csr 0xee1000
        [ Time of day clock on the Sun-2/120 or Sun-2/170. ]
device      tod0 at mb0 csr vme busmem 0x200800
        [ Time of day clock on the Sun-2/160 or Sun-2/50. ]
```

8.3. Kernel Configuration Procedures

Now we begin the actual walkthrough.

The walkthrough assumes you're familiar with the information presented above; assumes you have chosen a system configuration file to serve as your template for creating your own specific system configuration file; and, finally, assumes that the essential kernel source and object files are located in the */usr/sys* directory (this is how the system is shipped).

There are two paths: one for servers and one for standalone systems. The only difference is that the servers' path includes procedures for configuring a special kernel for clients. Please use the appropriate procedures for your system.

8.3.1. Kernel Configuration for Standalone Systems

1. Choose a name for your configuration of the kernel — in the following we will use *SYS_NAME* to indicate this name. Note that by convention this name is all upper case.
2. Change directory to the */usr/sys/conf* directory, and make a copy of the model configuration file you are using as a template (*TEMPLATE_NAME*) for your own specific configuration file.

   This copy is the basis for your own specific configuration file; it is named *SYS_NAME* and you will edit it to reflect your system specifications (so it must be 'writable'):

   ```
   tutorial# cd /usr/sys/conf
   tutorial# cp TEMPLATE_NAME SYS_NAME
   tutorial# chmod +w SYS_NAME
   ```

3. Create the *../SYS_NAME* directory to contain the kernel image. Remember: since the system build utility */etc/config* places its output files here, the directory **must** have the same name as your system configuration file:

   ```
   tutorial# mkdir ../SYS_NAME
   ```

4. Edit your new system configuration file to reflect your system specifications. This part of the procedure takes some care and thought. We suggest you:
   - Carefully look over the annotated copy of the *GENERIC* configuration file provided above to make sure you include all the mandatory general system description lines and all lines which are mandatory for Sun systems;
   - Re-read the ***Specific System Description Lines*** section to be sure you have used the correct "**config . . .**' line;
   - Look through the annotated *GENERIC*, again, to make sure you have included all lines for all the devices and pseudo-devices in your configuration (and only these lines).
5. When you have completed editing your configuration file, run the */etc/config* program (from */usr/sys/conf*) to generate the files needed to compile and link your kernel:

   ```
   tutorial# /etc/config SYS_NAME
   Don't forget to run make depend
   ```

   While ***config*** is running watch for any errors. Never use a kernel which ***config*** has complained about; the results are unpredictable.

   A successful run of ***config*** on your kernel configuration file generates a number of files in the kernel configuration directory (*../SYS_NAME*). These files are described in the introductory section above; unless you are curious about how the kernel's autoconfiguration scheme works, you should never have to look at any of them.
6. Now, change directory to your kernel configuration directory (*../SYS_NAME*). Then, use ***make***(1) to make source code dependencies and to build the new kernel:

```
tutorial# cd ../SYS_NAME
tutorial# make depend
[ lots of output ]
tutorial# make
[ lots of output ]
```

Note: if you have specified multiple bootable kernel images in your system configuration file with multiple **config** lines, and you wish to make only one of those multiple kernel images, type **make** *imagename* (where *imagename* is the *kernelname* specified on the **config** line of the configuration file) instead of simply **make** for the last command. Using **make** without arguments generates all kernels specified in the configuration file.

7. Now you can install your new kernel and try it out.

   First move the original kernel to another (safe) place, then copy the new kernel to the place of the original, and finally boot the system up with this new kernel. In the example, we assume you used **vmunix** for your *kernelname*; substitute your own kernel image name for **vmunix** if you used a different one:

```
tutorial# cp /vmunix /vmunix.old
tutorial# mv vmunix /vmunix
tutorial# /etc/halt
        The system goes through the halt sequence, then
    the monitor displays its prompt, at which point you
    can boot the system:
>b vmunix
        The system boots up multi-user, and then
    you can try things out.
```

8. Since at this point normal system performance is a highly, but not absolutely, certain indicator of a trouble-free kernel, if the system appears to work you may proceed with some confidence. You have successfully completed installation. Congratulations!

   If, on the other hand, the new kernel does not seem to be functioning properly, halt the system and boot from the original kernel. Then move the faulty kernel away and re-install the original in its place. Once you are booted up on the original, you can go about trying to fix the faulty kernel:

```
tutorial# /etc/halt
> b vmunix.old -s
tutorial# cd /
tutorial# mv vmunix vmunix.bad
tutorial# mv vmunix.old vmunix
tutorial# ^D              [ Brings the system up multi-user ]
```

8.3.2. Kernel Configuration for Servers

The following procedures assume you want to create two kernels: one for the server itself, and one for the all the clients. The clients' kernel configuration file includes the entire set of devices used by all the client systems. The kernel image is installed in */pub*.

1. Choose a name for your server's configuration of the kernel — in the following we will use *SERVER_NAME* to indicate this name. Note that by convention this name is all upper case.

2. Change directory to the */usr/sys/conf* directory, and make a copy of the model configuration file you are using as a template (*TEMPLATE_NAME*) for the server's own specific configuration file.

   This copy is the basis for your own specific configuration file; it is named *SERVER_NAME* and you will edit it to reflect your system specifications (so it must be 'writable'):

   ```
   tutorial# cd /usr/sys/conf
   tutorial# cp TEMPLATE_NAME SERVER_NAME
   tutorial# chmod +w SERVER_NAME
   ```

3. Create the *../SERVER_NAME* directory to contain the kernel image. Remember: since the system build utility */etc/config* places its output files here, the directory **must** have the same name as your system configuration file:

   ```
   tutorial# mkdir ../SERVER_NAME
   ```

4. Edit your new system configuration file to reflect your system specifications. This part of the procedure takes some care and thought. We suggest you:

   - Carefully look over the annotated copy of the *GENERIC* configuration file provided above to make sure you include all the mandatory general system description lines and all lines which are mandatory for Sun systems;
   - Re-read the ***Specific System Description Lines*** section to be sure you have used the correct "**config . . .**' line;
   - Look through the annotated *GENERIC*, again, to make sure you have included all lines for all the devices and pseudo-devices in your configuration (and only these lines).

5. When you have completed editing your configuration file, run the */etc/config* program (from */usr/sys/conf*) to generate the files needed to compile and link your kernel:

   ```
   tutorial# /etc/config SERVER_NAME
   Don't forget to run make depend
   ```

   While *config* is running watch for any errors. Never use a kernel which *config* has complained about; the results are unpredictable.

   A successful run of *config* on your kernel configuration file generates a number of files in the kernel configuration directory (*../SERVER_NAME*). These files are described in the introductory section above; unless you are curious about how the kernel's autoconfiguration scheme works, you should never have to look at any of them.

6. Now, change directory to your kernel configuration directory (*../SERVER_NAME*). Then, use *make*(1) to make source code dependencies and to build the new kernel:

   ```
   tutorial# cd ../SERVER_NAME
   tutorial# make depend

   [ lots of output ]

   tutorial# make

   [ lots of output ]
   ```

   Note: if you have specified multiple bootable kernel images in your system configuration file with multiple **config** lines, and you wish to make only one of those multiple kernel images, type **make** ***imagename*** (where ***imagename*** is the ***kernelname*** specified on the **config** line of the configuration file) instead of simply **make** for the last command. Using **make** without arguments generates all kernels specified in the configuration file.

7. Now prepare a kernel for your clients in the same way. When editing the configuration file (called *CLIENT_KERNEL_NAME* in the following), remember to include the entire set of devices used by all the machines:

```
tutorial# cd /usr/sys/conf
tutorial# cp TEMPLATE_NAME CLIENT_KERNEL_NAME
tutorial# chmod +w CLIENT_KERNEL_NAME
[ Edit CLIENT_KERNEL_NAME to reflect all clients' systems.
    Be especially careful with the device description lines. ]
tutorial# /etc/config CLIENT_KERNEL_NAME
tutorial# cd ../CLIENT_KERNEL_NAME
tutorial# make depend

