UNIX 4.3BSD

System Administrator Guide

ISI Release 4.1 (4.3BSD)

Integrated Solutions
1140 Ringwood Court
San Jose, CA 95131
(408) 943-1902

490197 Rev. C

May 1988

READ THIS FIRST

This manual contains instructions on the installation and operation of 4.3BSD UNIX on Integrated Solutions, Inc. (ISI) computer systems and CPU boards. These systems and boards use the 68010 or 68020 processors and are available in two families:

- The Optimum series, based on the LSI-11 bus (Q-bus).
- The Optimum V series, based on the VMEbus. The Optimum V series includes Diskless and Cluster Nodes.

Who Should Read This Guide?

You should read this manual if you are responsible for the installation or maintenance of your system. Casual users (people who log in with user accounts to do normal work) do not ordinarily need this manual.

This manual assumes familiarity with UNIX; it is not intended for first-time UNIX users.

What Does This Guide Cover?

This manual explains how to get started with UNIX, including

- Installing UNIX from tape (if necessary)
- Starting UNIX (''booting'')
- Optimizing UNIX for your applications
- Adding terminals and users
- Setting up a network
- Troubleshooting

A separate publication, *Field Upgrade Procedure, UNIX 4.2BSD to 4.3BSD*, describes the upgrade procedure for existing 4.2BSD systems.

This manual contains eight sections and four appendices:

Section 1 provides introductory material on UNIX and system administration.

Section 2 tells how to turn on power for the system and log in.

Section 3 describes setting up the system for normal operation.

Section 4 tells how to alter the system for your configuration.

Section 5 describes system network connections.

Section 6 describes setting up and operating a cluster network.

Section 7 describes setting up and operating a diskless network.

Section 8 describes day-to-day maintenance and system operation procedures.

Appendix A tells how to install UNIX from a release tape.

Appendix B describes some hardware diagnostic programs.

Appendix C is a configuration worksheet, where you can make notes on your system.

Appendix D provides space to insert any release notes for your system.

Some Naming Conventions for This Guide

This is a brief glossary of references you will see in this manual:

ISI is Integrated Solutions, Inc.

Optimum System is the Optimum series of ISI systems based on the LSI-11 bus (Q-bus).

Optimum V System is the Optimum V series of ISI systems based on the VMEbus.

Optimum V WorkStation is the special graphics display system; largely the same as the Optimum V System, but with a graphics subsystem (processor board, display memory board, high-resolution monitor, keyboard, and mouse).

A **node** is any system in a local network with its own CPU board. For example, in a cluster network, the server system and each cluster are nodes.

A **server node** is the central node for a network of other nodes without disks of their own. For diskless networks, the server node can be an Optimum System, an Optimum V System, or an Optimum V WorkStation, and must have an Ethernet controller. For cluster networks, the server node must be an Optimum V16 or V24 System/WorkStation.

A **Cluster Node** is a set of boards that occupy a portion of the card cage of a V16 or V24 system. The cluster node may be a graphics node (WorkStation), or it may be a non-graphics node (Compute Node) with two serial lines for terminal connections (console and Port 0).

A **Cluster Compute Node** is a cluster node without a graphics subsystem.

A **Cluster WorkStation** is a cluster node with a graphics subsystem.

A **Diskless Node** is a system with no disk of its own, connecting through an Ethernet network to a server node. The diskless node may be a graphics node (WorkStation), or it may be a non-graphics node (Compute Node).

A **Diskless Compute Node** is a diskless node without a graphics subsystem.

A **Diskless WorkStation** is a diskless node with a graphics subsystem.

A **hostname** is the official system name for any node, generally assigned with **admin** (8).

The word ''system'' applies generically to any of the following:

- An Optimum System
- An Optimum V System
- An Optimum V WorkStation (stand-alone)
- An Optimum V Cluster Compute Node
- An Optimum V Cluster WorkStation (a graphics node)
- An Optimum V Diskless Compute Node
- An Optimum V Diskless WorkStation (a graphics node)
- The server node of a cluster or diskless network (a server node is an Optimum System, Optimum V System, or Optimum V WorkStation)

Throughout this manual, you will see these uses of **boldface** and *italics:*

Boldface

- Section headings.
- References to UNIX commands (**tar** (1) refers to the command **tar**; the "(1)" shows that you can find a description of **tar** in "Section 1: Commands" of the UNIX 4.3BSD *User's Reference Manual* (URM)).
- Commands that you type to UNIX exactly as printed (for example, "Enter **fsck** and press RETURN").
- Messages that UNIX prints on your screen (for example, **login:**).
- User account names (for example, the **root** login account, the group **operator**).

Italics

- Names of manual (for example, the *Optimum V24 System Installation Manual*).
- Names of files (for example, */etc/passwd*).
- Parts of commands that you must replace with real values (for example, *dev* refers to a hardware device; you must type the actual name of the device where you see *dev*).

When you see **boldface** and *italics* mixed, as in this example:

Enter this command:

dev(0,0)**vmunix**

this means that you should replace *dev* with the name of a real UNIX device, and type the rest as shown. For example:

sm(0,0)vmunix

Related Documents

You should have these documents:

- The UNIX 4.3BSD Reference Set, seven volumes:
  - — *UNIX User's Reference Manual* (URM)
  - — *UNIX User's Supplementary Documents* (USD)
  - — *UNIX Programmer's Reference Manual* (PRM)
  - — *UNIX Programmer's Supplementary Documents*, Volume 1 (PS1)
  - — *UNIX Programmer's Supplementary Documents*, Volume 2 (PS2)
  - — *UNIX System Manager's Manual* (SMM)
  - — *User Contributed Software* (UCS)
- VI Reference Card for the vi (1) editor

If you are upgrading an existing UNIX 4.2BSD system, you should also have this document:

- *Field Upgrade Procedure, UNIX 4.2BSD to 4.3BSD*

These manuals document the Optimum V series:

- *Optimum V8 System/WorkStation Installation Manual*
- *Optimum V13/V16 Site Preparation and Installation Manual*
- *Optimum V24 System/WorkStation Installation Manual*
- *VME-68K10 Hardware Reference Manual*
- *VME-68K20 Hardware Reference Manual*
- *VME-HSMEM Hardware Reference Manual*
- *VME-HSMEM-8/4 Hardware Reference Manual*
- *VME-SCSI/U Host Adapter Hardware Reference Manual*
- *VME-QIC2/X Hardware Reference Manual*
- *VME-TC50/X Hardware Reference Manual*
- *VME-ICP16/8X Hardware Reference Manual*
- *VME Graphics Subsystem Hardware Reference Manual*
- *VME Color Graphics Subsystem Hardware Reference Manual*

There are many good introductory UNIX reference books. One such book, *Introducing the UNIX System* (McGilton and Morgan, 1983: McGraw-Hill), includes a section on system administration.

Copies of the above-mentioned manuals and UNIX book can be ordered through your local ISI sales office.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

SECTION 1: INTRODUCTION

If you have not yet done so, please read "Read This First" at the beginning of this guide.

This section describes the functions of a System Administrator and provides background knowledge on the hardware and software that you will use.

This section contains the following subsections:

1.1 What Does a System Administrator Do?

1.2 Introduction to Hardware

1.3 Introduction to Software

1.1 What Does a System Administrator Do?

The UNIX operating system requires attention if it is to run well. No two UNIX installations are exactly alike; your system's UNIX configuration is different from others.

A System Administrator has four principal duties:

- To install UNIX on a computer system.
- To keep UNIX running on that system, adjusting UNIX as necessary.
- To record changes to (and/or problems with) the system, keeping a log of activities.
- To back up the system (save copies on tape) on a regular basis.

These duties require familiarity with UNIX and with the hardware and software components of your computer system. "Related Documents" in "Read This First" suggests reading material concerning your Optimum or Optimum V system. It also suggests some basic texts on UNIX geared for general knowledge.

To function as a System Administrator, you must be able to perform these tasks:

- Log in to the system as superuser (see Section 1.3, "Introduction to Software")
- Move easily through the UNIX file system
- Edit files with a UNIX editor (vi (1)[†], ex (1), ed (1))
- Follow the procedures throughout this guide
- Maintain a system log (notebook)

1.2 Introduction to Hardware

As a System Administrator, you do not have to handle much hardware directly, but you do need to know how UNIX deals with the hardware. Figure 1-1 shows a block diagram of an Optimum or Optimum V system.

[†] References of the form name (X) refer to a subsection named name in Section X of the UNIX 4.3BSD Reference Set. Section 1, 6, and 7 are in the URM volume, Sections 2, 3, 4, and 5 are in the PRM volume, and Section 8 is in the SMM volume. See "Related Documents" in "Read This First".

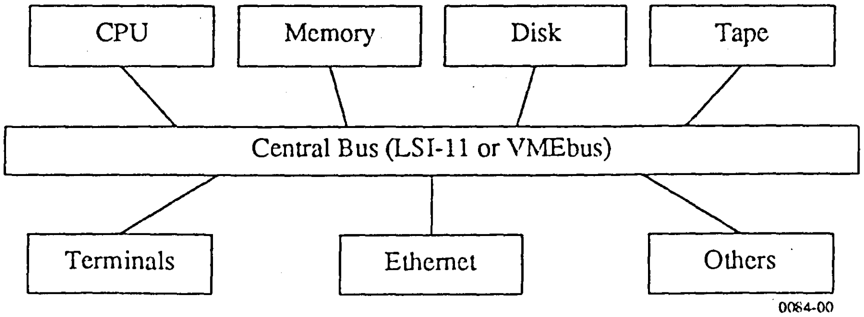


Figure 1-1. Block Diagram of an Optimum or Optimum V System

Check your Installation Manual if you are interested in the physical connections. For now, consider these components in the way that UNIX sees them:

Central Bus (LSI-11 or VMEbus): The means by which other system components communicate with each other. The bus is a set of channels through which signals pass from board to board. The "backplane" is the rigid piece of hardware to which the boards physically attach; the "bus" is the format and design of signal arrangements on the backplane. ISI uses two different buses for two separate product lines: the Optimum series of systems, based on the DEC LSI-11 bus ("Q-bus"), and the Optimum V series, based on the VMEbus.

CPU (central processing unit): The "brain" of the system. The CPU executes UNIX commands. Programmable Read-Only Memory (PROM) chips on the CPU contain a rudimentary monitor and loading facility for UNIX. You use these PROMs, with a limited set of commands, to bring up the system and to execute the larger body of UNIX from disk or tape.

Memory: Volatile, fast-access storage for programs and data.

Disk: A magnetic storage medium for large amounts of data. UNIX refers to each disk as a separate device, signified by two letters and a number (for example, "sm1"). The letters describe the model and type of controller board that supports the disk; the number distinguishes one disk and/or controller from another in the system. Each disk, in turn, holds seven or eight logical storage areas called "partitions," labeled a-g[h]. Do not confuse partitions with "cylinders;" cylinders are physical parts of the disk, while partitions are larger areas that include multiple cylinders.

Tape: A removable magnetic storage medium for long-term storage or for transferring data between systems. UNIX refers to each tape subsystem (the tape drive and its controller) as a separate device, signified by two letters and a number, as with disks (for example, "ts0").

Terminals: A device humans use to communicate with UNIX. To UNIX, a terminal is simply a special kind of file: UNIX writes to the file (things appear on your screen) and reads from the file (you type something on your keyboard). UNIX refers to these special files as "ttys" (for example, *tty01*, *tty02*, and so forth). You must tell UNIX what kind of terminal you are using, what settings (baud rate and parity, for example) you have made on the terminal, and where you are plugging it into the system.

Ethernet and other devices: Each is a special case. The Ethernet board allows networking to other systems (see Section 5.1). Other boards support terminal connections or line printers (see Section 3.4); the Optimum V WorkStation uses special boards for the graphics monitor.

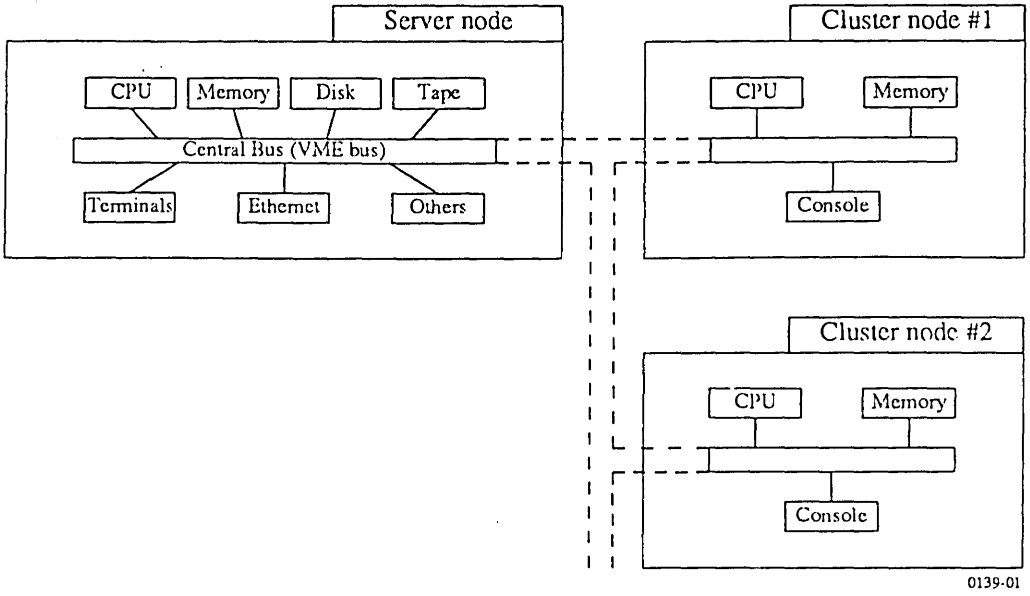


Figure 1-2. Block Diagram of a Cluster Node Configuration

Figure 1-2 shows a block diagram of a Cluster Node configuration. A cluster configuration consists of a server node that provides central disk storage and cluster nodes. The cluster nodes share a common system package with their associated server node.

Cluster nodes and server nodes are physically installed in the same card cage. By adding additional boards to an Optimum V16 or V24 System, you can create independent computer system-, each with its own memory, CPU, and peripherals, yet sharing common disk storage.

The server and cluster nodes communicate through the VMEbus. A UNIX device "driver" supports this communication and is named "VB". When you install a cluster node system, you must include this driver in the system configuration information.

See "Section 6: Setting Up Cluster Nodes" for the specific UNIX features that support cluster nodes.

Cluster Nodes are similar to Diskless Nodes. The difference is that cluster nodes are contained within a single card cage and communicate via a "shared memory" (VME-CMEM), whereas diskless nodes are independent physical units that communicate via an Ethernet network. Figure 1-3 shows a block diagram of a Diskless Node configuration.

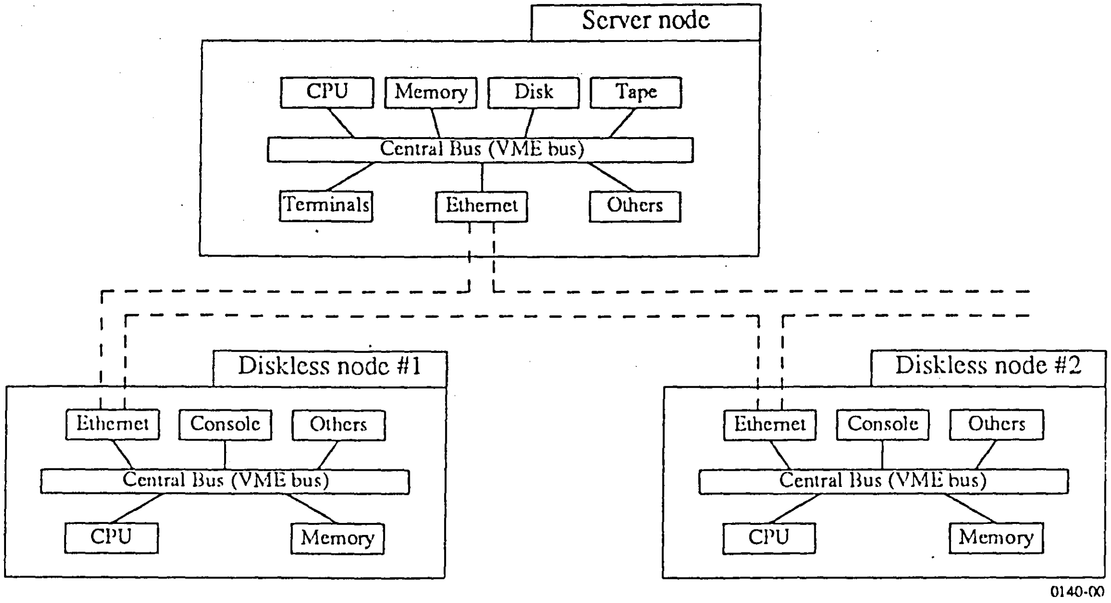


Figure 1-3. Block Diagram of a Diskless Node Configuration

The diskless node also uses a server node for disk storage. System administration tasks for cluster nodes and diskless nodes are almost identical.