[ lots of output ]

tutorial# make

[ lots of output ]
```

8. Now you can install both new kernels and try them out.

   - To install the server's kernel, first move the original kernel to another (safe) place, then copy the server's new kernel to the place of the original, and finally boot the server up with this new kernel. In the example, we assume you used **vmunix** for your ***kernel-name***; substitute your own kernel image name for **vmunix** if you used a different one:

```
tutorial# cd ../SERVER_NAME
tutorial# cp /vmunix /vmunix.old
tutorial# mv vmunix /vmunix
tutorial# /etc/halt
        The system goes through the halt sequence, then
    the monitor displays its prompt, at which point you
    can boot the system:
>b vmunix
        The system boots up multi-user, and then
    you can try things out.
```

   - To install the clients' kernel, **make sure all the clients are halted**, save the original kernel (if there is one), install the new kernel image in ***/pub***, and then test it out by booting up one of the clients:

```
tutorial# cd /usr/sys/CLIENT_KERNEL_NAME   [or wherever your client kernel is]
tutorial# cp /pub/vmunix /pub/vmunix.old
tutorial# mv vmunix /pub/vmunix

[ On the client machine: ]
>b vmunix
```

9. Since at this point normal system performance is a highly, but not absolutely, certain indicator of a trouble-free kernel, if your system(s) appears to work you may proceed with some confidence. You have successfully completed installation. Congratulations!

   If, on the other hand, either of the new kernels does not seem to be functioning properly, halt all systems and boot from the original kernel. Then move the faulty kernel away and re-install the original in its place. Once you are booted up on the original, you can go about trying to fix the faulty kernel. For example, on the server:

```
tutorial# /etc/halt
> b vmunix.old -s
tutorial# cd /
tutorial# mv vmunix vmunix.bad
tutorial# mv vmunix.old vmunix
tutorial# ^D             [ Brings the system up multi-user ]
```

Chapter 9

Installing UNIX on Tapeless Workstations

This chapter describes how to install UNIX on a standalone workstation which does not have a resident tape drive. To do this, you use the tape drive on another, fully installed machine — which we call your **remote host** — and perform the installation on your **target machine** across the Ethernet.

Please note that the remote host machine must either be configured as a server or a standalone machine; it may not be a client. If the machine is configured as a standalone, you must make it 'look' like a network disk server for remote installation. To do this, there is one primary requirement: the machine's kernel must have been generated from a system configuration file which includes the device description lines:

```
pseudo-device   ether
pseudo-device   nd
```

9.1. Overview of the Installation Procedure

A "remote installation" is very similar to standard UNIX installation; steps are:

1. Complete UNIX installation on your remote host. See chapters 1 through 8 of this document for procedures. As noted above, the remote host must either be configured as a server or a standalone machine, and must have the "pseudo-device nd" and "pseudo-device ether" lines included in its system configuration file.
2. If your remote host is configured as a standalone system, you must enable it as a server and turn its */usr* file system into a public network disk. If your remote host is configured as a server, this step is unnecessary.
3. Determine the network information necessary for installation. You will need to know the remote host's host number (in hexadecimal), and the target machine's hardware Ethernet address.
4. Load the mini file system onto the public disk of your remote host from the distribution tape.
5. Boot ***diag*** over the network; run ***diag*** to format (if necessary) and label your disk.

6. Boot the standalone ***copy*** program over the network. Run ***copy*** to copy a mini-file system over the network into the swap area on your disk.
7. Boot the mini UNIX system.
8. Edit the */etc/hosts* and */.rhosts* files.
9. Extract the root file system from the distribution tape.
10. Boot UNIX in single user state.
11. Run the ***setup*** program to extract the */usr* file system from the tape and initialize the network files.
12. Boot the full UNIX system.
13. Load optional software from the distribution tapes if you wish.
14. Reconfigure the system kernel. This step is mandatory for systems with only 1 MByte of memory, and highly recommended for systems with more.

Please look over the notes in Chapter 1 of this manual — especially the sections on *Definitions* and *Conventions* — before starting the actual installation procedure described below. Then, you are ready to begin.

9.2. Configuring the Remote Host as a Network Disk Server

PLEASE NOTE: Follow the procedures in this section only if you are using a standalone machine rather than a server as your remote host. You can proceed to the next section if your remote host is already configured as a server.

If the remote host machine is not configured as a network disk server, you need to turn the host's */usr* file system into a public network disk so that your target machine can access the files necessary for remote booting. Do the following on the remote host:

1. Edit the */etc/nd.local* file and add the following two lines. Remember to substitute the correct device abbreviation for the host's disk controller for ***disk*** (**ip** for an Interphase disk controller, or **xy** for a Xylogics disk controller, or **sd** for a SCSI disk controller):

```
user 0 0 /dev/disk0g 0 -1 -1
son
```

2. Enable the network disk server by typing the following (the argument to **MAKEDEV** below is three alpha characters "ndl" followed by 'zero' numeric):

```
host# cd /dev
host# MAKEDEV ndl0
host# /etc/nd - < /etc/nd.local
```

   Note that the device description lines mentioned above must be included in the machine's system configuration file for this to work.

3. Next, copy the files required for remote booting to the new public disk partition with the following commands. Remember to substitute the correct abbreviation for ***disk***:

```
host#  mkdir /usr/stand
host#  cp /stand/* /usr/stand
host#  ln -s /usr /pub
host#  cp /boot /pub/boot
host#  cd /usr/mdec
host#  installboot bootnd /dev/disk0g
host#  sync
host#
```

9.3. Determining Network Information

For later phases of remote installation, you need to know the remote host's host number (in hexadecimal) and the target's hardware Ethernet address. You must obtain this information now.

1. To determine the remote host's hexadecimal host number, find its entry in its own */etc/hosts* file. As you remember, entries consist of a machine's full Internet address (network number followed by host number) and name, for example:

```
192.9.200.48     augustus
192.9.200.50     julius
192.9.200.52     claudius
```

Here, julius' Internet address is 192.9.200.50; its network number is 192.9.200, and its host number (in decimal) is 50.

Since host numbers in */etc/hosts* are in decimal, and you need the remote host's host number in hexadecimal, you will need to convert. You can use *adb* for this if you wish:

```
host% adb
0thost_number_in_decimal = X
host_number_in_hex
^D
host%
```

2. To obtain the hardware Ethernet address of the target, power up the target workstation. You will see the PROM Monitor's power-up banner — which includes the hardware Ethernet address — and then the machine will start to auto boot. Stop the auto boot immediately by typing the appropriate abort sequence (if you don't know the abort sequence for the target machine, please see Chapter 1, *Abort Procedure*):

```
Self Test completed successfully.

Sun Workstation, [model type], [keyboard type]
ROM Rev N, some_number_MBytes memory installed
Serial #some_number, Ethernet address xx:xx:xx:xx:xx:xx

Auto-boot in progress . . .

[ abort by typing the appropriate abort sequence here ]

Abort at some address
>
```

Copy down the entire six bytes of the displayed Ethernet address.

9.4. Loading the Mini UNIX System on the Remote Host

Next, you copy the mini UNIX file system from the distribution tape to the remote host's public partition.

1. Load the distribution tape. If you have any questions about loading the tape, see the previous chapter, ***Loading the Bootstrap Program.***
2. Type the following on the remote host. Remember to replace ***tape*** with **mt** for the nine-track tape, **ar** for the Archive quarter-inch tape, or **st** for the SCSI tape controller. Also, if you are using a nine-track half-inch tape, use **20** for ***blk_factor*** ('bs=20b'); use **126** for a 1/4-inch tapes ('bs=126b'):

   ```
   host# mt -f /dev/nrtape0 rew
   host# mt -f /dev/nrtape0 fsf 3
   host# dd if=/dev/nrtape0 of=/pub/minifs bs=blk_factorb
   host# sync
   host#
   ```

   This takes about three minutes using a 1/2" tape, and less using a 1/4" cartridge.

9.5. Using *diag* to Format and Label the Target Machine's Disk

Now, you start to work on your target machine, and install UNIX from the remote host. The first step is to format your target's disk(s) with the ***diag*** utility.

Procedures for using ***diag*** in this remote installation are identical to those for a standard UNIX installation with one exception: during a standard installation you boot ***diag*** from the distribution tape; here, you boot from your remote host.

Boot ***diag*** from the remote host with the following boot command. Remember to replace ***e_interface*** with the appropriate abbreviation for your Ethernet controller (**ec** for the 3COM Ethernet Controller, or **ie** for the Sun-2 Ethernet Controller); replace ***host_number*** with the remote host's hexadecimal host number (obtained earlier). If you have more than one Ethernet Controller Board in your system, and you are booting from the second, third, etc., replace the "**0**" in the command with the controller's address on the Multibus (in hex).