Any system that uses an ISI Ethernet board (VME-ECX) or Excelan Ethernet board (VME-EC or ISEC) can connect to a network. Any Optimum V (VMEbus-based) system that is on the network and has a disk can act as a server node for diskless nodes. Several server nodes can co-exist on a network; each server node can support one or more diskless nodes.

See "Section 7: Setting Up Diskless Nodes" for the specific UNIX features that support diskless nodes.

1.3 Introduction to Software

You should be familiar with the UNIX hierarchical file structure. A file's full name pinpoints its location in that structure. For example, the filename */usr/lib/aliases* defines a file *''aliases''* in a directory *''lib,''* which in turn is an entry in the directory *''usr,''* and *''usr''* stems from the ''root'' directory (signified by ''/'').

hier (7) shows the skeletal structure of the UNIX file system. Cluster and diskless networks rely on the Transparent Remote File System (TRFS) to share a single file system among several nodes on the network. TRFS is an ISI utility that allows access to files on other nodes through a user-transparent set of network protocols.

If you are confused by directories and filenames, see "Read This First" for introductory documents on UNIX.

An important file in the root directory, */vmunix*, contains the ''kernel'' for the UNIX system. The kernel determines device configurations and other important system parameters. Section 4.2, "Configuring the Kernel", provides instructions on tailoring the kernel to your particular system.

The UNIX operating system, as shipped by ISI, consists of five major parts:

- A ''miniroot'' file system, used only when loading UNIX from a release tape.
- The root (''/'') file system.
- The */usr* file system.
- Optional graphics files.
- Stand-alone diagnostic programs, on a separate tape.

To install or make changes to these and other system files, and to perform many of the system administrator functions such as adding new users to the system, you must have unlimited access to the entire system. You do this by logging in to the system as **root** and giving the appropriate password. You set the root password as part of the first-time installation procedure. (See "Section 2: Starting UNIX From Disk" or "Appendix A: Starting UNIX From Tape".)

When you log in as **root**, you are now a superuser. This means that you have undeniable access to everything on the system. If you make an error (such as using the **rm** (1) command while you are in the wrong directory) you can destroy the file system, in which case you will have to re-install the UNIX software from the release tape. While superuser, you must be careful to protect system files and/or files containing work done by others. There are three ways to protect both yourself and others:

- Follow procedures in this guide *exactly.*
- If you have any doubt, copy files onto tape before making changes. Use the **tar**(1) or **dump**(8) commands to copy files onto tape where they will be safe.
- Keep a complete log of everything you do in a notebook near the console.

SECTION 2: STARTING UNIX FROM DISK

This section describes system startup and how to log in to UNIX. Your Installation Manual contains some of this information; this section offers more detailed instructions on troubleshooting.

When you first use an Optimum or Optimum V system, you do not need to install UNIX. The disks shipped from Integrated Solutions contain everything necessary for UNIX to autoboot (begin operation automatically).

If you do not have disks shipped from Integrated Solutions, or if you want to install UNIX from a release tape for any other reason, turn to "Appendix A: Loading UNIX From Tape."

Use the procedures in this section to boot server nodes for cluster or diskless networks. See "Section 6: Setting Up Cluster Nodes" or "Section 7: Setting Up Diskless Nodes" to boot other nodes on the network.

This section contains the following subsections:

2.1 What Is "Bootstrapping?"

"Bootstrapping," usually abbreviated to "booting," means loading and starting the operating system (or other program). UNIX does as much of the work as possible, essentially pulling itself up by its own bootstraps. This feature, booting without the help of humans, is "autobooting."

Your role in autobooting UNIX should be simple:

- Turn on power
- Give the current date and time
- Log in as superuser

2.2 Do You Have Everything You Need?

You must have a fully installed system, as described in the appropriate Installation Manual. For cluster or diskless networks you must have a fully installed and configured server node.

The boot procedure in Section 2.3 assumes that you have a disk ready for UNIX autoboot. If the disk does not contain the proper files, UNIX will not autoboot. In this case, refer to Section 2.6, "What to Do if the Autoboot Fails."

You must know the name of your system's boot disk; you will need to use this name in commands you enter. Use Table 2-1 to find the disk's name. Enter this disk name on the configuration worksheet in Appendix C.

Table 2-1. UNIX Disk Names

Disk Name	Device	Controller Name
Optimum V Systems and WorkStations		
VDS106 VED106	5 ¼-inch disks (SD-style) 106 Mbyte formatted	sd
VDS269 VED269	8-inch disks (SMD-style), 269 Mbyte formatted	sm
VDS418 VED418	9-inch disks (SMD-style), 418 Mbyte formatted	sm
VDS494 VED494	8-inch disks (SMD-style), 494 Mbyte formatted	sm
Optimum Systems		
DS36 ED36	5 ¼-inch disks, 36 Mbyte formatted	cl
DS67 ED67	5 ¼-inch disks, 67 Mbyte formatted	cl
DS85 ED85	5 ¼-inch disks, 85 Mbyte unformatted	cl
DS140 ED140	5 ¼-inch disks, 140 Mbyte unformatted	cl
DS330 ED330	SMD-style disks, 330 Mbyte formatted	hp
DS418 ED418	SMD-style disks, 418 Mbyte formatted	hp
DS474 ED474	SMD-style disks, 474 Mbyte unformatted	hp

You should also have these materials on hand:

- The appropriate Installation Manual (for example, the *Optimum V8 System/WorkStation Installation Manual*)
- The UNIX 4.3BSD Reference Set, seven volumes

2.3 Booting the System

If you encounter trouble during this boot procedure, refer to Section 2.6, "What to Do if the Autoboot Fails."

For details of what happens in the following procedure, see Section 2.5, "How Does the Autoboot Work?"

The following procedure, the UNIX Autoboot Procedure, describes the steps you take to autoboot UNIX.

UNIX Autoboot Procedure

	ACTION	WHAT IT DOES
Step 1.	Turn ON power for the terminal (or WorkStation monitor) connected as your system console.	Prepares the terminal to print boot messages.
Step 2.	Turn ON system power. See the appropriate Installation Manual for detailed power-up instructions. Check that the DC ok indicator lights up.	Provides power to the system and begins the autoboot. The console should show the message **Initialize all memory . .** (Optimum V WorkStations do not display this message.) After a short wait (about ten seconds), this message should appear on the console: **Integrated Solutions** VMEBUS*(QBUS)* 680*XX* *prom-level* boot:*date* : The colon (:) is a prompt from the UNIX PROMs (see Section 1.2).
Step 3.	If you have booted UNIX previously, and you know the root password, go on to *Step 4* of this procedure. If this is the first time for booting UNIX on this machine, you must boot UNIX single-user and assign a password to the root (''superuser'') account. Enter *dev*(0,0)**vmunix** where *dev* is the 2-letter name for your boot disk, from Table 2-1. Follow the further instructions in the column at right.	Boots UNIX in single-user mode. The console now displays information about available devices and file systems, beginning with Integrated Solutions UNIX Release 4.0 #*n* (root@*system-name*) *date* **real mem=** *(physical memory)* **avail mem=** *(memory available for user programs)* **Using** *xx* **buffers containing . . .** and ending with a single-user UNIX prompt (#). To set the root password, use the command **passwd root** and follow the instructions of the command prompts. After setting the root password, enter the command **reboot** and resume this procedure with the next step.

(continued on next page)

UNIX Autoboot Procedure *(continued)*

	ACTION	WHAT IT DOES
Step 4.	You have assigned, or already know, the root password for this system. The screen shows the message **Integrated Solutions** VMEBUS*(QBUS)* 680*XX* *prom-level* boot:*date* : Enter the character "@" (SHIFT-2 on most terminals) and press RETURN.	Tells the PROMs to continue the autoboot. Entering any other character interrupts the autoboot. After 15 seconds, the autoboot continues anyway, with this message: : **AUTOBOOT:** *dev*(0,0)**vmunix** where *dev* is the name of the system's boot disk (see Table 2-1). The console now displays information about available devices and file systems, beginning with **Integrated Solutions UNIX Release 4.0 #*n*** (root@*system-name*) *date* **real mem=** *(physical memory)* **avail mem=** *(memory available for user programs)* **Using *xx* buffers containing . . .** and ending with **You have 60 second(s) to enter date/time:**
Step 5.	Enter the current date and time, in this format: *yymmddhhmm* For example, to enter April 15, 1987, at 2:30 p.m., enter 8704151430 and press RETURN.	Tells UNIX what date and time it is. See date(1) for details on the entry format. After one minute without an entry, UNIX assumes the last known date and time, and continues the autoboot. After displaying several more messages and checking file system consistency, the console shows this prompt: Integrated Solutions UNIX *(system-name) (terminal line) date* login: This is the UNIX login prompt. UNIX is now running in multi-user mode.
Step 6.	Log in as the superuser by entering root as your login name. You assigned the root password while in single-user mode (see *Step 3* of this procedure).	Logs in to UNIX, giving access to a working C shell (UNIX command processor). You can now issue UNIX commands to the shell prompt. This is the UNIX multiuser prompt: *system-name#* where *system-name* is originally "UNKNOWN". You will assign the system name in a later procedure.

End of procedure

After a successful boot, you are ready to set up your system for normal multiuser UNIX operation. Turn to "Section 3: Setting Up Your System." If you are changing your hardware configuration, and need to adjust UNIX for the new hardware, see "Section 4: Configuring Your System."

2.4 Shutting Down the System

CAUTION

Failure to shut down the system in an orderly fashion may result in lost data. In particular, if you simply turn off power without an orderly shutdown you will possibly lose data.

If you are running multiuser UNIX, use the command **shutdown** (8) in the following procedure. If you are running single-user UNIX (the console shows the single-user prompt (#)), skip to *Step 2* of the following procedure.

If your machine is a server for cluster or diskless network nodes, shut down the individual cluster or diskless nodes with the procedures in "Section 6: Setting Up Cluster Nodes" or in "Section 7: Setting Up Diskless Nodes." After shutting down any nodes that use this server, use the following UNIX Shutdown Procedure for the server node.

Use the UNIX Shutdown Procedure outlined below to shut down the system.

NOTE

You must be logged in as superuser (root) at the system console before you can run this procedure.

UNIX Shutdown Procedure

	ACTION	WHAT IT DOES
Step 1.	Enter this command at the system console: shutdown +*n* where *n* is a time in minutes.	Notifies users of shutdown, waits for the specified time, and then kills all processes. When shutdown is complete, the console shows the single-user prompt (#). CAUTION If you turn off your system at this point, you risk losing data. You must finish all steps of this procedure.
Step 2.	Enter these commands: cd / sync;sync umount -a reboot	Unmounts the file systems and updates the superblock. This protects the disks from accidental erasure when power goes off. Wait until the PROM prompt (:) appears.
Step 3.	Turn OFF system power. See the appropriate Installation Manual for specific power-down instructions.	Removes power from the system. The system is now off.
Step 4.	If the system will be off for some time, turn OFF power to connected devices, such as the system console.	Conserves power.
	End of procedure	

2.5 How Does the Autoboot Work?

Suggested reading:

- ''Fsck—The UNIX File System Check Program,'' *UNIX 4.3BSD System Manager's Manual* (SMM:5)
- fsck (8)
- init (8)
- rc (8)
- reboot (8)

UNIX will try to autoboot under several conditions:

- Turning on system power (see Section 2.3, "Booting the System")
- Recovering from a system crash
- Receiving a reboot (8) command from the superuser
- Receiving a fastboot (8) command from the superuser

All of these, in one way or another, involve the rc (8) command file, a set of instructions that UNIX uses in booting. As superuser, you can control the speed and caution of the boot process, ranging from a slow and careful boot to a fast boot that ignores possible damage to data on the disks.

Figure 2-1 shows a sample of a normal boot. This example uses an Optimum V8 System with Release 4.0 software and a VDS106 disk subsystem.

```
Initialize all memory . .

(Integrated Solutions logo appears here)

Integrated Solutions
VMEBUS  68020  4.4/1  boot:Wed Apr 22 19:15:44 PDT 1987
: AUTOBOOT: sd(0,0)vmunix

Integrated Solutions UNIX Release 4.0 #1 (root@UNKNOWN) Wed Apr 1 16:58:26 PST 1987
real      memory = 2.25M
available memory = 1.11M
using 90 buffers containing 0.23M of memory

  SD0  at address 0xffffe0/077777740   vector 0x78/0170  level 6
     sd0   at SD0  slave 0     78.36M  (25 x 7 x 917)
                  a:15884          b:33400          c:abdefg:160475  d:15884
                  e:55936          f:38850          g:def:110775
    TS0  at address 0xfff550/077772520   vector 0x94/0224
         ts0    at TS0  slave 0         1/4" QIC2 tape drive

    EX0  at address 0xff0000/077600000   vector 0x40/0100
         ex0    at EX0  slave 0         (08:00:14:20:02:72) HV 0.0
  root on sd0a, dump on sd0b, args on sd0b, swap on sd0b

16.67MHz MC68020 CPU
CTC system clock
sd0: 11 remapped sectors
***  WARNING:  Should run interleaved swap with >= 2Mb  ***
Automatic boot in progress...

Wed Apr 22 19:19:02 PDT 1987
You have 60 second(s) to enter date/time: 8704230935
Thu Apr 23 09:35:00 PDT 1987
/dev/sd0a: 261 files, 4116 used, 3305 free (73 frags, 404 blocks, 0.3% fragmentation)
/dev/sd0g: 3113 files, 30063 used, 12544 free (132 frags, 3103 blocks, 0.3% fragmentation)
Thu Apr 23 09:36:33 PDT 1987
check quotas: done.
starting system logger.
starting local daemons: bootd routed named timed sendmail.
preserving editor files
clearing /tmp
standard daemons: update cron accounting.
starting network daemons: inetd rwhod printer.
Thu Apr 23 09:36:58 PDT 1987

Integrated Solutions UNIX  (UNKNOWN)  (console)  Thu Apr 23 09:37:03 PDT 1987

login:
```

Figure 2-1. Sample of a Normal Autoboot

Autobooting follows the steps in Table 2-2.