```
> b e_interface(0,host_number)stand/diag
```

When you type this, the monitor boots ***diag*** from the network disk server. When ***diag*** starts up, it displays a sign on message:

```
Version sccs version_number and date
Disk Initialization and Diagnosis

When asked if you are sure, respond with 'y' or 'Y'
```

Once you see this sign on message, you can continue with the procedures given in Chapter 4 of this manual, step number 2 of the first section, ***Phase One: Specifying Hardware Configuration.*** Procedures are identical to those given in Chapter 4. When you're done, return to this point.

9.6. Loading the Mini UNIX System

When you're done formatting and labeling your disk(s), you're ready to load the mini UNIX system from the remote host to your disk. To do this, you use the standalone *copy* program which you boot from the remote host. Proceed as follows.

1. Boot the standalone *copy* program from the remote host with the following commands. Replace *e_interface* with the proper device abbreviation for your Ethernet controller, and replace *host_number* with the remote host's host number (in hex). Also, if you are not booting from the first Ethernet Controller Board in your system, use the board's Multibus address (in hexadecimal) rather than "0" in the boot command:

```
> b e_interface(0,host_number)stand/copy
Boot: e_interface(0,host_number)stand/copy
Load: e_interface(0,host_number,0)boot

Boot: e_interface(0,host_number,0)stand/copy
[ ...messages displaying sizes of copy program... ]
```

2. The *copy* program prompts you for the source (`From:`) and destination (`To:`) of the copy. When you respond to its queries, remember to substitute the correct device abbreviations for *e_interface* and *disk*:

```
Standalone Copy
From: e_interface(0,host_number,0)minifs
To: disk(0,0,1)
```

   Copying in the mini UNIX system takes about four minutes using a half-inch tape, and about seven minutes using a quarter-inch cartridge. This process loads the mini UNIX file system into the swap area on the disk. When it completes, the *copy* program returns control to the monitor:

```
Copy completed
>
```

9.7. Booting the Mini UNIX System

The next step is to boot UNIX in single-user state and specify the location of its root file system.

1. First, bring in the main boot program:

```
> b e_interface(0,host_number)boot
```

2. Now you can tell the the bootstrap program to boot the mini UNIX system from your own disk. Because this boot is an unusual one, you must specify the **-a** (for ask me) option on the boot command, and also the **-s** (come up single user) option, as follows:

```
Boot: disk(0,0,1)vmunix -as
Size: 366592+61440+98828 bytes   [ numbers vary with system level ]
Sun UNIX 4.2 (GENERIC) #145: Tue Sept 11 20:35:13 PDT 1984
Copyright (c) 1984 by Sun Microsystems, Inc.

[ ...about a dozen lines of configuration messages... ]

root device?
```

2. As the mini UNIX system comes up, it displays some messages about the configuration of the system on which it is running, and finally queries you, asking for its root file system. The root file system at this stage is "*disk***0***", which has a special meaning to the mini UNIX system. Since this notation looks ambiguous, let me specify: if you have a Xylogics disk controller, your root device is **xy0***; if you have an Interphase controller, it is **ip0***; and if you have a SCSI disk controller, it is **sd0*** — the asterisk is part of the device name:

   ```
   root device? disk0*
   ```

Depending upon your hardware, you may be asked to set the date at this point:

```
using number buffers containing number bytes of main memory
WARNING: no tod clock -- CHECK AND RESET THE DATE!
```

If so, continue with the section *Setting the Date* in Chapter 5, and return to this point when you've finished that subsection; otherwise, continue with the next section.

9.8. Editing the */etc/hosts* and */.rhosts* Files

Now, edit the ***/etc/hosts*** file on both the remote host and target machines, to make them aware of each other's existence. Also, edit ***/.rhosts*** on the remote host only. Proceed as follows:

1. Edit ***/etc/hosts*** on the remote host machine, and add an entry for the target machine. You will remember that entries in ***/etc/hosts*** consist of each machine's full Internet address (network number and host number), and name. Taking an earlier example:

   ```
   192.9.200.48    augustus
   192.9.200.50    julius
   192.9.200.52    claudius
   ```

   Here, julius' Internet address is 192.9.200.50; its network number is 192.9.200, and its host number is 50 (if these terms are confusing to you, please see Chapter 1, *Network Terminology.*)

   When you add an entry for the target, please remember that the target's host number **must be between 1 and 255, and must be unique on your network**. Also **copy down the target's full Internet address**: you will need it during the ***setup*** phase of remote installation which follows.

2. Still on the remote host machine, edit the ***/.rhosts*** file, adding an entry (hostname only) for the target machine. This will allow you to perform remote processes 'on' the remote host 'from' the target machine at the super-user level (for example, the remote extract in the next phase of installation). If the file does not exist, create it.

3. Now edit ***/etc/hosts*** (which should be nearly empty) on the target machine, and add the entries for the target and the remote host.

4. Run the */etc/ifconfig* program on the target machine:

   ```
   # /etc/ifconfig e_interface0 your_target_name
   ```

9.9. Loading the Root File System

Now you can run the *rxtr* (remote extract) shell script on the target machine to copy the root file system across the Ethernet and write it at the proper location on your disk.

1. Remount the distribution tape on the remote host's tape drive (if it is not still mounted).
2. Run *rxtr* by typing the following command from the target machine's root directory. Replace ***disk*** and ***tape*** with the correct device abbreviations, and ***remote_host_name*** with the machine name of your remote host:

```
target# disk=disk tape=tape host=remote_host_name rxtr
[ . . .incredible amount of messages . . . ]
```

3. During the first part of the extraction process, which takes about 30 minutes using a 1/4" cartridge and about 15 using a 1/2" tape, *rxtr* displays your disk super-block back-up numbers. These are crucial if you ever need to repair a corrupted disk, so copy them down:

```
+ cd /dev
+ ./MAKEDEV disk0
[ ...Possible messages from MAKEDEV.
  You may ignore any Warning messages here... ]
[ When the superblock backup numbers are displayed copy them down: ]
super-block backups (for fsck  -b#) at:
        32,  3048, 6064, 9080, 12096, 15112 [ numbers vary with disk type ]
+ sync
+ /etc/fsck  /dev/rdisk0a
[ ...and so on... ]
Root filesystem extracted
target#
```

At this point, a complete UNIX root file system has been copied onto your disk.

9.10. Booting the UNIX System

1. Now type a couple of **sync** commands to flush all I/O activity to the disks, then get back to the monitor by typing an abort sequence (see Chapter 1 for the abort sequence appropriate to your model). The monitor responds by displaying a message like:

```
target# sync
target# sync
target# abort using the appropriate abort sequence here
Abort at some address
>
```

2. When you see the monitor > sign, boot the UNIX system in single-user state:

```
> b vmunix -s
Boot: disk(0,0,0)vmunix -s
Load: disk(0,0,0)boot
Boot: disk(0,0,0)vmunix
Size: 366592+61440+98828 bytes    [ numbers vary with system level ]
Sun UNIX 4.2 (GENERIC) #145: Tue Sept 11 20:35:13 PDT 1984
Copyright (c) 1984 by Sun Microsystems, Inc.
[ ...Several lines of configuration messages... ]
#
```

You are now up and running UNIX as the single, super-user. The next job is to configure your system and load the */usr* file system.

9.11. Using *setup* to Configure Your System

At this point you invoke the ***setup*** program to configure your system. If you need an introduction to ***setup***, please see Chapter 6, ***Using Setup to Configure Your System.*** The following is the ***setup*** dialogue for remote installations.

> Note that if you are converting a standalone workstation to use NFS, do NOT run ***setup***. Instead, continue with Section 10.2, Step 4.

Please remember that you will be prompted for the target's hostname, network address, and host number; and for the hostname and host number of your remote host. Also, you will be asked for your domain name. Have this information at hand before starting ***setup***.

Begin the program by typing the ***setup*** command. ***setup*** begins by requesting global information:

```
target# setup

Sun Microsystems Configuration System

Global Information

     1) network disk server
     2) standalone
     3) standalone with remote tape

Enter the number for your environment: 3

You have a standalone system with remote tape; is this correct? (y/n): y

Host Information

Enter your hostname: your_hostname

Network numbers may be either class A, B or C
Their formats are:

    Class A:  nnn          (  0 <= nnn <= 127)
    Class B:  nnn.b1       (128 <= nnn <= 191)
    Class C:  nnn.b1.b2    (192 <= nnn <= 255)

b1 and b2 are one byte (0-255) quantities

Enter your network number (default is 192.9.200): network number

Your network number is number entered; is this correct? (y/n): y
```

```
Enter your host number

This is a number between 1 and 255: target's_host_number

Your hostname and host number are:

        target's_hostname:     target's_host_number

Is this correct? (y/n): y

Do you want to pick a domain name? (y/n): y or n

[ If y ]
Enter the domain name: domain_name

Tape Information

        1) 1/4" SCSI tape (st)
        2) 1/2" Magnetic tape (mt)
        3) 1/4" Archive tape (ar)

Enter the number for the type of tape: (1-3): number

You have specified a tape type; is this correct? (y/n): y

Enter the name of the remote host that the
tape type is attached to: remote_host_name

Enter the host number for remote_host_name: remote_host_number

The tape type is attached to remote_host_name with host number remote_host_number

Is this correct? (y/n): y

You have completed the configuration questions.
Continuing will destroy any existing files under /usr
and any client partitions if you are a server.

Do you want to begin configuration? (y/n): y

Updating /etc/hosts
[ ...A few lines of configuration messages... ]
```

If your distribution is on 1/4-inch tape cartridges, ***setup*** will prompt you to change to the second cartridge about two minutes after it begins its back-end routine. Insert the second cartridge and type 'RETURN' to continue the routine; it takes approximately 25 minutes to complete.

When ***setup*** completes its back-end work, your shell prompt returns. **If you are installing a yellow pages master or a yellow pages slave server, continue.** If you are not installing a yellow pages master, continue your installation by booting the full UNIX system, as described in the next section.

If the machine you are installing is going to be a yellow pages master server or a yellow pages slave server, you must prepare it for that role now.

If it is to be a master yellow pages master server, perform the following:

Move to the */etc* directory and execute the following:

```
# cd /etc
# mount /usr
# /bin/csh
# /bin/hostname hostname
# /bin/domainname domainname
# ifconfig e_interface hostname
# cd /etc/yp
# ypinit -m
```

If the machine is to be a yellow pages slave server, perform the above steps with the following variations:

a) Ensure that its yellow pages master is up and running.

b) Edit the file */etc/hosts* and add the following line:

   internet_address master_name

   where the internet address is that of the slave server, and where the *internet_address* and the *master name* are separated by a tab character.

c) Replace the line `ypinit -m` with the line:

   ```
   ypinit -s master_name
   ```

The program *ypinit* leads you through an interactive session. The following example shows the dialog for a yellow pages master; the case for a yellow pages slave server is slightly different:

```
Installing ypdata will require that you answer a few questions.
Questions will all be asked at the beginning of the procedure.

Do you want this procedure to quit on non-fatal errors [y/n: n] n

At this point, we have to construct a list of the hosts which will run
yp servers.  hostname is the list of yp server hosts.  Please
continue to add the names for other hosts, one per line.  When you are
done with the list, type a <ctl D>.

   Next host to add: some host name
   [ and so on... ]
   Next host to add: <ctl d>

The current list of servers looks like this:


hostname
some host name
[ and so on... ]

Is this correct? [y/n: y] y

There will be no more questions.  The remainder of this procedure
should take 5 to 10 minutes.

Building...

[ list... takes a while ]

hostname has been setup as a yp master server without any errors.

If there are running slave yp servers, run yppush for any data bases
which have been changed.  If there are no running slaves, run ypinit on
those hosts which are to be slave servers.

#
```

Note that you need not worry about the action indicated in the last paragraph printed by ***ypinit***; it occurs automatically when you reboot.

For additional information on yellow pages, see the "System Administration Manual".

9.12. Booting the Full UNIX System

When ***setup*** finishes its back-end routine, you have a complete ***/usr*** file system on disk, and all machine-specific files have been initialized. If you completed the instructions for yellow pages installation, it should also be ready to run. Finally, you boot the full UNIX system.

1. First, halt the system, using the ***/etc/halt*** command. This shuts the system down in an orderly manner, and returns control to the monitor:

```
tutorial# /etc/halt
Syncing disks . . . .  done
UNIX halted

>
```

2. Now you can simply boot the UNIX system:

```
> b
Boot: disk(0,0,0)vmunix
Load: disk(0,0,0)boot
Boot: disk(0,0,0)vmunix
Size: 366592+61440+98828 bytes
Sun UNIX 4.2 (GENERIC) #145: Tue Feb 21 20:35:13 PDT 1984
Copyright (c) 1984 by Sun Microsystems, Inc.

[  . . . Many lines of configuration messages . . . ]

tutorial login: root
tutorial#
```

You can now use the full UNIX system on this machine.

If you'd like to load the any of the optional software provided with the distribution to your target's disk, follow the procedures in Chapter 7, *Loading Optional Software.* Otherwise, see Chapter 8 for kernel configuration procedures.

Chapter 10

Converting Systems to Use the NFS

This chapter describes how to convert pre-NFS systems with UNIX already installed, to operate with the NFS. While this chapter refers to many places in previous chapters, it is not part of the normal installation procedure described there.

This chapter describes how to convert two different types of systems; nd servers and their diskless clients, and diskful workstations.

10.1. Converting nd Servers and their Diskless Clients

An nd server and its diskless clients function together as an entity. This means that as the server is converted, the clients must be also. Note that steps 1 through 6 of this procedure concern mostly the server, and step 7 deals specifically with the clients' partitions on the server.

Because machines need ***nd*** to boot, an NFS server still needs a ***/pub*** partition. However, unlike an *nd*-only configuration, where ***/pub*** contains ***/usr/bin***, ***/usr/ucb***, ***/usr/etc***, and so on, under NFS ***/pub*** only contains ***/pub/vmunix***, ***/pub/boot***, ***/pub/stand*** and ***/pub/bin***. There is a separate file system mounted on ***/usr*** which contains all ***/usr*** binaries. In an ***nd***-only configuration, ***/usr/bin*** was a symbolic link to ***/pub/usr/bin***. In the NFS, ***/usr/bin*** is no longer in ***/pub***. Instead, the server gets ***/usr/bin*** off its own disk, while a client gets it because he has mounted the server's ***/usr*** onto his ***/usr***. Similarly, a client now mounts ***/lib*** from the server's ***/lib*** rather than the ***nd*** style of having ***/lib*** symbolically linked to ***/pub/lib***. The other standard remote mount is called ***/usr2***, where users' home directories live in the NFS scheme.

This creates a potential problem when a client mounts a server's ***/usr*** on his directory. Some files in ***/usr*** should be private, such as ***/usr/adm***, ***/usr/spool***, ***/usr/tmp***. To get around the problem in the NFS, these private files are symbolic links to ***/private/usr***. In an ***nd*** configuration, there were a few files from ***/usr/lib***, such as ***crontab***, ***aliases***, and ***sendmail.cf*** that were private; they are now symbolic links to ***/private/usr/lib***. Thus, when you convert to NFS ***setup*** puts ***/private/crontab*** and the others in new locations like ***/private/usr/lib/crontab*** and so forth.

A layout of the 2.0 file system on a diskless client, as displayed by the ***df*** and ***ls*** commands, appears later in this chapter.

Installation

Converting from an *nd*-only system to an NFS system is similar to doing a full upgrade from one version of an *nd*-only release to another. Most of the changes to where files live are handled automatically by the new version of ***setup***.

Use the following steps on an ***nd*** server when upgrading to NFS:

(1) Warn Users

Warn users that they should move all files they wish to save into their home directories. The only files outside of their home directories that will be saved are ***/usr/lib/crontab***, */usr/spool/mail*, and, as explained below, part of ***/etc/passwd***. Remember, ***/etc/hosts***, ***/etc/rc*** and so on will be lost.

(2) Dump

Dump the entire disk(s), making sure to dump all partitions.

(3) Label

You must relabel the disk but not reformat it. In the NFS, the **g** partition (which contains ***/pub*** and ***nd*** partitions) is much smaller than it used to be. ***Diag*** provides a built-in label called `alternate` (see Chapter 4). Use it if you don't want to make your own label. It is designed for 12 users on an Eagle, 4 users on a Fujitsu 2284 disk. If you choose not to use `alternate`, and would rather make your own label, read on.

The **a**, **b**, and **c** partitions should be left the same size they are in the default label (which is also what they are in the new `alternate` label). To compute the size of the other partitions, remember the following: you typically want ***/pub*** to be 6 MB and each user's ***nd*** partition to be 5 MB. For example, if you want to set up an Eagle with 7 users, each with a 12 MB swap space, then you will need `6 + 7*(5+12) = 125 MB` of space for the **g** partition. The **d** partition is for ***/usr***. The 2.0 distribution tape uses 22 MB of ***/usr***, so you will want to make the **d** partition at least 30 MB. (If you want space for man pages or games, remember to allow extra space as explained in the installation manual.) Any space left should be put into the **e** partition, which is where users' home directories should go. In this scheme, the **f** and **h** partitions are unused.

(4) Run *setup*

Run ***setup***, as explained in Chapter 6. Unlike pre-NFS versions, it now puts ***/usr/bin***, ***/usr/ucb***, etc. on the **d** partition of the disk (the partition where ***/usr*** is mounted), and it mounts ***/usr2*** on the **e** partition of the disk. In the NFS scheme, ***/usr2*** holds users' home directories, and users' files must be restored to that location in step 7 below. (Below we explain how to edit ***/etc/passwd*** to reflect the new location of users' home directories.)

(5) Install Yellow Pages Master

If this is the first machine you are converting to NFS, you should make it the "master" yellow pages server now. We list the necessary steps here; for more explanation see the "System Administration Manual".

The following example shows a machine called ***galaxy***, with a yellow pages domain called ***kodak***. On ***galaxy***, type the following (remember to substitute the Ethernet board type and Ethernet controller number in the examples):