Table 2-2. How the Autoboot Works

WHAT UNIX DOES	SEE . . .
The PROMs initialize all memory; this clears the computer memory in preparation for loading a program (for example, UNIX) from disk or tape.	
The PROMs load UNIX into memory and begin execution. This executes the kernel (*/vmunix*) and begins a consistency check of the file systems. The console shows the known devices and their current configurations. In Figure 2-1, UNIX finds one disk sd0 and one tape drive ts0. sd0 has seven partitions, lettered *a*-*g*. Note that partition *c* actually refers to the entire disk, while partition *g* refers to partitions *d*, *e*, and *f* combined. In practice, you ordinarily deal only with partitions *a*, *b*, and *g*.	reboot (8)
UNIX prints some messages about system hardware. In Figure 2-1, these messages • Identify the CPU • Show parity status • Identify the system clock • Warn you to set up two swap devices (ignore this message)	
UNIX executes the */etc/rc* command file. rc begins by printing the message Automatic boot in progress... on the console.	rc (8)
rc prompts for the current date and time. After one minute with no entry, UNIX assumes the last known date and time. At this prompt, you can safely exit the rc script by entering two ^C's (CTRL-C twice).	date (1)

(continued on next page)

Table 2-2. How the Autoboot Works *(continued)*

WHAT UNIX DOES	SEE . . .
rc runs the file system check procedure, */etc/fsck -p*. This prints diagnostic messages on the console, then gives one of five results: • Normal completion; rc prints the date and time and continues with the autoboot. • Restart; rc decides to reboot again from the beginning. This happens when fsck fixed problems in the root (/) file system. • **Automatic reboot failed... help!** fsck encountered bad disk areas, and needs your help in proceeding. Section 2.6, "What to Do if the Autoboot Fails," covers this case. • **Reboot interrupted** means that something caused fsck to stop before finishing. This could happen if you sent an interrupt signal (^C) from the console. You should restart the boot with **reboot** (8), as described in Section 2.6. • **Unknown error in reboot** means just what it says. You should run fsck manually, as described in Section 2.6.	fsck (8)
rc calls the commands **swapon** and **mount** to set up the disks and file systems for availability. Both of these commands rely on the file */etc/fstab* for information on disk allocation. These calls should not generate any messages on the console.	swapon (8) mount (8) fstab (5)
rc runs quotacheck (8) to see that the current file systems have not overrun their allowed sizes (see Section 8.6, "Accounting"). This generates a console message when the quota check is finished. rc then runs quotaon (8) to turn on quotas for all file systems in */etc/fstab* that show quotas. quotaon should not print anything on the console unless you have set quotas in some file system; the ISI UNIX release has quotas turned off.	quotacheck (8) quotaon (8)
rc executes the command file */etc/rc.local*. This file contains commands that you use to customize UNIX to your site. The *rc.local* file on your release tape and disk performs these functions: • sets your system hostname to UNKNOWN. See Section 3.3, "Initializing Your System," to change this. • builds a ''message of the day'' in */etc/motd* to print for users when they log in. You can edit */etc/motd* to say whatever you want; *rc.local* simply ensures that the first line of */etc/motd* describes the current UNIX version. • starts up a set of daemons (processes that run in the background).	rc (8) hostname (1) daemons: syslogd (8) routed (8C) named (8C) timed (8) sendmail (8) graphics: fontd (8)

(continued on next page)

Table 2-2. How the Autoboot Works *(continued)*

WHAT UNIX DOES	SEE . . .
rc clears the temporary file area, */tmp*, after saving any */tmp* files dealing with editors. The editors (**vi**, **ex**, and **ed**) use */tmp* to store scratch copies of files being edited, and can use these scratch files to reconstruct lost files if the editor or the system crashes.	**ex** (1)
rc starts up a set of standard daemons. One of these daemons, **inetd** (8), starts up additional daemons. The file */etc/inetd.conf* defines these additional network daemons.	**update** (8) **cron** (8) **accton** (8) **rwhod** (8) **lpd** (8) **inetd** (8)
rc finishes, and UNIX prints the current time. UNIX now prompts for login.	

2.6 What to Do if the Autoboot Fails

Read this section if the procedure in Section 2.3, "Booting the System" did not reach a login prompt. For difficulties after the login prompt, after you have logged in as **root**, see "Section 3: Setting Up Your System."

If the boot failed, find the appropriate symptom in Table 2-3.

Some difficulties in booting a graphics workstation may be related to graphics hardware. See the appropriate *Installation Manual* for your workstation for instructions on connecting a regular ASCII terminal to the workstation for diagnostic purposes. You can then boot UNIX using the ASCII terminal.

Table 2-3. Symptoms of Boot Failure

SYMPTOM	SOLUTION
The terminal does not respond to any typed commands. Pressing ^C (holding down the CTRL key and pressing "c") gets no response.	Check the Run and DMA lights on the front panel of your system. If these lights are flickering, the system is in the middle of some process. Wait for it to finish. (The file system check can take several minutes to complete.) If it does not finish in several minutes, press the RESET button; this will restart the boot process. CAUTION Pressing RESET while the DMA light is flickering may result in data loss.
	You are using the wrong RS232C cable for this terminal. The console port, the "Port 0" line, and other serial ports use "straight through" cables on non-graphic systems. You might have a "twisted" or "null-modem" cable, which would cause the terminal not to work. Expansion ports (DZ or DH breakouts) on Optimum (Q-bus) systems require "twisted" cables. Check the original *Installation Manual* sent with your system if you have any question as to what type of connector or cable is used. See Section 3.4, "Setting Up Terminals, Modems, and Printers," for a description of cables. Graphic systems use cables supplied with the graphic terminal.
	The terminal is not working properly. Check the baud rate (see the *Installation Manual* for terminal baud rates). Turn the terminal off, wait a few seconds, then turn the terminal on again. Press RETURN. If this does not work, and you have another terminal, swap terminals or swap lines to the other terminal. If you are booting a workstation, try using an ASCII terminal as described in your *Installation Manual.* Finally, check the instruction manual for the terminal. Many terminals have key sequences for "keyboard locks" and other crippling commands.
	A physical connection is loose somewhere. In this order, check the following: • Connections between the terminal and the system. • Boards are correctly seated in the card cage. • Connections on all cables inside the system.

(continued on next page)

Table 2-3. Symptoms of Boot Failure *(continued)*

SYMPTOM	SOLUTION
The boot stopped before reaching a login prompt, and the console now shows the PROM prompt (:).	Enter the character "@" (SHIFT-2 on most terminals) and press RETURN. This may restart the autoboot. If "@" does not work, restart the autoboot manually by entering *dev*(0,0)**vmunix** where *dev* is the name of the boot disk (see Table 2-1). **vmunix** is the file */vmunix*, the default UNIX kernel. You can use any other file name, if that file is a bootable kernel.
	If *dev*(0,0)**vmunix** returns a message like **unknown device, legal devices are:** *(list of device names)* check that you are using the proper name for the boot device.
	The message **file "vmunix" not found** means that the disk does not contain the file necessary for autobooting. You must boot off an alternate kernel. You might try *dev*(0,0)**vmunix.***C* where *C* would be Q10, V10, or V20, depending on your system. Other possible kernel names are *dev*(0,0)**vmunix.new** *dev*(0,0)**vmunix.old** *dev*(0,0)**vmunix.orig** If you cannot find a bootable kernel, install UNIX from a release tape. Turn to "Appendix A: Loading UNIX From Tape."
The boot stopped before reaching a login prompt, and the console now shows the single-user UNIX prompt (#).	Enter **reboot** and press RETURN. This should restart the autoboot. **reboot** (8) syncs the disk (updates the disk with any recent changes) and brings the system back to the PROMs for autoboot. When you see the PROM prompt (:), follow the boot procedure in Section 2.3.

(continued on next page)

Table 2-3. Symptoms of Boot Failure *(continued)*

SYMPTOM	SOLUTION
The message **WARNING: should run interleaved swap with >= 2 Mb** appears on the console, followed by the single-user UNIX prompt (#).	Ignore the warning message. Treat this symptom as the symptom above, using the **reboot** (8) command. If you have more than one disk, you should consider setting up a swap area on the second disk. See Section 4.3.2, "Optimizing Disk Efficiency."
A message similar to **parity error or power fail** appears on the console, and the boot stops.	This usually means that a power surge or drop interrupted the boot. Do nothing; UNIX will reboot automatically. If UNIX does not reboot automatically, use *dev*(0,0)**vmunix** at the ":" prompt, or **reboot** at the "#" prompt.
The message **Reboot interrupted** appears on the console, followed by the single-user UNIX prompt (#).	This means that something caused fsck(8), the automatic file system check program, to stop before finishing. This could happen if UNIX receives an interrupt signal (^C) from the console. Restart the autoboot with **reboot** (8).
The message **Unknown error in reboot** appears on the console, followed by the single-user UNIX prompt (#).	fsck (8) encountered something it could not deal with and could not even identify. Run fsck manually; see the next symptom.
The message **Automatic reboot failed... help!** appears on the console, followed by the single-user UNIX prompt (#).	fsck (8) encountered disk errors that it could not correct automatically. Run fsck manually. CAUTION You are now on dangerous ground; you could destroy data on the disk. Read the rest of this section carefully before doing anything else. At the single-user UNIX prompt (#) enter fsck. fsck now asks for decisions on cleaning up the disk. These decisions always require a yes or no answer (enter y or n and press RETURN).

(continued on next page)

Table 2-3. Symptoms of Boot Failure *(continued)*

SYMPTOM	SOLUTION
	fsck's prompts are largely self-explanatory. fsck describes the file system under question, and tells what it thinks is wrong. It then asks for a decision. These decisions take two basic forms (the actual wording varies, but you can recognize the form): • SALVAGE (RECONNECT, ADJUST, FIX, CLEAR) actions—these correct errors in indexing and pointers and such, things that you ordinarily do not see. Answer "yes" (y) to these questions. • REMOVE actions—these are the dangerous ones that remove data from the disk. Answer "no" (n) to these questions. After running fsck, the console shows a single-user UNIX prompt (#). Enter **reboot** and press return. Follow the autoboot procedure until it comes to fsck again (right after the **date/time:** prompt). fsck should then run without errors and the boot will continue. In this case, you can return to the normal autoboot procedure in Section 2.3. If fsck prints more errors, again run fsck manually. Enter fsck at the single-user UNIX prompt (#) and press RETURN. The goal is to remove as little data as possible from the disk. Follow this procedure: 1. Answer yes to all SALVAGE-form questions. 2. Answer yes to REMOVE-form questions at your discretion. Be careful; you can destroy data with these commands. It is usually safe to remove files and directories for which you know you have replacements on tape; you can fix these later by installing from tape. 3. Restart the autoboot with **reboot** (8). 4. If the autoboot does not finish normally, run fsck again with this same procedure. Eventually, fsck should allow the boot to finish. Return to the autoboot procedure in Section 2.3.

SECTION 3: SETTING UP YOUR SYSTEM

This section tells how to set up the UNIX system for normal access and use.

You should already have completed the procedures in

- "Section 2: Starting UNIX From Disk"
  or
- "Appendix A: Loading UNIX From Tape"

This means that you can now log in to the system as **root**, and that you have complete / and */usr* file systems mounted and ready. Check this by entering the **df**(1) command at the UNIX prompt (#). **df** should show both file systems and any other file systems on the disk(s). The *capacity* column should show all file systems partially filled (that is, greater than 0%).

This section contains the following subsections:

3.1 What Needs Setting Up?

When using a system shipped from Integrated Solutions, many of the following setup procedures are unnecessary. Follow the procedures anyway; you may want to change the existing setups. For example, you may want to add new user accounts or redefine the terminal connections.

If you have installed the UNIX system from a release tape, follow the procedures in this section to complete UNIX installation.

Use the procedures in this section to set up the server node of a cluster or diskless network. See "Section 6: Setting Up Cluster Nodes" or "Section 7: Setting Up Diskless Nodes" for procedures to finish setting up the complete networks.

These procedures tailor the system to your site's application. Use these procedures to

- give the system a name (hostname)
- define network administration parameters (local, distributed (TCP/IP), or NFS (YP))
- set up communication lines (terminals, modems, and printers)
- set up user accounts (logins)
- prepare user utilities (mail, uucp)

Before continuing, turn to "Appendix C: Configuration Worksheet" and fill it out as best you can. The procedures that follow will use this information.

When you set up your UNIX system, you will use an editor to modify files and add information. A display editor such as **vi** depends on having a terminal definition for it to operate properly.

If an editor does not give the expected display (for example, **vi** (1) runs in "open mode"), set the environmental variable TERM to match the terminal that you are using. Enter

```
setenv TERM termtype
```

where *termtype* is the **termcap** (5) name for the terminal. If you are using **sh** (1) rather than **csh** (1), use

```
TERM=termtype; export TERM
```

to set the TERM variable in the Bourne shell. See Section 3.4.1, "Defining Terminal Lines."

3.2 Using admin (8)

The **admin** (8) program provides automatic tools for many system administration tasks. You should use **admin** when possible, due to the complexity of many administration details. If you want to understand the UNIX system better, the procedures in this Guide supply information on the workings of **admin** so that you can see what is happening when **admin** executes.

The **admin** (8) program can help with these tasks:

- Initializing your system, setting up administrative conditions
- Adding or removing user accounts
- Setting up a network
- Setting up uucp
- Installing or maintaining a printer
- Installing cluster nodes
- Installing diskless nodes

CAUTION

> Execute **admin** (8) while in single-user UNIX mode, with */usr* mounted. Some **admin** functions can cause problems while in multiuser UNIX mode. For a cluster or diskless network, use **admin** only on the server node.

The following procedure tells you how to invoke the **admin** (8) administration program.

Procedure to Use admin (8)

	ACTION	WHAT IT DOES
Step 1.	Boot UNIX to single-user mode.	Brings the system to a safe condition for **admin** operations. See Section 8.3, "When Do You Boot Single-User?" to move between UNIX modes. Booting to single-user mode is a precaution, to safeguard against interference from other users during **admin** functions.
Step 2.	Enter the command **mount** and check that the */usr* file system is mounted.	Checks that the */usr* file system is mounted. If */usr* is not present, you must mount it with the **mount** (8) command: **mount** /dev/*devpart* /usr where *devpart* is the disk partition holding */usr*. Do not forget the /dev/ before *devpart*. For example, to mount the */usr* file system on the *g* partition of the second 269-Mbyte disk (VED269) of an Optimum V24 system, enter **mount** /dev/sm1g /usr
Step 3.	Enter the command /etc/admin <-*h*> The -h option runs **admin** in "hardcopy" mode. **admin** will avoid reprinting menus, printing only the appropriate prompt when a selection is necessary. Use this option if the console is a hardcopy or teletype device.	Invokes the **admin** (8) program. The first time that you run **admin** you will run through an initialization procedure (see Section 3.3.4, "Initializing With **admin** (8)"). You must complete this initialization sequence before using **admin**. Before **admin** offers its menu of administration procedures, it checks some administrative parameters. **admin** checks for these valid data: • hostname in */etc/rc.local* • password administration mode (local, TCP/IP, or Yellow Pages) • various network-related files are present See Section 3.3, "Initializing Your System", for further information on these parameters.

End of procedure

The **admin** administration program prints this menu:

```
                    ADMIN TOP LEVEL MENU

                    (A)     Display/modify admin parameters

                    (B)     User account maintenance

                    (C)     Diskless/cluster installation and deletion

                    (?)     Help

                    (Q or X)Exit from admin

                    Please enter letter for desired function:
```

Enter your choice by letter, with no carriage return. The subsequent menus and prompts ask you for specific information; in most cases, the prompts are self-explanatory.

3.3 Initializing Your System

The **admin** (8) program sets up many operating parameters for you the first time that you run it. Before running **admin**, have ready this information, as described in the following paragraphs:

- a hostname (system name)
- an internet network address
- a password administration mode (local, TCP/IP, or YP)

3.3.1 Choosing a Hostname

The original ISI UNIX release software does not define the system name.

The term "hostname" refers to the name of the system, as set by **admin** (8). The term "node" refers to any system, workstation, cluster node, or diskless node on a network.

Choose a hostname for your system (node). The hostname must be unique within your intended local network. Use a hostname that is less than twelve characters long. Do not use spaces within hostnames: instead, use underscores to separate words, as in v24_name.

You will provide this hostname to **admin** prompts. You might also use this hostname elsewhere: for example, if you configure a new kernel for the system (see "Section 4: Configuring Your System").

3.3.2 Choosing a Network Address

Choose an internet address for your system (node). If you will connect your node to more than one network (this is a "gateway" node), you must choose a different internet address for each network.

The file */etc/hosts* defines internet addresses and hostnames; look in this file for examples of internet addresses. Network addresses have this format:

n.n.n.n

where each *n* is a decimal number 1-255.

The first of the four numbers in an internet address is the "network number". If you have a local cluster network, for example, this number must be the same for all cluster nodes within a single card cage. A gateway node, connecting to more than one network, must use a different network number for each network.

As an example, if you are setting up a cluster network of three machines, you might choose the following internet addresses:

1.0.0.1
1.0.0.2
1.0.0.3

If you want to connect the server node of this cluster network to a diskless network (a local Ethernet network), you would give a different *network number* to the diskless network:

2.0.0.1
2.0.0.2
2.0.0.3

This will work for installations where there is no connection to a public network such as ARPANET or USENET. If your nodes will be hosts on a public network, you must register with the public network administrator to make certain that you do not choose a conflicting internet address. Contact the administrator for your ARPANET or USENET news feed node.

Enter the hostname and internet address in "Appendix C: Configuration Worksheet" under ''System name.'' If you are setting up the server node of a cluster or diskless network, also write the hostname and internet address in the ''Networks'' section of the Configuration Worksheet. You will be configuring only the server node at this time; to configure the cluster nodes, use the procedures in "Section 6: Setting Up Cluster Nodes."

3.3.3 Choosing a Password Administration Mode

Decide on the password administration mode for your node, from among these three choices:

1. Local: this is an isolated node that does not depend upon any other network host for administrative control or maintenance. Changes to */etc/passwd* have only local effects.
2. TCP/IP: this node may come under administrative control from another node on the network. admin will try to keep user identifications unique among an identified set of nodes. Modifying files on this node may affect other nodes on the network.
3. YP: this node is part of an NFS (Network File System) Yellow Pages network. Administrative changes may affect all nodes sharing the same YP server.

If you are installing the node in an existing network environment, the choice should be predetermined by the other nodes on the network. If you are installing in a fresh environment, give some thought to your long-range network configuration plans.

In the case of TCP/IP or YP administration, you must have ready this information:

- a list of the other nodes (hostnames and internet addresses) participating in this common administration.
- the name and address of the central password administration node on this network.

3.3.4 Initializing With admin (8)

You are now ready to use **admin** (8) to initialize your system. See Section 3.2, "Using **admin** (8)," and execute the **admin** program.

admin now prompts you for information. You will only have to enter this information once, the first time you run **admin**. You can change this information later, using the "Display/Modify admin parameters" menu choice in **admin**.

System initialization follows the steps in Table 3-1. **admin** does not actually change system files until you have completed the entire procedure; if you quit **admin** before performing all steps, the system will return to its previous state. When Table 3-1 refers to affected files, remember that **admin** does not actually write the modifications until the procedure is complete.

Table 3-1. **admin** System Initialization

What **admin** (8) Does	Affected Files
Prints the message **First-time admin initialization** This initialization occurs the first time you run **admin**. You must complete this initialization sequence before using other **admin** procedures. If **admin** prints the message --- Can't locate *filename* --- this means that the **admin** temporary file area */usr/lib/admin/local-data* where **admin** does record-keeping, is corrupt. The best thing is to remove the temporary file area and start from scratch with **admin**. Use the command rm -r /usr/lib/admin/local-data	—
Checks that you are in single-user mode. You cannot perform system initialization in multiuser mode.	—
Prompts you for a hostname. See Section 3.3.1, "Choosing a Hostname". This must be a unique hostname on your local network. Enter a question mark (?) for help.	*/etc/hosts* */etc/rc.local* */usr/spool/uucp/**
Prompts you for network controllers and internet addresses. See Section 3.3.2, "Choosing a Network Address". If you have no network connections, you do not have to answer these prompts. You must enter this information for each network controller: • Controller name (for example, "nw0" or "ttyh3") • Internet address (for example, "1.0.0.1") • Destination internet address, if the controller is a serial line After collecting this information, **admin** verifies it before continuing.	*/etc/hosts* */etc/rc.local*

(continued on next page)

Table 3-1. admin System Initialization *(continued)*

What admin (8) Does	Affected Files
Prompts for the password administration mode. See Section 3.3.3, "Choosing a Password Administration Mode". Choose one of the following: • Local • TCP/IP • Yellow Pages (you must have NFS)	—
Having collected the necessary information, admin performs these initialization tasks: • Puts the hostname in */etc/rc.local* (**hostname** (1)) • Puts network controller configurations in */etc/rc.local* (**ifconfig** (8)) • Adds network information to */etc/hosts* (**hosts** (5)) • Sets up uucp directories and files — Creates */usr/spool/uucp* — Creates configuration files in */usr/spool/uucp* — Creates accounting files in */usr/spool* — Changes ownership (uucp) and group ownership (daemon) of */usr/spool/uucp*	*/etc/hosts* */etc/rc.local* */usr/spool/uucp*

3.4 Setting Up Terminals, Modems, and Printers

The procedures in Sections 3.4.1, 3.4.2, and 3.4.3 tell how to set up terminal lines and how to specify the type of device connected to these lines. This includes connecting print devices like line printers or letter quality printers. Refer to Section 4.6 for information on making new devices.

Table 3-2 shows the type of cable needed to connect between peripherals and ports. See the appropriate Installation Manual for more details.

- "Standard" refers to a standard RS232C interface cable (also called a "straight-through" cable).
- "Twisted" refers to a RS232C interface cable with pins 2 and 3, and 8 and 20 *swapped* (also called a "null-modem" cable).

Table 3-2. Cables for Peripheral Ports

Peripheral Device	Console Port or Port 0	Other Serial Ports (Optimum V)	Serial Ports on an Optional Q-Bus Breakout Box
Terminals	Standard	Standard	Twisted
Modems	Twisted	Twisted	Standard

3.4.1 Defining Terminal Lines

Suggested reading:

- tset (1)
- ttys (5)

Define the system's terminal lines in the file */etc/ttys*. Use Table 3-3 to find the hardware connections; the right side of the table shows how to name the terminal lines.

Table 3-3. Terminal Line Names for */etc/ttys*

System	Controller	Number of Lines	Name
Optimum	dh (ISDH)	16	ttyh*X* (*X* is hex, 0-f) ttyi*X* for second DH
	dz (ISDZ)	8	tty*XX* (*XX* is decimal, 00-07)
Optimum V System[†]	cp (VME-ICP8/X)	8	ttyh*X* (*X* is hex, 0-7, or 0-f) Do not use lines ttyh[8-f]; use ttyi*X* for second VME-ICP8/X ttyj*X* for third, etc.
	cp (VME-ICP16/X)	16	ttyh*X* (*X* is hex, 0-f)
Optimum V WorkStation	ttyw	16	ttyw*X* (*X* is hex, 0-f)
All	std		Console port: console
	(modems are controlled same as terminals)		Dial-in lines (modems): ttyd*X* (*X* is hex, 0-f)
	pty		Pseudo-terminals (ptys): ttyp*X*, ptyp*X* *or* ttyq*X*, ptyq*X* (*X* is hex, 0-f)

[†] On an Optimum V24 serial port panel, ttyh0 actually connects to the port labeled "Port 1," and so on.

Use the following procedure, the Procedure to Define Terminal Lines, to define the terminal lines.

Procedure to Define Terminal Lines

	ACTION	WHAT IT DOES
Step 1.	Edit the file */etc/ttys*.	Defines terminal lines with this format: *tty getty type status comments* where *tty* is the name for the line, from Table 3-3. *getty* is the command to execute for the line, usually a call to getty (8). This field is usually in quotes, as it contains more than one word. When calling getty, specify the baud rate for the line. *type* is the type of terminal usually connected to the line, as defined in */etc/termcap* (termcap (5)). Some common types are • unknown when you do not know the terminal type • vt100 for the DEC VT100 and emulators • wyse50 for the Wyse 50 • f100 for the Freedom 100 • dialup for modems (ttyd*X*) • network for pseudo-terminals (ttyp*X*) • iswindow for the Optimum V WorkStation monitor *status* is one or more of the following: • on to enable this line • off to disable this line (the default) • secure to allow root logins on this line • desktop for an Optimum V WorkStation • window="*command*" to have init (8) execute the named *command* before starting getty. *comments* is a comment string that describes this line. Look in the file */etc/ttys* for examples, and see ttys (5).

(continued on next page)

Procedure to Define Terminal Lines *(continued)*

	ACTION	WHAT IT DOES
Step 2.	Add lines or edit existing lines to reflect your site's terminal lines.	Tailor the */etc/ttys* file to your site. */etc/ttys* should already contain line definitions; often all you need to do is choose the line you want to enable, and remove the initial "#" character that comments the line.
Step 3.	If you change */etc/ttys* and want to implement the change immediately (for example, after enabling or disabling a terminal line), enter the command **# kill -HUP 1**	Sends a hangup signal to **init**(8), making **init** notice the change in */etc/ttys*. **kill -1 1** also works.
Step 5.	**cd** to */dev* and check that each terminal line enabled in */etc/ttys* has an associated special file in */dev*.	Gives UNIX a way of communicating with the terminal on that line. For example, if */etc/ttys* contains the line **ttyi5 "/etc/getty std.9600" vt100 on** */dev* must contain the file */dev/ttyi5* Files named */dev/ttyXX* are special files; when UNIX writes to these files, it is actually writing to terminal screens. If */dev* does not contain the necessary tty files, see Section 4.6, "Making New Devices in */dev*."
Step 6.	Turn to "Appendix C: Configuration Worksheet" and note any changes.	Updates the reference sheet.

End of procedure

3.4.2 Setting Up a Modem

Set up a modem just as a terminal. Note that in Table 3-3 dial-in lines have the name

```
ttydX
```

where *X* is a hexadecimal digit signifying the port used on the serial controller.

Use these steps to set up a modem port:

1. Change the name of the */dev* special file for that port. For example, for the ninth port (numbered from 0) of a VME-ICP16/X, change */dev/ttyh8* to */dev/ttyd8*.
2. Add the modem port to the file */etc/ttys* with a line

   ```
   ttydX "/etc/getty std.baud" dialup on
   ```

   where *X* is the port number and *baud* is the baud rate.

For example, for a 1200-baud modem on the twelfth port of a VME-ICP16/X, put

```
ttyda "/etc/getty std.1200" dialup on
```

in */etc/ttys*. Change */dev/ttyha* to */dev/ttyda*.

Use a "twisted" RS232C cable (''straight'' on a Q-bus Optimum system breakout box) to connect the modem to the host port. Connect the modem to a phone line as instructed in the modem's installation manual.

When using a terminal and dialing in over the modem line, check that the terminal and modem baud rate at your end matches the baud rate at the host's end.

3.4.3 Setting Up a Printer

You can connect a parallel printer or a serial printer to your system. The following Procedure to Connect a Printer describes the necessary steps. If you are connecting a parallel printer, skip *Step 1*.

Parallel printers connect to ISI systems in two different ways:

- On an Optimum system, you connect the parallel printer to the PRINTER port.
- On an Optimum V System or WorkStation, you connect a parallel printer to the special port on the VME-ICP16/X (VME-ICP8/X) controller.

Because a parallel printer does not have an entry in */etc/ttys*, you must configure the parallel printer driver in the UNIX kernel (see "Section 4: Configuring Your System"). See lp (4) for a description of the parallel interface.

Serial printers connect to the system as serial devices, much like terminals, on an Optimum or Optimum V System or WorkStation.

The line printer software system includes these files and commands:

/usr/ucb/lpq	spooling queue examination program
/usr/ucb/lprm	program to delete jobs from a queue
/usr/ucb/lpr	program to enter a job in a printer queue
/etc/printcap	printer configuration and capability data base
/usr/lib/lpd	line printer daemon, scans spooling queues
/etc/lpc	line printer control program
/etc/hosts.lpd	list of hosts allowed to use the printers

Procedure to Connect a Printer

	ACTION	WHAT IT DOES
Step 1.	Skip this first step if you are connecting a parallel printer. Define the serial line as for a terminal (see Section 3.4.1) with these exceptions in the file */etc/ttys*: • Disable the printer line by specifying "off" in the status field. • Define the printer as a VT100 terminal type.	Sets up the printer with the necessary definitions for UNIX to reach the printer. The printer is essentially a disabled terminal; the "disabled" status simply means that getty (8) will not expect the printer to act like a terminal. For example, to describe a 9600-baud printer on the second port of a VME-ICP16/X controller, use `ttyh1 "/etc/getty std.9600" vt100 off` The "vt100" definition is a dummy; this is because */etc/ttys* relies on **termcap** (5) for its definitions. You define printers with **printcap** (5), as described in the next step.
Step 2.	Define the printer in the file */etc/printcap*. The default definition, **lp**, already in */etc/printcap*, will work for some printers. See **printcap** (5) to define other printers.	Defines printer specifications so that UNIX can format output. The */etc/printcap* entry defines the baud rate and other characteristics of the printer, and assigns a name to the printer. Use this name to specify which */etc/printcap* entry applies to the printer; **lp** is the default. See also the "Line Printer Spooler Manual" (SMM:6) in the *UNIX System Manager's Manual.*
Step 3.	Check that the */etc/rc* command file contains a command to start the printer queue daemon **lpd**. The line in */etc/rc* should look like `/usr/lib/lpd &`	Checks that */etc/rc* initializes the printer daemon when the system boots. This daemon should be running at all times.

(End of procedure)

3.5 Setting Up User Accounts

Use the **admin** (8) program to add or change user information automatically. When you invoke **admin** (8) have this information ready:

- Account name (user's login name). The login name should be at least six characters long, for security, but you can use shorter names.
- Group number (the group to which the user belongs). Look in */etc/group* for existing groups or to form a new one.
- User's real name.
- User's home directory on this node (a fully-specified pathname).
- User's preferred UNIX shell (usually */bin/csh*).
- User's home node, where s/he receives mail (for local networks). This will not be necessary if this node is under "local" **admin** (8) password administration.

admin follows the steps in Table 3-4 when adding a user account. If you terminate the **admin** session before completing the procedure, the */etc/passwd* file will remain untouched.

You might want to consider security and accounting needs on your system before you set up groups and users. If you need to restrict access to certain files by group or if you need to track system usage by group or department, you should separate users into different groups. Remember that members of the same group have access to all files created by that group, provided the group "read" permission is enabled (see chmod (1)).

Table 3-4. **admin** Adding User Accounts

What **admin** (8) Does	Affected Files
Displays the top-level menu. Choose item B, "User account maintenance."	—
Displays the user administration menu. Choose item A, "Add a user account."	—
Prompts you for information about the user. Enter this information: · • login account name • group number • real name • home directory on this node • preferred shell • home node, where mail should go (for "TCP/IP" and "YP" password administration)	—
Gives the new user a login account. The syntax of a new */etc/passwd* entry is this: *name:password:number:group:description:home:shell* where *name* is the login account name. *password* is the user's encrypted password. **admin** provides an initial value for this password; you should give this temporary password to the user and encourage him/her to run passwd (1) after logging in. *number* is a unique account number for this login. **admin** chooses the next available unique number on this node. Under "TCP/IP" or "YP" password administration, **admin** chooses a unique number with respect to all nodes under common administration. *group* is a group number. See group (5). *description* is a brief description of the user, for use by finger (1). In this case, this field is simply the user's real name. *home* is the user's home directory. *shell* is the shell used by UNIX when the user logs in. Most 4.3BSD users prefer */bin/csh*. Here is a sample */etc/passwd* entry: suzanne::23:10:Suzanne Koenig:/u/suzanne:/bin/csh	*/etc/passwd*
Creates the user's home directory, as supplied to prompts above. **admin** also changes ownership of the directory to the user's account and installs default versions of *.login*, *.logout*, and *.cshrc* files.	user's home directory *~/.login* *~/.logout* *~/.cshrc*

admin (8) provides the new user with default *~/.login*, *~/.logout*, and *~/.cshrc* files. If you wish, you can build new ones for your site and install them manually after running **admin.**

After installing *.login*, *.logout*, and *.cshrc* files, use **chown** (8) to give ownership to the user. This sets up the environment for a new user at login, reducing the shock for a new user who is unfamiliar to UNIX.

The *.login* file should contain commands to set up the terminal (for example, **setenv** TERM vt100), while the *.cshrc* file should hold C-shell commands (for example, **alias** commands (csh (1)).

Table 3-5 shows sample *.login* and *.cshrc* files for new users. **admin** (8) provides different versions of these files as the default.

The *.login* file executes once when the user logs in. The *.cshrc* file executes every time a C-shell is invoked. For example, if you execute a "shell" command in vi and want to have your familiar prompt rather than the default "%" C-shell prompt, you should place the **set prompt** command in *.cshrc*.