```
galaxy# cd /etc
galaxy# mount /usr
galaxy# /bin/csh
galaxy# /bin/hostname galaxy
galaxy# /bin/domainname kodak
galaxy# ifconfig e_interface# galaxy
```

Be sure to use your own host and domain names in place of `galaxy` and `kodak`.

Now restore the following six files from the dump tape you made earlier. These six will be served by the yellow pages server:

```
/etc/passwd
/etc/hosts
/etc/group
/etc/networks
/etc/protocols
/etc/services
```

(6) Edit *passwd*

You must edit the restored ***/etc/passwd*** file to change the location of users' home directories from ***/usr*** to ***/usr2***. The following example shows a typical user's password entry before:

```
clown:xBpRbBMaHg:1241:10:Bozo Clown:/usr/clown:/bin/csh
```

and after:

```
clown:xBpRbBMaHg:1241:10:Bozo Clown:/usr2/clown:/bin/csh
```

Now add an entry to the restored ***/etc/passwd*** file. Root-over-the-network requires a user id "`uid -2`". We recommend that the name for this entry be '`nobody`'. ***setup*** does this for you in the NFS releases. But to convert to NFS (as opposed to starting out-of-the-box), you restored your password file from a tape in a pre-NFS condition. You can get an example of the password entry from any password file created by ***setup***, or you can edit in the line:

```
nobody:*:-2:10::/:/bin/csh
```

Now install the master yellow pages server with the following:

```
galaxy# cd /etc/yp
galaxy# ypinit -m
```

(7) Restore On The Server

First restore the users' home directories on the server. Note that users' home directories are now in ***/usr2***. So, for example, user **john**, whose files were dumped as ***/usr/john*** must be restored as ***/usr2/john***. Do this for all clients of the server.

Next, since this is a server machine, add a ***find*** to periodically remove files of the form "`.nfs*`" if they have been around for more than 7 days.

Example:

```
find / -name .nfsjunk -mtime +7 -exec rm -f{} ;
```

(8) Restore On The Client Partition

After completing the restores for all the users, restore some files from each user's ***nd*** partition, and some files from the server's file system, onto the client's partitions.

Four files must be restored individually from each user's ***nd*** partition: ***/private/crontab***, ***/private/aliases***, ***/private/sendmail.cf***, and ***/usr/spool/mail***.

Restore ***/private/crontab*** to a user's ***nd*** partition as ***/private/usr/lib/crontab***.

Restore ***/private/aliases*** to ***/private/usr/lib/aliases***.

Restore */private/sendmail.cf* to */private/usr/lib/sendmail.cf.*

Restore */usr/spool/mail* to */private/usr/spool/mail.*

(9) Edit ***crontab***

Now edit the */private/usr/lib/crontab* file you restored and make three changes: 1) remove or comment out any *find* command that starts at "/" — the root. 2) remove or comment out any 'calendar' entry. 3) Change a line in each client's *crontab* file. Before you change it, it looks like:

```
15 4 * * * find /usr/preserve -mtime +7 -a -exec rm -f {} \;
```

Change it by adding a '/' (slash) at the end of preserve:

```
15 4 * * * find /usr/preserve/ -mtime +7 -a -exec rm -f {} \;
```

(10) Edit ***passwd*** **again**

Next, edit each client's */etc/passwd* to include the encrypted password for both root and the client. You can get these from a copy of */etc/passwd* on the server. Put the encrypted characters in the password field of the client's */etc/passwd.* For example, from an entry like this:

```
root:rtyBcEpNhHqz:0:10:Deus:/:/bin/csh
```

you would copy 'rtyBcEpNhHqz' into the client's password field, without needing to know the unencrypted password. Remember that the home directory field for each user in */etc/passwd* was set up to be */usr2* under NFS. For example, make sure the field looks like this for each client:

```
zippy:OVceErnaqI:1492:10:Zippy the Hacker:/usr2/zippy:/bin/csh
```

(11) Restore more files

Restore the following files from the server's file system, unless the client has special versions of his own he wants restored instead: */etc/printcap, /etc/ttys, /etc/ttytypes,* and */etc/remote.* You may also have others at your site.

If the client is not going to use the yellow pages, then also restore the following files: */etc/group, /etc/services, /etc/protocols, /etc/networks, /etc/hosts* to the client partition.

Finally, any files peculiar to your disk (*/usr/local/emacs* for example), should be restored onto */usr.* However, almost all programs must be recompiled first, since 1.x binaries will not normally run on 2.0 systems.

Daemon Roll Call

This section tells what new daemons to expect. The 2.0 *setup* makes new */etc/rc* and */etc/rc.local* files that startup new daemons. The list of processes displayed by "`ps ax`" is very different in the 2.0 system, and could seem intimidating. Here are the new daemons and a short explanation. Unless otherwise noted they appear on both servers and clients:

1) */etc/portmap* is the internet daemon used by the remote procedure call world. This daemon must be alive and well since all rpc based services (like yellow pages and NFS) depend upon it. The command `/usr/etc/rpcinfo -p` will tell if your portmap daemon is ok.

2) */etc/ypbind* is necessary if, and only if, your machine uses the yellow pages. Therefore, if you do not wish to take advantage of the yellow pages, make sure that this daemon is not started. The *ypbind* daemon remembers where a yellow pages server is located so that each

new UNIX process need not try to locate a server when it needs one.

3) */etc/biod*, there are usually four of these daemons; they dive into the kernel and never return. The daemons perform the job of read-ahead and write-behind from and to network file systems. It exists only on NFS client machines. Any NFS client must have at least one ***biod*** process. ***biod*** means bio daemon; kernel writers recognize bio as meaning buffered input/output.

4) */etc/nfsd*, exists only on NFS servers, and is the server side of */etc/biod*.

5) */etc/ypserv*, this daemon implements the yellow pages server. Very few machines will run this daemon. For administrative convenience, each ***nd*** server will typically run a copy of */etc/ypserv*.

6) */usr/etc/rpc.rstatd*, this daemon is the server side of the remote speedometer tools. ***inetd*** only starts this daemon if somebody's speedometer tool is monitoring the machine.

7) */usr/etc/rpc.mountd*, is the server side of mount. It is started by ***inetd*** when a client does a ***mount*** or ***showmount*** to the server.

File System Layout In 2.0

To fully explain the new layout of the 2.0 file system on diskless clients, the output from ***df*** commands below shows the mounted file systems on a server and on one of its clients — notice where the client file systems are mounted from. Following that, the output from ***ls*** commands shows the contents of various directories. The machine "ganymede" is a diskless client of "titan":