Table 3-5. Sample *.login* and *.cshrc* Files

File	Comments
# Sample .login	Comment (see csh (1) for the following commands)
set prompt="\!# "	Numbers each csh prompt for **history** commands
set history=20	**history** will remember 20 commands
set path=(Spath ~/bin)	Adds *~/bin* to the command search path
set mail=60	Checks for new mail every minute (default 10 minutes)
set notify	Lets you know immediately when jobs finish
set noclobber	Helps protect files from accidental erasure
set ignoreeof	Prevents accidental logout from stray signals
# Set terminal to vt100	Comment (replace *vt100* with the terminal name)
set noglob	Local disable of variable expansion, for next line
set term=('tset -S -k ?vt100')	Prompts user for termtype at each login, default *vt100*
setenv TERM Sterm[1]	Sets the environmental variable TERM
setenv TERMCAP Sterm[2]	Sets the environmental variable TERMCAP
unset term noglob	
# Execute at login	Comment (these commands run at login, before first prompt)
echo	Throws a blank line (RETURN)
uptime	Prints current time and system load
echo	
msgs -fp	Prints any system messages (mailed to msgs (1))
echo	
# Sample .cshrc	Comment (see csh (1) for alias command)
alias ls 'ls -F'	Friendlier ls (1) command
alias rm 'rm -i'	Prevents accidental removal of files
alias mv 'mv -i'	Same
alias cp 'cp -i'	Same
alias pd pushd	Easier to type
alias ppd popd	
alias m more	
alias j jobs	
alias c clear	
alias h history	

3.6 Setting Up a Mail System

Electronic mail allows users to communicate with each other on a single node or across linked nodes. This section describes how to use the UNIX **mail** (**binmail** (1)) and **Mail** (**mail** (1)) commands. See Section 3.7, "Setting Up Uucp," for a description of networked mail systems (mailing between nodes).

The program */bin/mail* (**binmail** (1)) invokes a simple mail facility, a holdover from Version 7 UNIX. */bin/mail* is indeed simple, and it runs quickly, but it has little power and the interface is terse at best. For most purposes, use */usr/ucb/Mail*, which is linked to */usr/ucb/mail* (**mail** (1)).

The environmental variable PATH determines which **mail** program executes when you enter the word **mail** to a csh prompt. If the path has */usr/ucb* before */bin*, as the default path does, UNIX uses */usr/ucb/Mail*. To override this, use the **alias** command

```
alias mail /bin/mail
```

or simply specify **/bin/mail** when you want that program. Entering **Mail** will always invoke */usr/ucb/Mail*.

The default UNIX autoboot starts the mail facilities automatically. The default */etc/rc* file **rc** (8) starts the **comsat** (8C) daemon (with the **inetd** (8) daemon), and */etc/rc.local* starts the **sendmail** (8) daemon.

Unless something breaks, administering the mail system simply means ensuring that users know how to use it. One way that you can help is to give new users a *~/.mailrc* file, along with the *.login* and *.cshrc* files. **admin** (8) does not provide a default *.mailrc* file as it does the others. Table 3-6 shows a sample *.mailrc* file.

Table 3-6. Sample *.mailrc* File

File	Comments
# Sample .mailrc	Comment (see mail for these commands)
ignore message-id	Clears extraneous trash from mail headers
ignore received	
ignore sender	
ignore status	
ignore date	
set ask	Prompts for a **Subject:** line when you send mail
set autoprint	Speeds up reading mail
set quiet	Keeps screen uncluttered
set EDITOR /usr/ucb/ex	Chooses an editor for the edit command in mail
set VISUAL /usr/bin/vi	Chooses an editor for the **visual** command in mail

Another of your tasks is to maintain the system **aliases** (5) file, */usr/lib/mail/aliases*. This file allows users to mail to a list of other people as a single group. For example, **engineering** may be an alias that includes twenty people. Aliases may themselves contain aliases; **engineering** may have the */usr/lib/mail/aliases* definition

```
engineering: softeng hardeng techsupport
```

where **softeng**, **hardeng**, and **techsupport** are previously-defined aliases. **aliases** (5) explains the alias system.

For further information on the mechanics of mail, see the "Mail Reference Manual" and "Sendmail Installation and Operation Guide" (SMM:7) in the *UNIX System Manager's Manual*.

3.7 Setting Up Uucp

Uucp (UNIX-to-UNIX copy) transfers files between nodes over a phone line. Use this to copy data between nodes or to connect the node to a distributed mail network.

For networking connections involving an Ethernet board, see "Section 5: Setting Up Your Network."

To connect two UNIX nodes with a uucp link, one site must have an automatic call unit and the other must have a dial-up line with an auto-answer modem. It is best if both sites have both.

Uucp is complicated. The description in this section outlines only the bare bones of installing and operating uucp. See "Uucp Implementation Description" (SMM:9) in the *UNIX System Manager's Manual* for more information.

Uucp software resides in three major directories: */usr/bin* (user commands), */usr/lib/uucp* (system commands), and */usr/spool/uucp* (spooling area). These commands are in */usr/bin*:

/usr/bin/uucp	file-copy command
/usr/bin/uux	remote execution command
/usr/bin/uusend	binary file transfer using mail
/usr/bin/uuencode	binary file encoder (for **uusend** (1C))
/usr/bin/uudecode	binary file decoder (for **uusend** (1C))
/usr/bin/uulog	scans session log files
/usr/bin/uusnap	gives a snapshot (summary) of uucp activity
/usr/bin/uupoll	polls remote system until an answer is received
/usr/bin/uuname	prints a list of known uucp hosts
/usr/bin/uuq	gives information about the queue

These are the important files and commands in */usr/lib/uucp*:

/usr/lib/uucp/L-devices	list of dialers and hard-wired lines
/usr/lib/uucp/L-dialcodes	dialcode abbreviations
/usr/lib/uucp/L.aliases	hostname aliases
/usr/lib/uucp/L.cmds	commands remote sites may execute
/usr/lib/uucp/L.sys	systems to communicate with, how to connect, and when
/usr/lib/uucp/SEQF	sequence numbering control file
/usr/lib/uucp/USERFILE	remote site pathname access specifications
/usr/lib/uucp/uucico	uucp protocol daemon
/usr/lib/uucp/uuclean	cleans up garbage files in spool area
/usr/lib/uucp/uuxqt	uucp remote execution server

The spooling area contains these important files and directories:

/usr/spool/uucp/C.	directory for command, "C." files
/usr/spool/uucp/D.	directory for data, "D.", files
/usr/spool/uucp/X.	directory for command execution, "X.", files
*/usr/spool/uucp/D.*machine	directory for local "D." files
*/usr/spool/uucp/D.*machine*X*	directory for local "X." files
/usr/spool/uucp/TM.	directory for temporary, "TM.", files
/usr/spool/uucp/LOGFILE	log file of uucp activity
/usr/spool/uucp/SYSLOG	log file of uucp file transfers

Use the following procedure, the Procedure to Set Up Uucp, to set up uucp data bases and administer the uucp software. This procedure is very terse; again, refer to "Uucp Implementation Description" (SMM:9) in the *UNIX System Manager's Manual* for more information.

Procedure to Set Up Uucp

	ACTION	WHAT IT DOES
Step 1.	Check that these files exist: */usr/spool/uucp/D*.node-name */usr/spool/uucp/D*.node-name*X* where "node-name" is the hostname of your node, as it will be used in mail addresses.	Checks for spooling directories. **admin**(8) sets up these two directories, with the proper name for your node, during **admin** initialization. See Section 3.3, "Initializing Your System."
Step 2.	If you have an autodialer on the node, edit these files: */usr/lib/uucp/L-devices* */usr/lib/uucp/L-dialcodes* */usr/lib/uucp/L.sys* "Uucp Implementation Description" gives the formats for these files.	Allows dialing in to other nodes. *L-devices* lists modem lines, *L-dialcodes* gives abbreviations for phone numbers, and *L.sys* describes the nodes you can dial into, complete with logins and passwords. An additional file, *L.cmds*, defines the commands that may be executed by a remote host.
Step 3.	Use the commands uucp (1C) uusend (1C) uux (1C) to send and receive information through uucp connections.	Activates the phone lines and transfers files as ordered. Some sites specify certain times when they are available, and you cannot dial in outside those times.
Step 4.	Send mail through uucp connections with the mail (1) command. See mailaddr (7) for the format of mail addresses through uucp connections.	Stores outgoing mail, then dials in to other nodes and delivers that mail. Also picks up incoming mail. Make an entry in */usr/lib/crontab* (see cron (8)) to autodial to the mail addresses at given times.
Step 5.	Check uucp with the **uulog** (see uucp (1C)) command.	Gives information on uucp use, indexed by nodes or by users.
Step 6.	As System Administrator, occasionally run the uuclean (8) command.	Clears old files out of the */usr/spool/uucp* directories. These can get out of hand quickly if not cleaned occasionally.

End of procedure

3.8 Backing Up for Safety

After finishing the procedures in this section, save a copy of the UNIX system on magnetic tape. Then you can quickly restore the system in case of disaster.

Run a "level 0 dump" as described in Section 8.7, "Backing Up Your System."

Also, update "Appendix C: Configuration Worksheet" with any changes to the system configuration.

SECTION 4: CONFIGURING YOUR SYSTEM

This section tells how to adjust the UNIX system for different devices. This includes adding peripheral devices and changing the UNIX kernel.

If you are using a complete system shipped from Integrated Solutions, the system was completely configured at the factory. Use this section only if you want to change that shipped configuration.

If you started the UNIX system from a release tape, follow these procedures to configure the system.

This section contains the following subsections:

4.1 When to Change Your Configuration

Reconfigure the system under these conditions:

- Loading a fresh version of UNIX from tape
- Adding user-written device drivers
- Optimizing the kernel for efficiency
- Changing hardware (adding or removing devices)

All of these follow a general procedure:

1. Make changes to hardware or device software
2. Change the kernel while saving the old kernel
3. Reboot with the new kernel and check modified functions
4. Install the new kernel

Keep a log of changes to the system in "Appendix C: Configuration Worksheet."

4.2 Configuring the Kernel

The "kernel" is the core of the UNIX operating system. Without accurate information in the kernel, UNIX has no way to communicate with you or with peripheral devices.

The kernel usually resides in the file */vmunix*. When the system boots, this file is the starting point for UNIX's operation.

The 4.3BSD UNIX release tape includes a generic kernel with definitions for all bootable devices. When the system boots from this kernel, the autoboot software determines which devices exist and activate the associated software drivers.

4.2.1 Adding Your Own Device Drivers

You can write your own drivers for devices other than those in the release kernel. This requires proficiency in the C language and experience with UNIX device drivers. Read *Building Berkeley UNIX Kernels with Config* (SMM:2) and **intro** (4) for advice.

The following Procedure to Add Drivers to the Kernel shows the general procedure for adding new drivers to the kernel.

MOUNT -a

Procedure to Add Drivers to the Kernel

	ACTION	WHAT IT DOES
Step 1.	Edit the files */sys/conf/files* */sys/conf/devices.is68k*	Provides information for the device. See the comments in the files, and read *Building Berkeley UNIX Kernels with Config* (SMM:2).
Step 2.	Edit the file */sys/is68k/conf.c* (or, for source licensees:) */sys/conf/machine/conf.c* to add appropriate entries to the bdevsw and cdevsw tables.	Defines routines for access to the device. See the comments in the file itself.
Step 3.	Put the device driver, with a name of the form "dev.*c*", in the directory */sys/is68kdev*.	This is where config (8) expects to find device drivers.
Step 4.	Edit the file */sys/conf/*SYS*_DIST* where "SYS" is one of • *V20* for the VME-68K20 • *V10* for the VME-68K10 • *Q10* for the IS-68K (Q-bus) to add the device driver.	Defines the device driver for the kernel, so that UNIX can use the device. Look at the entries in the file, and read *Building Berkeley UNIX Kernels with Config* (SMM:2).
Step 5.	Make a new kernel, as described in Section 4.2.3.	Implements the new device driver; you can now test the device and driver.
Step 6.	Turn to "Appendix C: Configuration Worksheet" and note any changes you made.	Updates the reference sheet.

End of procedure

4.2.2 Optimizing Your Kernel for Efficiency

Use this procedure to streamline the kernel for greater efficiency.

A generic release kernel comes with definitions for many devices. Eliminating unused devices decreases the size of the kernel and allows UNIX to run faster.

Use the following Procedure to Optimize the Kernel to reconfigure the kernel for best performance.

Procedure to Optimize the Kernel

	ACTION	WHAT IT DOES
Step 1.	Edit the file */sys/conf/*SYS_*DIST* where "SYS" is one of • *V20* for the VME-68K20 • *V10* for the VME-68K10 • *Q10* for the IS-68K (Q-bus)	Locates the kernel definitions for what devices are in the system. The generic distribution kernel from Integrated Solutions defines all of the available devices.
Step 2.	Comment out any lines that do not apply to your configuration. A hash (#) at the beginning of a line comments out the line.	Eliminates unneeded device definitions from the kernel. You should trim the definitions down to only those used on your system. Table 4-1 shows controller names and descriptions for Optimum Systems. Table 4-2 shows controller names and descriptions for Optimum V Systems and Work Stations.
Step 3.	Change the "options" line(s), and other configuration definitions near the top of the file, to reflect your configuration.	Assigns configuration variables and removes unwanted options in the kernel. See Tables 4-1 and 4-2 for descriptions of options. The server node for cluster or diskless networks must define the maxusers line by this formula: maxusers is equal to *#users* + (6 x *#nodes)* where *#users* is the maximum number of users on the server node *#nodes* is the number of connected cluster or diskless nodes For example, a server node with eight users (maximum) and three diskless nodes would calculate (8 + (6 x 3)), resulting in this config line: maxusers 26
Step 4.	Make a new kernel, as described in Section 4.2.3.	Implements the kernel changes. You now have an optimized, faster system.

End of procedure

Table 4-1. config Definitions for Optimum V Systems and WorkStations

Variable Type	Name	Description
machine	is68k	The default, always defined
cpu	"M68020"	VME-68K20
	"M68010"	VME-68K10
ident	"V20"	VME-68K20
	"V10"	VME-68K10
timezone	*n*	Number of hours west of GMT
	dst	Daylight Savings Time
maxusers	*n*	Number of users
options	VBUS	Always defined
	GWS	Graphics
	INET	Internet
	TRFS	Transparent Remote File System
	APPLETALK	AppleTalk
	DEBUGGER	Kernel debugger
	ENETFILTER	Remote booting over Ethernet
	QUOTA	Quotas for file system use
	TCP_COMPAT_42	Compatible with 4.2BSD TCP
	UDP_COMPAT_42	Compatible with 4.2BSD UDP
config	vmunix	The default, always defined
controller (main)	QB0	*at nexus ?* The default, always defined
controller (disk)	SD{0,1}	VDS106
	SM{0,1,2}	VDS269, VDS418, VDS494
	SP0	Spanned disk
	LD{0,1}	Optical disk
controller (tape)	TS{0.1}	VTS25/R, VTS50/R, VTS6250/R
controller (serial)	CP{0,1}	VME-ICP16/X, VME-ICP8/X
controller (FFP)	SK0	VME-FFP Fast Floating Point board
controller (Ethernet)	NW{0,1}	VME-ECX
	EX{0,1}	VME-EC (Excelan)
controller (cluster)	VB0	Cluster nodes
pseudo-device	ether	Ethernet for TRFS
	imp	Internet IMP for INET
	loop	Internet loopback for INET
	pty	Pseudo-terminals
	rd	Remote disk
	sl 8	Serial line for INET
	vty	Graphics

Table 4-2. config Definitions for Optimum Systems

Variable Type	Name	Description
machine	is68k	The default, always defined
cpu	"M68010"	IS-68K (Q-bus)
ident	"Q10"	IS-68K (Q-bus)
timezone	*n*	Number of hours west of GMT
	dst	Daylight Savings Time
maxusers	*n*	Number of users
options	QBUS INET TRFS APPLETALK DEBUGGER ENETFILTER QUOTA TCP_COMPAT_42 UDP_COMPAT_42	Always defined Internet Transparent Remote File System AppleTalk Kernel debugger Remote booting over Ethernet Quotas for file system use Compatible with 4.2BSD TCP Compatible with 4.2BSD UDP
config	vmunix	The default, always defined
controller (main)	QB0	*at nexus ?* The default, always defined
controller (disk)	EL{0,1}	DS36, DS67, DS85, DS140
	HP{0,1}	DS33, DS418, DS474
	SP0	Spanned disk
controller (tape)	TS{0,1}	TS25, TS50
controller (serial)	DZ{0-7}	ISDZ
	DH{0-7}	ISDH
controller (printer)	LP{0-3}	ISLP line printer board
controller (FFP)	SK0	ISFFP Fast Floating Point board
controller (Ethernet)	EX{0.1}	ISEC
	IL{0.1}	Interlan Ethernet
controller (HPIB)	IB0	IEEE-488 (HPIB)
pseudo-device	ether	Ethernet for TRFS
	imp	Internet IMP for INET
	loop	Internet loopback for INET
	pty	Pseudo-terminals
	rd	Remote disk
	sl 8	Serial line for INET

4.2.3 Making a New Kernel

You must create a new kernel when

- You have changed your hardware configuration
- You have changed files in */sys/conf*

In any case, you should already have edited the file */sys/conf/*SYS*_DIST* before reaching this procedure. Use the following Procedure to Make a New Kernel to create a new kernel from the */sys/conf* files with config (8).