```
ganymede% rsh titan df
Filesystem            kbytes    used   avail capacity  Mounted on
/dev/xy0a               7295    4178    2387    64%    /
/dev/xy0g               5695    3698    1427    72%    /pub
/dev/xy0d              39315   30945    4438    87%    /usr
/dev/xy2g             326215  226602   66991    77%    /usr2
/dev/xy2b              15899      44   14265     0%    /usr/doctools

ganymede% df
Filesystem            kbytes    used   avail capacity  Mounted on
/dev/nd0                4775    3813     484    89%    /
/dev/ndp0               5695    3698    1427    72%    /pub
titan:/lib              7295    4178    2387    64%    /lib
titan:/usr             39315   30945    4438    87%    /usr
titan:/usr2           326215  226602   66991    77%    /usr2
```

```
ganymede% ls -l /
total 124
lrwxrwxrwx  1 root          8 bin -> /pub/bin
drwxr-xr-x  2 root       1536 dev
drwxr-xr-x  3 bin        1536 etc
drwxr-xr-x  2 bin         512 lib
drwxr-xr-x  2 root       4096 lost+found
drwxr-xr-x  2 bin          24 mnt
drwxr-xr-x  3 root        512 private
drwxr-xr-x  5 root        512 pub
drwxrwxrwx  2 bin         512 tmp
drwxr-xr-x 16 root        512 usr
drwxr-xr-x 34 root       1024 usr2
lrwxrwxrwx  1 root         11 vmunix -> /pub/vmunix

ganymede% ls -l /pub
total 1448
drwxr-xr-x  2 bin        1024 bin
-rwxr-xr-x  1 root      22283 boot
drwxr-xr-x  2 root       4096 lost+found
drwxr-xr-x  2 bin         512 stand
-rwxr-xr-x  1 root     460800 vmunix

ganymede% ls -l /usr
total 34
lrwxrwxrwx  1 root         16 adm -> /private/usr/adm
drwxr-xr-x  2 bin        2048 bin
lrwxrwxrwx  1 root         18 crash -> /private/usr/crash
drwxr-xr-x  3 bin         512 dict
drwxrwxr-x  2 root         24 doctools
drwxr-xr-x  2 bin        1024 etc
drwxr-xr-x  2 bin        7680 hosts
drwxr-xr-x 22 bin        1536 include
drwxr-xr-x 17 bin        2560 lib
drwxrwxrwx  6 root       1024 local
drwxr-xr-x  2 root       4096 lost+found
drwxrwxrwx  2 root         24 man
drwxrwxr-x  2 root        512 mdec
lrwxrwxrwx  1 root         21 preserve -> /private/usr/preserve
drwxr-xr-x  2 bin         512 pub
drwxr-xr-x  3 bin         512 sccs
lrwxrwxrwx  1 root         18 spool -> /private/usr/spool
lrwxrwxrwx  1 root         16 tmp -> /private/usr/tmp
drwxr-xr-x  2 bin        1536 ucb

ganymede% ls -l /usr/lib | grep " ->"
lrwxrwxrwx  1 root         24 aliases -> /private/usr/lib/aliases
lrwxrwxrwx  1 root         28 aliases.dir -> /private/usr/lib/aliases.dir
lrwxrwxrwx  1 root         28 aliases.pag -> /private/usr/lib/aliases.pag
lrwxrwxrwx  1 root         24 crontab -> /private/usr/lib/crontab
lrwxrwxrwx  1 root         28 sendmail.cf -> /private/usr/lib/sendmail.cf
```

```
ganymede% ls -l /private/usr
total 7
drwxr-xr-x  2 bin          512 adm
drwxr-xr-x  2 bin           24 crash
drwxrwxrwx  2 root         512 lib
drwxr-xr-x  2 bin           24 preserve
drwxr-xr-x 12 bin          512 spool
drwxrwxrwx  2 bin          512 tmp

ganymede% ls -l /private/usr/lib
total 16
-rw-rw-rw-  1 root        1103 aliases
-rw-rw-rw-  1 root           0 aliases.dir
-rw-rw-rw-  1 root        1024 aliases.pag
-rw-r--r--  1 root         205 crontab
-r--r--r--  1 root       11828 sendmail.cf
```

10.2. Converting Diskful Workstations to use NFS

This section describes how to convert a diskful workstation to use NFS.

The NFS enables diskful workstations to reduce local disk storage by sharing the executable files in */usr* by mounting desired directories from an NFS server when the diskful workstation boots.

> The conversion process in this section is designed specifically for diskful workstations. To convert servers and diskless clients, use the instructions provided earlier in this chapter.

Use the following steps:

(1) Record Your User ID

Look in the */etc/passwd* file and record your uid (user id) for use later in this process.

(2) Dump

Dump your entire disk(s).

(3) Follow Install Manual Instructions

Follow the instructions in Chapters 1 through 5 up until the point it tells you to run ***setup***. Then, instead of running ***setup*** go to the next step.

(4) Edit *fstab*

Edit your */etc/fstab* file. If you have a Xylogics SMD disk, the entire file should look like the one below. If you have a SCSI disk, substitute `sd` for `xy` in the example: Note that you may have to use the "ed" editor, as "vi" may not be available (See *ed*(1)).

```
/dev/xy0a /      4.2 rw 1 1
/dev/xy0g /usr2 4.2 rw 1 2
nfs_server_name:/usr /usr nfs rw,hard 1 1
```

Be sure to replace ***nfs_server_name*** with the name of your NFS server.

(5) Edit *rc.local*

Edit *rc.local*, so that the lines

```
/bin/hostname nohostname
/bin/domainname nodomainname
```

have ***nohostname*** replaced with your hostname and ***nodomainname*** with your domainname. These are explained in Chapter 2.

(6) Edit *hosts*

Add a line for your machine to ***/etc/hosts*** that has the same format as the example below, but uses your own internet address and machine name:

internet_address ***machinename*** better add NFS server also

(7) Make new directories

Use *mkdir* to create the following directories:

```
/usr2
/private
/private/usr
/private/usr/adm
/private/usr/crash
/private/usr/preserve
/private/usr/spool
/private/usr/tmp
/private/usr/lib
```

(8) Run *newfs*

should have 10.1.5 here

Do the following ***/etc/newfs*** (remember to substitute the proper value for ~~***tape***~~ disk:

```
newfs /dev/rtapeg
```

disk run fsck

must do stuff described at end of chap 6 (a comment and yp line in rc.local)

(9) Reboot your machine

Now, prepare your machine for halting, take it down, and boot multiuser. Note that the default boot command (>**b**) should work.

Try out a few things to make sure they work, before you begin the next steps and start restoring your files.

(10) Make one more directory

Do the following ***mkdir***

```
mkdir /private/usr/spool/mqueue
```

(11) Create files

Use ***touch*** to create the following files:

```
/private/usr/adm/lastlog
/private/usr/adm/messages
/private/usr/adm/msgbuf
/private/usr/adm/shutdownlog
/private/usr/adm/usracct
/private/usr/adm/wtmp
```

(12) Restore files

From the dump tape you made earlier, restore your home directory files to ***/usr2***, which is mounted on the 'g' partition. Remember, files dumped as ***/usr/john*** must be restored as ***/usr2/john***.

Next, restore the following files individually:

Restore ***/usr/lib/crontab*** to your ***nd*** partition as ***/private/usr/lib/crontab***.

Restore ***/usr/lib/aliases*** to ***private/usr/lib/aliases***.

Restore all of ***/usr/spool*** to ***/private/usr/spool***. **(13) Edit *crontab***

Edit the ***/private/usr/lib/crontab*** file you restored and make three changes: 1) remove or comment out any ***find*** command that starts at "/" — the root. 2) remove or comment out any 'calendar' entry. 3) Change a line in ***crontab*** as follows:

Change:

```
15 4 * * * find /usr/preserve -mtime +7 -a -exec rm -f {} \;
```

Change it by adding a '/' (slash) at the end of preserve:

```
15 4 * * * find /usr/preserve/ -mtime +7 -a -exec rm -f {} \;
```

(14) Install *sendmail*

Now, you must install ***sendmail*** as described in the "System Administration Manual". Note that the files ***/usr/lib/sendmail.main.cf*** and ***/usr/lib/sendmail.subsidiary.cf*** should already be in /usr/lib, which is symbolically linked to /private/usr/lib. not true usr/lib/sendmail.cf fc

(15) Edit *passwd*

restore orig passwd

Next, edit your ***/etc/passwd*** to include the uid that you recorded above, and make sure that the home directory field for each user is ***/usr2*** instead of ***/usr***. Note that eventually you will want to assign passwords to both root and yourself; you can do this with ***passwd***.

For example, make sure the field looks like this:

```
zippy:OVceErnaqI:1492:10:Zippy the Hacker:/usr2/zippy:/bin/csh
```

(16) Restore more files

Now, restore the following files. Note that, unlike the files restored above, these return to their original locations. The files are: ***/etc/printcap***, ***/etc/ttys***, ***/etc/ttytypes***, ***/etc/remote*** (you may have others).

If you do not plan to use the yellow pages, then also restore the following files: ***/etc/group***, ***/etc/services***, ***/etc/protocols***, ***/etc/networks***, ***/etc/hosts***.

Now, restore other files peculiar to your disk. For example, ***/usr/lib/emacs*** should be restored to ***/usr***. Note however, that almost all programs must be recompiled as 1.x binaries do not normally run on a 2.0 system.

Appendix A

Sample Configuration Files

The following pages present several sample configuration files for several basic machine configurations. The files are close, if not exact, renderings of the following files in your */usr/sys/conf* directory (as the system is shipped). We list them here with the basic workstation models they correspond to:

ND100	Diskless, tapeless Sun-2/100U with Ethernet, Sun-1 keyboard, and Sun-1 mouse. This is a typical client on a local network.
ND120	Diskless, tapeless Sun-2/120 with Ethernet, Sun-2 mouse and keyboard, and a Sky FFP Board. This is a typical Sun-2 client with added number-crunching capability.
ND50	Diskless, tapeless Sun-2/50 with Ethernet, Sun-2 mouse and keyboard. This is a typical Sun-2 client on a local network.
SDST120	Standalone Sun-2/120. This workstation has its own SCSI disk and SCSI tape. In addition, it is connected to an Ethernet, and has a Sky FFP Board. This is a typical Sun-2 standalone workstation.
SDST160	Sun-2/160 with two SCSI disks and one SCSI tape — configured as a fileserver.
XY100	Sun-2/100U Fileserver. Sun-2/100U with one SMD disk attached to a Xylogics Controller, Ethernet, Sun-1 mouse and keyboard. This machine might provide storage to several clients on the local network.
XYAR100	Sun-2/100U Fileserver with 1/4" tape. The addition of the tape drive makes this machine suitable for use as a server for software installations, as well as providing some back-up capability.
XYMT150	Sun-2/150U Fileserver with two SMD disks and 1/2" tape. This is a typical configuration for a server on a local network. Two disks provide storage for several clients, and the tape serves for installation as well as backups.
XYMT160	Sun-2/160 Fileserver with up to four SMD disks, two SCSI disks, 1/2" tape and SCSI tape. This template file for a fileserver includes most supported disk and tape devices; choose the appropriate devices and edit accordingly.

The files are intended to help you understand how the kernel configuration file 'works', and to give you some templates for basic configurations. We advise against simply copying one of the files into place during configuration — please make sure your machine has EXACTLY the devices described in the file, and that the "config", "ident", and "options" lines are correct for your system.

```
#
# Diskless Model 100
#
machine         sun
cpu             "SUN2"
ident           "ND100"
timezone        8 dst
maxusers        2
options         INET
options         RPC
options         NFS

config          vmunix    root on nd

pseudo-device   rpc
pseudo-device   nfs
pseudo-device   pty
pseudo-device   inet
pseudo-device   ether
pseudo-device   loop
pseudo-device   nd
pseudo-device   win32
pseudo-device   dtop1
pseudo-device   ms1
pseudo-device   kb1
controller      mb0 at nexus ?
device          ropc0 at mb0 csr 0xee0800
device          zs0 at mb0 csr 0xeec800 flags 3 priority 2 # cpu ports
device          ec0 at mb0 csr 0xe0000 priority 3
device          bwone0 at mb0 csr 0xc0000 priority 3
device          pi0 at mb0 csr 0xee2000
device          tod0 at mb0 csr 0xee1000
```

```
#
# Diskless Model 120
#
machine         sun
cpu             "SUN2"
ident           "ND120"
timezone        8 dst
maxusers        2
options         INET
options         RPC
options         NFS

config          vmunix    root on nd

pseudo-device   rpc
pseudo-device   nfs
pseudo-device   pty
pseudo-device   inet
pseudo-device   ether
pseudo-device   loop
pseudo-device   nd
pseudo-device   win32
pseudo-device   dtop1
pseudo-device   ms1
pseudo-device   kb1
controller      mb0 at nexus ?
device          ropc0 at mb0 csr 0xee0800
device          sky0 at mb0 csr 0x2000 priority 2
device          zs0 at mb0 csr 0xeec800 flags 3 priority 2 # cpu ports
device          zs1 at mb0 csr 0xeec000 flags 3 priority 2 # video ports
device          ec0 at mb0 csr 0xe0000 priority 3
device          ie0 at mb0 csr 0x88000 priority 3
device          bwtwo0 at mb0 csr 0x700000 priority 3
device          tod0 at mb0 csr 0xee1000
```

```
#
# Diskless Model 50
#
machine         sun
cpu             "SUN2"
ident           "ND50"
timezone        8 dst
maxusers        4
options         INET
options         SYSACCT
options         RPC
options         NFS

config          vmunix    root on nd

pseudo-device   rpc
pseudo-device   nfs
pseudo-device   pty
pseudo-device   inet
pseudo-device   ether
pseudo-device   loop
pseudo-device   nd
pseudo-device   win128
pseudo-device   dtop4
pseudo-device   ms3
pseudo-device   kb3
pseudo-device   ingres
device          zs0 at mb0 csr all virt 0xeec800 flags 3 priority 3 # cpu
device          zs1 at mb0 csr all virt 0xeec000 flags 0x103 priority 3 # video
device          ie0 at mb0 csr vme virt 0xee3000 priority 3
device          bwtwo0 at mb0 csr vme obio 0x0 priority 4
```

```
#
# Model 120 with one SCSI disk and tape
#
machine         sun
cpu             "SUN2"
ident           "SDST120"
timezone        8 dst
maxusers        2
options         INET
options         RPC
options         NFS

config          vmunix    root on sd

pseudo-device   rpc
pseudo-device   nfs
pseudo-device   pty
pseudo-device   inet
pseudo-device   ether
pseudo-device   loop
pseudo-device   win32
pseudo-device   dtop1
pseudo-device   ms1
pseudo-device   kb1
controller      mb0 at nexus ?
controller      sc0 at mb0 csr 0x80000 priority 2
disk            sd0 at sc0 drive 0 flags 0
tape            st0 at sc0 drive 32 flags 1
device          ropc0 at mb0 csr 0xee0800
device          sky0 at mb0 csr 0x2000 priority 2
device          zs0 at mb0 csr 0xeec800 flags 3 priority 2 # cpu ports
device          zs1 at mb0 csr 0xeec000 flags 3 priority 2 # video ports
device          zs2 at mb0 csr 0x80800 flags 3 priority 2
device          zs3 at mb0 csr 0x81000 flags 3 priority 2
device          ec0 at mb0 csr 0xe0000 priority 3
device          ie0 at mb0 csr 0x88000 priority 3
device          bwtwo0 at mb0 csr 0x700000 priority 3
device          tod0 at mb0 csr 0xee1000
```