Procedure to Make a New Kernel

	ACTION	WHAT IT DOES
Step 1.	In the directory */sys/conf*, create a (or edit an existing) config (8) configuration file. If you have completed the procedures in either Section 4.2.1 or 4.2.2, you have already completed this step.	Creates the configuration file that config (8) uses to build the new kernel. The original distribution configuration files from ISI have the names */sys/conf/V20_DIST* */sys/conf/V10_DIST* */sys/conf/Q10_DIST* You can edit the appropriate file for your system, or you can copy the original to a new file name and use the new file as the configuration file.
Step 2.	If you are using a configuration file with a name different from the original ISI distribution files, enter the command mkdir /sys/*name* V15-DIST where *name* is the name of the configuration file in */sys/conf*.	Creates a directory in */sys* for kernel files. This directory already exists for the ISI distribution config files.
Step 3.	Enter these commands: cd /sys/conf config -o *name* where *name* is the name of the configuration file.	Runs the config (8) program on the configuration file. See *Building Berkeley UNIX Kernels with Config* (SMM:2) for details on config operations. If you are a source customer, do not use the *-o* option.
Step 4.	Enter these commands: cd /sys/*name* make depend where *name* is the name of the configuration file.	Generates the rules for source code dependencies in the kernel files. config created the *Makefile* in this directory in the last step.

(continued on next page)

devaddr.c can't find"

Procedure to Make a New Kernel *(continued)*

	ACTION	WHAT IT DOES
Step 5.	Enter the command **make vmunix**	Creates a bootable kernel file named *vmunix* in the current directory.
Step 6.	Enter the command **cp vmunix /vmunix.new**	Copies the new kernel to a file */vmunix.new* in the root directory (*/*). CAUTION Do not replace the old kernel until you have tested the new one.
Step 7.	Reboot the system with the **shutdown** (8) and **reboot** (8) commands, as described in Section 2.4, "Shutting Down the System." Continue until the PROM prompt (:) appears.	Shuts down UNIX, bringing the system down to the boot PROMs. You can now reboot with the new kernel.
Step 8.	Enter this command at the PROM prompt (:): *dev*(0,0)**vmunix.new** where *dev* is the name of the system's boot disk.	Boots UNIX with the new kernel. Note that this boot procedure is identical to the usual boot procedure except for specifying *vmunix.new* rather than *vmunix* as the kernel.
Step 9.	If the system boots, install the new kernel with these commands: mv /vmunix /vmunix.orig mv /vmunix.new /vmunix	Moves the new kernel to */vmunix.* UNIX will now use this kernel for autobooting. Always save the previous kernel as a backup. Write the name of the backup kernel in "Appendix C: Configuration Worksheet," under "Comments."

End of procedure

4.3 Configuring Disks

There are four main reasons for changing the disk configuration:

- Adding a disk to the system (or taking one away)
- Optimizing disk efficiency
- Creating a spanned disk
- Adding a new file system

4.3.1 Adding or Removing a Disk

When you add or remove a disk, you must make changes to some of the system files. See Section 4.5, "Changing Your Hardware Configuration," for procedures that tell you which files must change when you change hardware.

After you add (or remove) a disk, you must decide how to allocate file systems across the new (or remaining) disks. Section 4.3.2, "Optimizing Disk Efficiency," describes file system allocation.

You can also allocate file systems across physical disk boundaries. This type of disk allocation is called a *spanned disk* and changes previous restrictions on file system size and file system location. (Prior versions of UNIX restricted file systems to a single disk partition.) Section 4.3.3, "Creating a Spanned Disk," describes how to set up a spanned disk.

4.3.2 Optimizing Disk Efficiency

Every physical disk drive has seven or eight partitions, labeled *a-g[h]*. Typically, UNIX uses the partitions in this way:

a: Holds the root (/) file system or a backup copy of it.

b: Holds a paging and swapping area.

c: A dummy partition; actually refers to the entire disk. Note that this includes all partitions (*a-g[h]*) plus sector maps created by **bad144** (8).

g: The largest area, contains the *d*, *e*, and *f* partitions, used for large file systems.

h: On larger drives, typically used for user files.

Assign file systems to these partitions in such a way as to balance system load and optimize speed.

To balance the disk load, you must allocate five areas across the available disk partitions:

1. The root (/) file system
2. The */tmp* file system
3. The */usr* file system
4. The user files (login accounts)
5. Paging activity

With one disk, you have at least three partitions (*a*, *b*, and *g*) for allocation; with two disks you have at least six total partitions and so on. Clearly, with one small disk, you must double up some of the areas in the list above.

Table 4-3 shows a suggested division for disk allocation with one, two, or three disks.

Table 4-3. Example of Disk Allocation

Areas	Number of Disks		
	1	2	3
/	0a	0a	0a
/tmp	0a	1a	1a
/usr	0g	0g	0g
users	0g	1g	1g. 2g
paging	0b	0b, 1b	0b. 1b, 2b

Use the following procedure, the Procedure to Change the Disk Configuration, to reapportion disk resources.

Procedure to Change the Disk Configuration

	ACTION	WHAT IT DOES
Step 1.	Edit the file */etc/fstab*. In this file, list the disk partitions you want to use, and what file systems you want to mount on them. Table 4-3 suggests some file system allocations. See *Step 4* to add a second swap space (a swap space on another disk). Partition *b* on the first disk is always assumed as a swap space and needs no entry in */etc/fstab*.	Assigns file systems to disk partitions. **mount** (8) these file systems automatically with the **mount -a** command. Use this format for an fstab (5) entry: *devpart filesystem type options freq pass* where *devpart* is the name of the disk partition, as defined in the */dev* directory. For example, the *g* partition of a second SD disk in an Optimum V system would be */dev/sd1g*. *filesystem* is the file system that you want to mount on this disk partition. *type* is the type of file system, from these choices: • 4.3 for a standard block special device • nfs for remote disk access with NFS • swap for a swap partition • ignore for an unused fstab entry *options* is is a comma-separated list of options, as described under fstab (5). *freq* is the dump frequency, in days, for this file system (see dump (8)). *pass* is the order in which fsck (8) tests these file systems. For example, the fstab entry /dev/sd0a / 4.3 rw 1 1 connects the root file system (/) to */dev/sd0a*, a standard UNIX block device, in read/write mode. dump (8) will recommend backups for this file system every day, and fsck (8) will check this file system before all others.

(continued on next page)

Procedure to Change the Disk Configuration *(continued)*

	ACTION	WHAT IT DOES
Step 2.	Enter these commands: **dmesg** **diskpart -d** *disktype* where *disktype* is the type of the system disk, from Table A-4.	Compares the expected disk configuration to the actual configuration. **dmesg** (8) shows what the kernel thinks of as disk configuration, and **diskpart** (8) shows the entry from */etc/disktab.* The two should match, with **diskpart** showing one or two fewer cylinders than **dmesg**. **dmesg** prints for each disk drive the partition sizes and a line of this form: (*sectors/track* x *tracks/cylinder* x *#cylinders*) If **diskpart** shows sector or track counts different from **dmesg**, use **diskpart** to change the entry in */etc/disktab.* See **disktab** (5) and the comments in */etc/disktab* for the format of **disktab** entries. **diskpart** should show one or two fewer cylinders than **dmesg**. The extra cylinders shown by **dmesg** contain bad block information. See **bad144** (8).
Step 3.	Check that all of the file systems referenced in */etc/fstab* already exist. • If */etc/fstab* refers to a */mnt* file system, for example, check that the root directory (/) has a directory *mnt* in it. • If */etc/fstab* refers to a partition *sd1g*, for example, check that there is a special file */dev/sd1g*.	Checks that everything is in place for UNIX to implement your disk partitioning. If either of these checks fails, use the procedure in Section 4.3.4 to create a new file system. If you have only one disk, you are now done with this procedure.
Step 4.	If you want to create a swap space on the second disk, continue with this procedure. Otherwise, stop now. Add this line to */etc/fstab:* *devpart* ? swap where *devpart* is the partition of the second disk for use as swap space (for example. sd1b). ? is any character or string, ignored but necessary to fill the "filesystem" field of the fstab entry.	Defines a swap space on the second disk. UNIX will now mount this swap space as part of the autoboot (see **swapon** (8)). */etc/fstab* should look something like this: /dev/sd0a / 4.3 rw 1 1 /dev/sd0g /usr 4.3 rw,noquota 1 3 /dev/sd1a /tmp 4.3 rw 1 1 /dev/sd1g /u 4.3 rw 1 4 /dev/sd1b ? swap

(continued on next page)

Procedure to Change the Disk Configuration *(continued)*

	ACTION	WHAT IT DOES
Step 5.	Add this line to */etc/rc.local:* swapon -a	Activates all swap devices listed in */etc/fstab.* Without this line in */etc/rc.local,* swapping and paging occur on the default partition only.
Step 6.	Enter the command cd /usr/sys/conf Edit the appropriate configuration source file by finding the line config vmunix and replacing it with config vmunix swap on *dev0b* and *devXpart* where *dev* is the device name, *X* is the device number, and *part* is the partition.	Defines the swap areas for UNIX. This information is part of the kernel. For example, to add the *b* partition on device number 2 for the gd disk of an Optimum V system, enter config vmunix swap on gd0b and gd2b
Step 7.	Remake the kernel as described in Section 4.2.3.	Implements the change by providing a new kernel with the new swap information. You do not need to change the config (8) configuration file for the kernel.

End of procedure

4.3.3 Creating a Spanned Disk

By using a spanned disk, you can create a file system that is larger than a single physical disk, or that combines partitions of multiple disks. Without spanned disks, a UNIX file system is limited to a single disk partition.

A spanned disk is useful, for example, when you have several small partitions on two (or more) disk drives and you want to combine these small partitions for one single file system. Or, you might have a case where even one large partition is not big enough; you can add partitions from other drives until you reach the desired size.

You can define up to four (4) spanned disks, *sp0c* through *sp3c*.

You may not use a spanned disk under these conditions:

- The root file system (/) may not reside on a spanned disk.
- The swap area may not reside on a spanned disk.
- You may not use the *c* partition as part of any spanned disk. Instead, if you want to use an entire disk as part of a spanned disk, specify entries for the *a, b, d, e, f, g* (and, if available, *h*) partitions.

The following Procedure to Create Spanned Disks outlines the steps for dynamically allocating a spanned disk. See sp (4I) for further details.

Procedure to Create Spanned Disks

	ACTION	WHAT IT DOES
Step 1.	Determine the *(major,minor)* pairs of each partition that makes up the spanned disk. The *major* number refers to the controller type (see intro(4)) for the disk drive that your system uses. The *minor* number refers to the partition number.	The *(major,minor)* entries you create in */etc/sptab* tell the spanned disk driver how to configure the spanned disks. To determine major and minor device numbers, use the command ls -l /dev/*devNp* as described in Section 4.3.2, "Optimizing Disk Efficiency." The **ls -l** command shows the major and minor numbers for the device. For example, this command shows the (major,minor) pair (1,6) for the *g* partition of a VDS106 disk: # ls -l /dev/sd0g brw------- 1 root 1, 6 Aug 5 1986 /dev/sd0g
Step 2.	Add the spanned disk description to */etc/sptab*. For example, the two lines below describe two spanned disks: sp0c ((1,6),(1,13)) sp1c ((1,8),(1,11),(1,12))	The first spanned disk, sp0c, consists of drive 0, partition *g*, and drive 1, partition *f*. The second spanned disk, sp1c, consists of drive 1, partitions *a*, *d*, and *e*. See diskpart(8) for information on how to list partition sizes on various disks.

(continued on next page)

Procedure to Create Spanned Disks *(continued)*

	ACTION	WHAT IT DOES
Step 3.	Edit the file */etc/fstab* to assign file systems to the spanned disk(s).	Associates the spanned disk with a file system, and defines file system parameters. See fstab (5) and Section 4.3.2, "Optimizing Disk Efficiency." For example, this fstab entry defines a spanned disk holding the */usr* file system: /dev/sp0c /usr 4.3 rw 1 2
Step 4.	Enter these commands: cd /dev MAKEDEV sp0 *sp1 sp2 sp3* where you specify to the *MAKEDEV* command only the spanned disks you have defined in */etc/sptab.*	Creates device special files in the */dev* directory for the spanned disk(s). For more information on */dev/MAKEDEV*, see Section 4.6, "Making New Devices in */dev*."
Step 5.	Manually invoke spconfig using the -a option. Enter the command /etc/spconfig -a	This causes the system to recognize the spanned disk configuration immediately.
Step 6.	Add a call to spconfig in */etc/rc* with a line of the form: /etc/spconfig -a	This causes the system to recognize the spanned disk configuration during subsequent boot procedures. Place this line before any references to mount (8) or fsck (8).
Step 7.	Make a file system for this spanned disk with the mkfs (8) command.	Defines the file system for this disk. You must use the mkfs (8) command, as described in the sp (4I) man page, rather than the newfs (8) command. You can now mount (8) and use the new file system.

End of procedure

If you want to modify a spanned disk configuration, follow the steps shown in the Procedure to Modify a Spanned Disk.

All changes to spanned disk configurations must be done while in single-user mode with all file systems unmounted.

Procedure to Modify a Spanned Disk

	ACTION	WHAT IT DOES
Step 1.	If the spanned disk already has a file system on it, **dump** (8) the file system to tape.	Saves an existing file system. You can restore the file system to the reconfigured spanned disk after the end of this procedure. See Section 8.7, "Backing Up Your System," for instructions to dump (and later restore) the file system.
Step 2.	Boot UNIX to single-user mode. Unmount all file systems with the command **umount -a**	Prepares for changes to the spanned disk configuration. You must be in single-user mode with the file systems unmounted.
Step 3.	Edit the file */etc/sptab*. Change all *(major,minor)* pairs to 0,0.	Resets the spanned disk configuration.
Step 4.	Type the command /etc/spconfig -a	Deconfigures the existing spanned disk(s).
Step 5.	Edit the file */etc/sptab*. Change all *(major,minor)* pairs to match the new configuration for the spanned disk(s).	Defines the new spanned disk configuration.
Step 6.	Type the command /etc/spconfig -a	Configures the new spanned disk(s).
Step 7.	If you have changed the size of the spanned disk, you must use the mkfs (8) command to create a new file system for the disk.	Makes a new file system to fit the modified spanned disk. You must use the mkfs (8) command, as described in the sp (4I) man page, rather than the newfs (8) command. You can now mount (8) and use the new file system.

End of procedure

4.3.4 Adding a New File System

You may want to make a new file system when you find that some part of an existing file system is growing beyond available space. For example, if you currently keep user accounts in the */usr* file system, you may want to create a */u* file system on a second disk.

The following Procedure to Make a New File System assumes that you know what file system you want to create and on what disk partition you want it.

Procedure to Make a New File System

	ACTION	WHAT IT DOES
Step 1.	Enter the command cd /dev; ls	Lists the contents of the */dev* directory. Check that the desired disk partition is one of the entries in this directory. The entry should be of the form *devNp* where *dev* is the name of the disk, from Table A-1 *N* is the device number (0 for the first disk, and so on) *p* is the partition, a letter *a-h* If it is not there, use the *MAKEDEV* script to create it. See Section 4.6, "Making New Devices in */dev*." NOTE SMD-type devices are numbered sm0, sm2, sm4, and so on.
Step 2.	Enter this command: newfs /dev/r*devpart disktype* where *devpart* is the disk partition (for example, sd0g). Do not forget the /dev/r before *devpart*. (For an Optimum System, with a Q-bus, leave out the r.) *disktype* is the type of disk. Use the name from the */etc/disktab* entry (for example, ved106) (see Table A-4).	Readies the disk partition for use with a file system. newfs (8) will print some information about available space. For example, to create a new file system on the *g* partition of the second 106-Mbyte disk (VED106) of your Optimum V system, enter newfs /dev/rsd1g ved106 NOTE You must use mkfs (8) rather than newfs to configure a spanned disk. See sp (4I).

(continued on next page)

Procedure to Make a New File System *(continued)*

	ACTION	WHAT IT DOES
Step 3.	Enter the command ls -F *dir* where *dir* is the directory in which you want to put the file system. For example, to list the */usr* file system, use ls -F /	Shows the contents of the parent directory for the file system. Check that the file system already exists in this directory; you cannot mount a directory that does not exist. The *-F* option to ls(1) prints a slash (/) after directories in the list. Therefore, for a */mnt* file system, you should see mnt/ in the root (/) directory. If the desired directory does not exist, create it with the mkdir(1) command.
Step 4.	Enter this command: mount /dev/*devpart directory* where *devpart* is the disk partition (for example, sm2g). Do not forget the /dev/ before *devpart.* *directory* is the path to the head of the file system (for example, /mnt)	Mounts the file system on the requested directory. For example, to mount the file system */u* on the *g* partition of the second 418-Mbyte (VED418) disk of your Optimum V system, enter mount /dev/sm2g /u
Step 5.	Edit the file */etc/fstab* to assign the file system to a disk partition.	Associates the file system with a disk partition, and defines file system parameters. See fstab(5) and Section 4.3.2, "Optimizing Disk Efficiency." For example, this fstab entry defines a */usr* file system on the seventh (*g*) partition of a VDS106 disk: /dev/sd0g /usr 4.3 rw 1 2

End of procedure

4.4 Selecting Tape Device Files

Tape device special files reside in the directory */dev*, with names of the form **mt**. However, the */dev/tape* directory provides a larger set of tape device special files. Tape device files are still available through the standard naming convention (such as *mt0*, *rmt12*, etc.) but the standard names are linked to tape devices in the directory */dev/tape*. You can use the standard files, or you can use the wider variety of tape devices in */dev/tape* directly.

The tape devices in */dev/tape* fall into three basic families:

- *q24* — conforms to QIC-24 format specifications
- *q11* — conforms to QIC-11 format specifications
- *std* — the default format, set by ISI to QIC-24 format specifications

Filenames in the */dev/tape* directory consist of descriptive strings connected by underscores. The filename describes the function of the tape device. For example, the tape device *std_blk_norew_0* is the default, logical-block-mode, non-rewinding device for the first tape drive (ts0). This tape device happens to be linked to */dev/mt4* and */dev/nmt0*.

This is the syntax for filenames in the */dev/tape* directory:

format_<feature_...>drive#

where

format is as defined in Table 4-4.

feature is any of **blk**, **chr**, **dens**, **f512**, **swap**, or **norew**, as defined in Table 4-4.

drive# is the drive number, **0** for the first drive, **1** for the second, and so forth.

Table 4-4 shows the meanings for each string in the */dev/tape* filenames. Table 4-5 shows the links between */dev/tape* files and the older filenames in */dev*.

Table 4-4. Filename Strings for */dev/tape* Tape Device Files

String	Meaning
blk	Block mode. Data is written to and read from the tape in equal-sized blocks, defined by the tape device block size. Each block is a separate read or write operation for the tape drive. Every tape device must use either "blk" mode or "chr" mode.
chr	Character (variable length) mode. Data is written to and read from the tape in records of variable length. Each tape record begins with a definition of record length, allowing the tape drive to read or write the entire record in one operation. Every tape device must use either "chr" mode or "blk" mode.
dens	Variable density mode. Half-inch tape drives ordinarily operate at 1600 bits per inch (bpi). The "dens" mode takes advantage of optional 3200/6250 bpi capabilities in some tape drives. The desired density should be selected by a switch on the tape drive. (Note: the UNIX tape driver does not currently support the "dens" flag. The hardware switch on the drive itself is sufficient for selecting higher density operations.)
f512	The tape drive will transfer "raw" data, stripped of the usual header and format bytes. The block size on the tape is dependent upon the block size used by the individual tape controller.
swap	Byte-swapping mode. In this mode, the order of upper and lower bytes is reversed. This is for VAX compatibility.
norew	Non-rewinding mode. In this mode, the tape drive will not rewind at the completion of a "close" operation. The tape drive pointer will be positioned at the beginning of the next record on the tape.

Table 4-5. */dev/tape* Links to */dev* Filenames

/dev/tape Filename	*/dev* Filename
std_blk_0	mt0
std_chr_0	rmt0
std_blk_norew_0	nmt0
std_blk_norew_0	mt4
std_chr_norew_0	nrmt0
std_chr_norew_0	rmt4
std_blk_swap_0	smt0
std_chr_swap_0	srmt0
std_blk_swap_norew_0	snmt0
std_chr_swap_norew_0	snrmt0
std_blk_dens_0	mt8
std_chr_dens_0	rmt8
std_blk_dens_norew_0	mt12
std_blk_dens_norew_0	nmt8
std_chr_dens_norew_0	nrmt8
std_chr_dens_norew_0	rmt12

4.5 Changing Your Hardware Configuration

Table 4-6 shows the files to change for each hardware change you can make to the system. Use these general steps to change hardware:

1. Make software changes
2. Remake the kernel if necessary (see Section 4.2.3)
3. Make new special files in */dev* if necessary (see Section 4.6)
4. Shut down the system with the **shutdown** (8) command
5. Change hardware
6. Reboot the system with autoboot
7. Check for proper function
8. Note changes in "Appendix C: Configuration Worksheet"

Table 4-6. Files Affected by Hardware Change

Device	File	Change
Disk	*/sys/conf/*SYS_*DIST*	Add appropriate driver (Only one driver if several disks are the same type)
	*/sys/*SYS_*DIST/*	For user-written drivers
	*/dev/*devpart	One file for each partition
	*/dev/r*devpart	Raw file for each partition
	/etc/disktab	Definitions of disks
	/etc/fstab	Allocate partitions
	/etc/sptab	Allocate spanned-disk partitions
	*/sys/*SYS_*DIST/swapvmunix.c*	Add second swap area
	*/sys/*SYS_*DIST/spconfig.c*	Allocate spanned-disk partitions
Tape	*/sys/conf/*SYS_*DIST*	Add appropriate driver
	*/dev/tape/** /dev/mt?,	Files for various modes
	/dev/rmt?, */dev/smt?,*	(see **mtio** (4))
	/dev/srmt?, */dev/nmt?,*	Raw files for tape
	/dev/nrmt?, */dev/snmt?,*	
	/dev/snrmt?	
Terminals	*/sys/conf/*SYS_*DIST*	Add driver for controller (for example, for VME-ICP16/X)
	/etc/ttys	Define each terminal line
	/etc/termcap	Definitions of terminal types
	/dev/tty??	Special files for terminals
Ethernet	*/sys/conf/*SYS_*DIST*	Add appropriate driver
	/dev/pty?	Pseudo-terminals for network
	/dev/ttyp?	Pseudo-terminals for network
	/etc/rc	Start up network daemons
	/etc/rc.local	More network daemons
	/etc/inetd.conf	Internet daemons
ISFFP/VME-FFP	*/sys/conf/*SYS_*DIST*	Add SK0 to controllers
	/etc/rc.local	Start up */etc/fpinit*

4.6 Making New Devices in /dev

Make new files in */dev* under these conditions:

- Adding a disk or tape drive
- Changing model of disk or tape drive
- Adding a terminal controller
- Creating pseudo-terminals (pty)
- Creating graphics devices (ttyw)
- Creating devices for remote booting on Ethernet (enet)
- Adding a parallel printer

Make new device files with the *MAKEDEV* script in */dev*. Use the following Procedure to Make New Devices in */dev* to make new devices files.

Procedure to Make New Devices in ***/dev***

	ACTION	WHAT IT DOES
Step 1.	Enter the command cd /dev	Moves to the */dev* directory.
Step 2.	Use the ls(1) command to list the special files in this directory. Check that the special files you need (see Table 4-7) do not already exist.	Avoids redundancy. If the files you need are already there, stop this procedure now.
Step 3.	Enter the command MAKEDEV *nameN* . . . where *name* is the name of the device, from Table 4-7. *N* is the number of the device; usually 0 for the first one, 1 for the second device of the same type, and so on. SMD-type disks are numbered 0 for the first one, 2 for the second SMD-type device, 4 for the third SMD-type device, and so on. Note that there is no space between *name* and *N*.	Makes the required special files. Usually a single *MAKEDEV* command will generate several files. Table 4-7 shows the files generated by each command. Note that Table 4-7 combines devices from all ISI systems. You cannot make some of these devices on your system. To check which devices your system supports, read the comments in */dev/MAKEDEV*. For example, to make special files for two 106-Mbyte disks and one TS-11 tape drive on your Optimum V system, enter MAKEDEV sd0 sd1 ts0

End of procedure

Table 4-7. Arguments to the *MAKEDEV* Command

Device	Type	Name for MAKEDEV	Files Created
Standard devices	console	std	*console*
	drum		*drum*
	mem		*mem*
	null		*null*
	tty0		*tty0*
	tty		*tty*
	ttyw		*ttyw0, ttyw1*
Disks	SMD-type disks on Optimum	hp	*hp*[0-7][a-h], *rhp*[0-7][a-h]
	5¼-inch disks on Optimum	el	*el*[0-7][a-h], *rel*[0-7][a-h]
	SMD-type disks on Optimum V	sm	*sm*[0-7][a-h], *rsm*[0-7][a-h]
	5¼-inch disks on Optimum V	sd	*sd*[0-7][a-h], *rsd*[0-7][a-h]
Tape	TS-11 emulation	ts	*mt*[0,4,8,12], *rmt*[0,4,8,12], *nmt*[0,8], *nrmt*[0,8] non-swap devices, for half-inch tape drives only, begin with s: *smt*[0,4,8,12], *srmt*[0,4,8,12], *snmt*[0,8], *snrmt*[0,8] second unit increments numbers by 1: *mt*[1,5,9,13], *rmt*[1,5,9,13], *nmt*[1,9], *nrmt*[1,9] and so on
Terminal controllers	VME-ICP8/X	cp[0-3]	*tty*[h-k][0-f] (use only the first eight)
	VME-ICP16/X	cp[0-3]	*tty*[h-k][0-f]
	ISDH	dh	*tty*[h-k][0-f]
	ISDZ	dz	*tty*[0-7][0-7]
Graphics terminal	WorkStation	ttyw	*ttyw*[2-f], *font*
Pseudo-terminals	ptys	pty[0-2]	*pty*[p-r][0-f] *tty*[p-r][0-f]
Ethernet interfaces	Ethernet	enet0	*enet*[0-3]*a*
Parallel Printer	ISPP	lp	*lp*[0-7]
	VME-ICP16/8/X	cp	*cp*[0-3]

SECTION 5: SETTING UP YOUR NETWORK

UNIX 4.3BSD provides support for the Defense Advanced Research Projects Agency (DARPA) standard internet protocols IP, ICMP, TCP, and UDP. This section tells how to connect to networks using an Ethernet board.

If you are using a system shipped with an Ethernet board from Integrated Solutions, the system is already equipped for connection to a network. You must set up some network data bases, however, as described in Section 5.3.

This section does not discuss network connections involving telephone lines. For information on uucp connections over a modem, see Section 3.7, "Setting Up Uucp."

For information on the network for cluster nodes, see "Section 6: Setting Up Cluster Nodes." For information on the network for diskless nodes, see "Section 7: Setting Up Diskless Nodes."

This section contains the following subsections:

5.1 What Is a Network?

An Ethernet board in your system lets you connect to local area networks through coaxial cables that are attached to other Ethernet boards. These boards are also called ‘‘controller boards,’’ or ‘‘controllers.’’ Through these controllers and networks you can log in to other systems and execute limited commands there.

Electronic mail travels through the network, and your system can automatically forward mail if you set it up to do so (see Section 5.4).

Different networks use different ‘‘protocols,’’ formats for exchanging electronic information. When you connect a system to a network, you must set up the connection for that network's protocol. In each case, consult the manual page for the appropriate network in Section 4 of the *UNIX Programmer's Reference Manual* (for example, ip (4P) for Internet Protocol).

Using Integrated Solution's transparent remote file system (TRFS), you can access files and devices on selected remote systems as easily as on your own. With TRFS you need not log in to the other systems: all linked remote systems operate as though they are one big system. See Section 5.6 for a further discussion of TRFS.

5.2 Configuring the Network

Suggested reading:

- **intro** (4N)
- **inet** (4F)
- **ip** (4P)
- **tcp** (4P)
- **inetd** (8)
- **ifconfig** (8C)

Configuring for a network involves starting daemons and making sure that certain files are in place.

Use the following procedure, the Procedure to Configure the System for the Network, to configure the system for the network.

Procedure to Configure the System for the Network

	ACTION	WHAT IT DOES
Step 1.	Check the file */etc/rc* to check that it includes commands to start the following network daemons: • inetd (8C) • rwhod (8C)	Starts the network software. These processes run continually in the background. Each command should be of the form /etc/*netdaemon*;

Step 2.	Check the file */etc/rc.local* to see that it includes commands to start these network mail daemons:
	• sendmail (8)
	• routed (8C)
		Starts mail daemons at autoboot. These processes run continually in the background.
		The commands should be of the form
		if [-f /etc/routed]; then
		/etc/routed;
		fi
		and
		if [-f /usr/lib/sendmail]; then
		(cd /usr/spool/mqueue; rm -f lf*)
		/usr/lib/sendmail -bd -q1h &
		fi

Step 3.	Check the file */etc/inetd.conf* to see that it contains listings for these daemons: • comsat (8C) • fingerd (8C) • ftpd (8C) • hostnamed (8C) • rexecd (8C) • rlogind (8C) • rshd (8C) • telnetd (8C) • tftpd (8C)	Starts site-specific network daemons. See inetd (8) for the format of the */etc/inetd.conf* file.

(continued on next page)

Procedure to Configure the System for the Network *(continued)*

	ACTION	WHAT IT DOES
Step 4.	In */etc/rc.local*, check for a line near the top of the file, of this form: /etc/ifconfig *devN* inet 'hostname' all on one line, for each of your network controllers, where *dev* is the Ethernet controller: • nw for the VME-ECX controller • ex for the Excelan controller • il for the Interlan controller • tty for a serial line *N* is the controller number: • 0 for the first controller of a given type, 1 for the next, and so on • the port number (tty*xx*), for a serial line 'hostname' is the word "hostname" in back-quotes	Checks for network controller configuration. **admin** (8) creates this line in */etc/rc.local* during **admin** initialization. **ifconfig** (8C) enables the Ethernet controller and defines the network protocol. See **ifconfig** (8C) for the options, which change for each network. This example configures a host named "v24server" with a VME-ECX board, using the **arp** (4) protocols (default) and "trailer" encapsulation: /etc/ifconfig nw0 inet 'hostname' Note that with UNIX 4.3BSD, you no longer need to disable trailers under the **arp** protocols.
Step 5.	Check the */dev* directory for Ethernet files named *enet[0-3]a*. If these files do not already exist, create them with the commands cd /dev MAKEDEV enet0	Checks for Ethernet devices for remote booting. **admin** (8) creates these devices during **admin** (8) initialization. Do this only if you want to be able to boot other nodes over the network.
Step 6.	Use the procedure in Section 3.4.1, "Defining Terminal Lines," to define 16 lines labeled ttyp*[0-f]*.	Defines 16 pseudo-terminals of type "network." available for network functions. The daemons need these in order to operate with the network. You create the special files for these pseudo-terminal in the next step of this procedure. The */etc/ttys* entries should be ttyp*[0-f]* none network secure

(continued on next page)

Procedure to Configure the System for the Network *(continued)*

	ACTION	WHAT IT DOES
Step 7.	Check the */dev* directory for pseudo-terminal files named *ttyp[0-f]* and *ptyp[0-f]*. If these files do not already exist, create them with the commands cd /dev MAKEDEV pty0	Creates 16 master-slave pairs of files for pseudo-terminals. The master file is */dev/ptyp?* and the slave is */dev/ttyp?*. If you ever get error messages saying that you do not have enough ptys, create 16 more with the commands cd /dev MAKEDEV pty1 Remember to add appropriate entries in */etc/ttys*. If you need more than 32 ptys, you must change the kernel as described in "Section 4: Configuring Your System."
Step 8.	Check the file */sys/conf/*SYS_*DIST*, where "SYS" is one of • *V20* for the VME-68K20 • *V10* for the VME-68K10 • *Q10* for the IS-68K (Q-bus) to see that the Ethernet driver (nw, ex, or il) for your system is defined. Remake the kernel (see Section 4.2.3) if necessary, if you change this file.	Configures the kernel for Ethernet operations. If the necessary drivers are already in the kernel, this step is unnecessary.
Step 9.	Turn to "Appendix C: Configuration Worksheet" and note any changes.	Updates the reference sheet.