```
#
# Model 160 with 1 or 2 SCSI disks and 1 SCSI tape
#
machine           sun
cpu               "SUN2"
ident             "SDST160"
timezone          8 dst
maxusers          4
options           INET
options           RPC
options           NFS

config            vmunix    root on sd0 swap on sd0 and sd1

pseudo-device     rpc
pseudo-device     nfs
pseudo-device     pty
pseudo-device     inet
pseudo-device     ether
pseudo-device     loop
pseudo-device     nd
pseudo-device     win128
pseudo-device     dtop4
pseudo-device     ms3
pseudo-device     kb3
pseudo-device     ingres
controller        mb0 at nexus ?
controller        sc0 at mb0 csr vme busmem 0x200000 priority 2 vector scintr 64
disk              sd0 at sc0 drive 0 flags 0
disk              sd1 at sc0 drive 1 flags 0
tape              st0 at sc0 drive 32 flags 1
device            sky0 at mb0 csr vme busio 0x8000 priority 2 vector skyintr 168
device            zs0 at mb0 csr all virt 0x0eec800 flags 3 priority 3 # cpu
device            zs1 at mb0 csr all virt 0x0eec000 flags 0x103 priority 3 # video
device            ie0 at mb0 csr vme virt 0x0ee3000 priority 3
device            cgtwo0 at mb0 csr vme busmem 0x400000 priority 3
device            bwtwo0 at mb0 csr vme obio 0x0 priority 4
device            tod0 at mb0 csr vme busmem 0x200800
```

```
#
# Model 100 with one Xylogics disk
#
machine         sun
cpu             "SUN2"
ident           "XY100"
timezone        8 dst
maxusers        2
options         INET
options         RPC
options         NFS

config          vmunix    root on xy

pseudo-device   rpc
pseudo-device   nfs
pseudo-device   pty
pseudo-device   inet
pseudo-device   ether
pseudo-device   loop
pseudo-device   win32
pseudo-device   dtop1
pseudo-device   ms1
pseudo-device   kb1
controller      mb0 at nexus ?
controller      xyc0 at mb0 csr all virt 0xebee40 priority 2
disk            xy0 at xyc0 drive 0
device          ropc0 at mb0 csr 0xee0800
device          zs0 at mb0 csr 0xeec800 flags 3 priority 2 # cpu ports
device          ec0 at mb0 csr 0xe0000 priority 3
device          bwone0 at mb0 csr 0xc0000 priority 3
device          pi0 at mb0 csr 0xee2000
device          tod0 at mb0 csr 0xee1000
```

```
#
# Model 100 with one Xylogics disk and Archive tape
#
machine         sun
cpu             "SUN2"
ident           "XYAR100"
timezone        8 dst
maxusers        2
options         INET
options         RPC
options         NFS

config          vmunix    root on xy

pseudo-device   rpc
pseudo-device   nfs
pseudo-device   pty
pseudo-device   inet
pseudo-device   ether
pseudo-device   loop
pseudo-device   win32
pseudo-device   dtop1
pseudo-device   ms1
pseudo-device   kb1
controller      mb0 at nexus ?
controller      xyc0 at mb0 csr all virt 0xebee40 priority 2
disk            xy0 at xyc0 drive 0
device          ropc0 at mb0 csr 0xee0800
device          zs0 at mb0 csr 0xeec800 flags 3 priority 2 # cpu ports
device          ec0 at mb0 csr 0xe0000 priority 3
device          ar0 at mb0 csr 0x200 priority 3
device          bwone0 at mb0 csr 0xc0000 priority 3
device          pi0 at mb0 csr 0xee2000
device          tod0 at mb0 csr 0xee1000
```

```
#
# Model 150 server with two Xylogics disks and one 1/2" tape
#
machine         sun
cpu             "SUN2"
ident           "XYMT150"
timezone        8 dst
maxusers        8
options         INET
options         SYSACCT
options         RPC
options         NFS

config          vmunix    root on xy

pseudo-device   rpc
pseudo-device   nfs
pseudo-device   pty
pseudo-device   bk
pseudo-device   sysacct
pseudo-device   inet
pseudo-device   ether
pseudo-device   loop
pseudo-device   nd
pseudo-device   win32
pseudo-device   dtop1
pseudo-device   ms1
pseudo-device   kb1
controller      mb0 at nexus ?
controller      xyc0 at mb0 csr all virt 0xebee40 priority 2
disk            xy0 at xyc0 drive 0
disk            xy1 at xyc0 drive 1
device          ropc0 at mb0 csr 0xee0800
device          sky0 at mb0 csr 0x2000 priority 2
device          zs0 at mb0 csr 0xeec800 flags 3 priority 2 # cpu ports
device          mti0 at mb0 csr 0x620 flags 0xff priority 4
device          ec0 at mb0 csr 0xe0000 priority 3
device          ec1 at mb0 csr 0xe2000 priority 3
device          ie0 at mb0 csr 0x88000 priority 3
controller      tm0 at mb0 csr all virt 0xeb00a0 priority 3
tape            mt0 at tm0 drive 0 flags 1
device          cgone0 at mb0 csr 0xe8000 priority 3
device          bwone0 at mb0 csr 0xc0000 priority 3
device          pi0 at mb0 csr 0xee2000
device          tod0 at mb0 csr 0xee1000
```

```
#
# Model 160 with up to 2 Xylogics controllers and 1/2" tape
# plus 1 or 2 SCSI disks and 1 SCSI tape
#
machine         sun
cpu             "SUN2"
ident           "XYMT160"
timezone        8 dst
maxusers        4
options         INET
options         RPC
options         NFS

config          vmunix    root on xy

pseudo-device   rpc
pseudo-device   nfs
pseudo-device   pty
pseudo-device   inet
pseudo-device   ether
pseudo-device   loop
pseudo-device   nd
pseudo-device   win128
pseudo-device   dtop4
pseudo-device   ms3
pseudo-device   kb3
pseudo-device   ingres
controller      mb0 at nexus ?
controller      xyc0 at mb0 csr all virt 0xebee40 priority 2 vector xyintr 72
controller      xyc1 at mb0 csr all virt 0xebee48 priority 2 vector xyintr 73
disk            xy0 at xyc0 drive 0
disk            xy1 at xyc0 drive 1
disk            xy2 at xyc1 drive 0
disk            xy3 at xyc1 drive 1
controller      sc0 at mb0 csr vme busmem 0x200000 priority 2 vector scintr 64
disk            sd0 at sc0 drive 0 flags 0
disk            sd1 at sc0 drive 1 flags 0
tape            st0 at sc0 drive 32 flags 1
device          sky0 at mb0 csr vme busio 0x8000 priority 2 vector skyintr 176
device          zs0 at mb0 csr all virt 0xeec800 flags 3 priority 3 # cpu
device          zs1 at mb0 csr all virt 0xeec000 flags 0x103 priority 3 # video
device          mti0 at mb0 csr all virt 0xeb0620 flags 0xffff priority 4 vector mtiintr 136
device          ie0 at mb0 csr vme virt 0xee3000 priority 3
controller      tm0 at mb0 csr all virt 0xeb00a0 priority 3 vector tmintr 96
controller      tm1 at mb0 csr all virt 0xeb00a2 priority 3 vector tmintr 97
```

```
tape            mt0 at tm0 drive 0 flags 1
tape            mt1 at tm1 drive 0 flags 1
controller      xtc0 at mb0 csr all virt 0xebee60 priority 3 vector xtintr 100
controller      xtc1 at mb0 csr all virt 0xebee68 priority 3 vector xtintr 101
tape            xt0 at xtc0 drive 0 flags 1
tape            xt1 at xtc1 drive 0 flags 2
device          cgtwo0 at mb0 csr vme busmem 0x400000 priority 3
device          bwtwo0 at mb0 csr vme obio 0x0 priority 4
device          tod0 at mb0 csr vme busmem 0x200800
```

Index