End of procedure

5.3 Setting Up Network Data Bases

Before you proceed with this section, you should have completed the system configuration outlined in the previous procedure. Your system must have an Ethernet board installed, the board must be connected to a transceiver, and all system configuration files must be updated.

Use the following procedure, the Procedure to Set Up Network Data Bases, to configure network data base files.

Procedure to Set Up Network Data Bases

	ACTION	WHAT IT DOES
Step 1.	Edit the files */etc/hosts* and */etc/networks* to add local nodes or networks to the distributed internet lists.	Modifies the data bases to include local networks that are not part of the internet. See hosts (5) and networks (5) for entry formats. Use other entries in */etc/hosts* and */etc/networks* as examples.
Step 2.	Edit the files */etc/hosts.equiv* and */.rhosts* to include the hostnames of "trusted hosts" on the network. Simply list the hostnames, one per line.	Identifies trusted hosts for your system. People from these nodes, who have accounts on your system as well, will be able to log in and execute commands through the network. See rshd (8C).
Step 3.	Edit the file */etc/ftpusers* to deny ftp access to named users. This file should always contain the accounts root and uucp.	Protects the system from unauthorized entry. ftp involves setting up a public access area for anonymous logins. See ftpd (8C) and Section 5.5, "Setting Up ftp" for advice on how to set up this area with restricted access.

End of procedure

5.4 Routing and Gateways

When connecting to non-local networks, you must set up routing information so that the "packets" of information will travel correctly. A non-local network in this context means a connection between two servers, such as an Ethernet connection between a server for a cluster system and a server for a diskless system. The connection between the cluster or diskless nodes and their respective servers is considered a local network.

In addition, your node may serve as a "gateway," a conduit between two separate networks. A cluster server that communicates to a larger network is the gateway between the cluster network and the larger network. A gateway node has some special configuration considerations.

5.4.1 Routing

UNIX can use two methods for routing between non-local networks:

- routed (8C). the routing daemon, can maintain the system routing tables. routed maintains up-to-date routing tables in a group of local area networks. Using the */etc/gateways* file, routed can also initialize static routes to distant networks.

  When routed starts up (usually from */etc/rc.local*), it reads the file */etc/gateways* and installs the routes defined there, then routed checks all local networks to see if they are already running a routed. If your routed finds another local routed, the two cooperate in maintaining a globally consistent view of routing in the local environment. This view can extend to include remote sites also running a routed (see routed (8C)).

- You can define a "wildcard" route to a smart gateway and depend on the gateway to provide ICMP routing information which dynamically creates a routing data base. Define the wildcard route by adding an entry of the form

  ```
  /etc/route add 0 smart-gateway 1
  ```

  to */etc/rc.local* . (See route (8C) for more information.)

The system will use the wildcard route as a last resort in routing packets to their destinations. If the smart gateway successfully generates a routing message, the system then adds that route to the routing tables.

This approach has some advantages over **routed** in that it dynamically creates a routing table entry; however, the routing table entry is not dynamically modified if an alternate route for the same destination is available. Because of this, should a smart gateway fail there is no way to maintain routing service, except for manual alteration of the routing table entry.

In an environment that contains only bridges, which do not generate routing redirect messages, **route** is not suitable.

The system always listens to, and processes, routing table redirect information, so it is possible to combine both of the above facilities. You might use **routed** to maintain information about routes to geographically local networks, while using the wildcard routing techniques for distant networks.

netstat (1) displays routing table contents as well as various routing statistics. For example,

```
netstat -r
```

displays the contents of the routing tables, while

```
netstat -s
```

shows the number of routing table entries created by the wildcard routing method described above.

5.4.2 Gateways

Gateway nodes handle a higher volume and wider variety of network operations than a simple network node. A special kernel option, GATEWAY, configures a node to be used as a gateway. This option increases the size of the routing hash tables in the kernel. Unless configured with that option, nodes with only a single non-loopback interface never attempt to forward packets or to respond with ICMP error messages to misdirected packets.

This change reduces the problems that may occur when different nodes on a network disagree as to the network number or broadcast address. UNIX 4.3BSD machines that forward packets back through the same interface on which they arrived will send ICMP redirects to the source host if it is on the same network. This improves the interaction of gateway nodes with nodes that configure their routes via default gateways and redirects. The generation of redirects may be disabled with the configuration option IPSENDREDIRECTS=0 in environments where it may cause difficulties.

Local area routing within a group of interconnected Ethernets and other such networks may be handled by **routed** (8C). Gateways between the Arpanet or Milnet and one or more local networks require an additional routing protocol, the Exterior Gateway Protocol (EGP), to inform the core gateways of their presence and to acquire routing information from the core. If necessary, contact ISI Customer Support for assistance.

5.5 Setting Up ftp

The **ftpd** (8C) server supports an anonymous "ftp" ("file transfer protocol") account. Because of the inherent security problems with such a facility, you should read this section carefully if you consider providing such a service.

Enable the anonymous account by creating a user ftp. When a client uses the anonymous account, a **chroot** (2) system call by the server restricts the client from moving outside that part of the file system where the user ftp home directory is located. Because of the **chroot** call, you must install in the ftp home directory certain programs and files used by the server process. Further, you must make certain that all directories and executable images are not writable.

The following procedure for directory setup preserves security:

```
# cd ~ftp
# chmod 555 .; chown ftp .; chgrp ftp .
# mkdir bin etc pub
# chmod 555 bin etc; chown root bin etc
# chmod 777 pub; chown ftp pub
# cd bin
# cp /bin/sh /bin/ls .
# chmod 111 sh ls
# cd ../etc
# cp /etc/passwd /etc/group .
# chmod 444 passwd group
```

When local users wish to place files in the anonymous area, they must be placed in a subdirectory. In the setup here, the subdirectory is *~ftp/pub.*

Another issue to consider is the copy of */etc/passwd* placed here. Users who use the anonymous account have read access to this file. They may then try to break the passwords of users on your node for further access. A good choice of users to include in this copy of */etc/passwd* might be **root, daemon, uucp,** and the ftp user. All passwords here should probably be "*".

Aside from the problems of directory modes and such, the ftp server may provide a loophole for interlopers if certain user accounts are allowed. The file */etc/ftpusers* is checked on each connection. If */etc/ftpusers* contains the requested user name, the request for service is denied. This file should normally include these accounts:

```
uucp
root
```

Accounts with nonstandard shells should be listed in this file. Accounts without passwords need not be listed in this file; the ftp server will not service these users.

5.6 Using TRFS (Transparent Remote File System)

The transparent remote file system (TRFS) allows you to access files on remote nodes as easily as on your own node.

To use TRFS, put the string */@system-name* before the pathname of files, where *system-name* is the hostname of the node you want to access. For example, to see the contents of the */usr* directory on the node linksystem, you would enter

```
ls /@linksystem/usr
```

TRFS is a *transparent* operation. Most UNIX commands function normally with TRFS arguments; you need not change any existing programs or commands when using remote files through TRFS.

All UNIX file protections still apply under TRFS. Assign permissions with **chmod** (1) as with local files.

TRFS uses a special protocol designed by Integrated Solutions. Enable TRFS by including TRFS as an option in the kernel configuration file (*/sys/conf/*SYS*_DIST*).

TRFS requires login accounts with the same user identifications on all linked nodes. For this reason, the */etc/passwd* file should be the same for all nodes on a TRFS link. There are two different ways of ensuring this:

1. Always use admin (8) when adding or changing user accounts, and define the admin password administration mode as "TCP/IP." See Section 3.3, "Initializing Your System." This means that all changes to */etc/passwd* affect all other nodes under the common password administration domain.

2. Link all of the password files over TRFS with the command

   ln -s */@centralhost/*etc/passwd /etc/passwd

   where *centralhost* is the main node for administration. When */etc/passwd* changes, it will change on all nodes on the TRFS link.

   CAUTION

   > When you link password files in this manner, you can no longer use **vipw** (8) to edit */etc/passwd.* Use **admin** (8), or use **vi** (1) directly. When you use **vi** (1), the system should be in single-user mode; otherwise you can wreak havoc with the password file.

Use the method from item (2), above, to link other commonly shared files and directories between nodes. For example, you can maintain a central library of programs on one node and link all others to that library. This way, when users access files in a linked directory, they do not have to know that it is on a remote node; the commands are the same as if the files were local.

Linking directories and files in this manner carries two major advantages:

- Saves disk space. The file resides on only one node.
- Makes network-wide updates easy. When you change a linked file, it changes for all nodes on the TRFS link.

5.7 Troubleshooting the Network

When you have a network, you will occasionally experience network problems. In many cases the cause of network problems is simply a loose electrical connection to the network. On networks such as the Ethernet a loose cable tap or misplaced power cable can result in severely deteriorated service. The **netstat** (1) program can help track down hardware malfunctions. In particular, look at the *-i* and *-s* options in the manual page.

After you carefully check the physical connections to the network, look at the software.

Check that the appropriate daemons are running in the background. Section 5.2 lists the daemon processes that should be running.

Check */dev* to see that these files exist:

/dev/ptyp[0-f]
/dev/ttyp[0-f]

Check the file */etc/ttys* for proper network definitions, as described in Section 5.2.

If you have checked the network connections, the daemon processes, and the */dev* files, and the network still does not operate, the problem should be referred to a qualified technician. ISI Customer Support can provide assistance.

Debug tools exist to help track down network problems. Most servers on the UNIX system accept a *-d* option, which forces all sockets to be created with debugging turned on. The SO_DEBUG option may be supplied before establishing a connection on a socket, in which case the system will trace all traffic and internal actions (such as timers expiring) in a circular trace buffer. You can then print this buffer with the **trpt** (8C) command. Consult the appropriate manual pages for more information.

SECTION 6: SETTING UP CLUSTER NODES

This section gives procedures to set up and operate a cluster node configuration. These special configurations require

- A server node (an Optimum V16 or Optimum V24 System/WorkStation)
- One or more Cluster Nodes (a set of boards with associated cables)
- Identical UNIX release levels on the server and cluster nodes
- Graphics software installed on the server node (if any cluster node is a graphics WorkStation)

The *Optimum V Cluster Node Installation/Configuration Manual* gives details on how to set up the hardware for a cluster node.

This section contains the following subsections:

6.1 What Is a Cluster Configuration?

Section 1.2, "Introduction to Hardware," describes the physical structure of a cluster configuration. For system administration purposes, a cluster configuration is several nodes (individual systems) that share common disk resources.

A server node operates as any normal Optimum V System or WorkStation. The server node provides the shared disk resource. The operating system is the same UNIX 4.3BSD used for other Systems and WorkStations. The kernel must contain the VB (VMEbus backplane) driver, the RD (remote disk) driver, the TRFS (Transparent Remote File System) option, and the ENETFILTER option.

Each of the cluster nodes uses files on the server node. Each cluster node has its own CPU, possibly booted from the same UNIX kernel as the server node. The server node and its associated cluster nodes share a common disk and a common file system. Any file that can be accessed from the server node can also be accessed from any of the cluster nodes, subject to standard UNIX protection mechanisms.

6.2 Directory Links for Cluster Nodes

All cluster nodes and the server node share a common file system. When you access a file from one node, that is the same as accessing that file from any other node. However, some files are unique to each cluster node, and to the server node itself.

The unique, node-specific files fall into two categories:

- Configuration or data base files that contain node-specific information
- Temporary files and log files, where two or more processes running under different kernels might try to write simultaneously

The first case includes files such as */etc/ttys*, which usually are different for each node on the network. The second case includes the */tmp* and */usr/tmp* directories, where two processes on different kernels specify the same name for a temporary file.,

For example, programs like sort(1) create temporary files to hold the sorted information. These files are created in the */usr/tmp* directory and have a name of the form */usr/tmp/stm$$*, where *$$* is expanded to represent the current process ID.

In single-processor systems, attaching the process ID to the temporary file name prevents collisions between concurrent sorts because all process IDs are unique. On a cluster network, two processes with the same ID might exist, which would create one temporary file rather than two separate temporary files. Therefore, separate processors must have separate temporary areas.

The UNIX kernel solves this problem by providing naming conventions for node-specific files and directories. The naming convention is based on the node hostname. The kernel provides unique names by expanding, or translating certain string variables when it evaluates the targets of symbolic links:

- $HOST translates to the hostname of the node from which the request originates, server or remote.
- $RHOST translates to the hostname of the remote node from which the request originates. If the request originates on the server node, $RHOST translates to a null string.
- $REMOTE translates to the string remote when interpreted on a remote node, and to a null string on the server node.
- $MACHINE translates to one of the following:
  - — Q10 for an Optimum (Q-bus) system
  - — V10 for an Optimum V system with a VME-68K10 CPU board
  - — V20 for an Optimum V system with a VME-68K20 CPU board

As an example, consider a symbolic link to the file *test.$RHOST*. When a command is issued on the server node that refers to this symbolic link, it tries to access a file "*test.*". A command issued from a cluster node named cws1 tries to access a file "*test.cws1*".

Through symbolic links, this process is transparent to the user. A file or directory name such as */etc/ttys* is actually a symbolic link to a file or directory name which has one of the string variables embedded. The kernel expands the string and accesses the correct file for the requesting node.

For example, the following command links the directory */tmp* to a directory name with the embedded string:

```
ln -s /.tmp.\$RHOST /tmp
```

(Note the backslash (\) precedes the dollar sign ($), which "escapes," or prevents shell interpretation.) On a network with a server node named central and cluster workstation nodes cws1 and cws2, a pathname that includes */tmp* will translate to

- (on central) */.tmp.*
- (on cws1) */.tmp.cws1*
- (on cws2) */.tmp.cws2*

Each of these */.tmp* directories then contains files specific to its related node. The user simply types /tmp in pathnames; the kernel handles the interpretation invisibly.

Specifying a file by its real name (for example, */etc/.tmp.cws1*) will always access that particular file.

When the **admin** (8) program configures a cluster network it makes node-specific links for these files and directories:

- */dev*
- */etc/fstab*
- */etc/motd*
- */etc/psdatabase*
- */etc/rc*
- */etc/rc.local*
- */etc/ttys*
- */etc/utmp*
- */tmp*
- */usr/adm*
- */usr/lib/crontab*
- */usr/lib/mail*
- */usr/spool/mqueue*
- */usr/spool/rwho*
- */usr/tmp*
- */vmunix*

Check your own UNIX installation for local programs that use temporary or log files. If any programs use temporary or log files in areas other than those listed above, and these programs can run on more than one processor simultaneously, you must create a symbolic link either for the files or for the directory in which they reside.

When you create node-specific symbolic links, you might want to make the "real" file or directory names begin with a period:

```
/tmp --> /.tmp.SRHOST
```

This way, the casual user will see only the link and not be confused by a profusion of names like

/tmp.cws1
/tmp.cws2
/tmp.cws3 . . .

since the ls (1) command does not normally show filenames that begin with a period (.).

6.3 Configuring Cluster Nodes

Configuring a cluster network involves four tasks:

1. Arranging for swap space.
2. Choosing a network address for the server node, which should already be done as shown in Section 3.3, "Initializing Your System."
3. Choosing a name for each cluster node.
4. Configuring the network with **admin** (8).

6.3.1 Allowing Swap Space for Cluster Nodes

Cluster nodes require swap space on the server node. The server node uses assigned disk partitions for swap space; the cluster nodes, having no disks of their own, use files for swap space.

Cluster nodes use a file */usr/spool/diskless/*name*.swap* on the server node, where "name" is the hostname of the cluster node. **admin** (8) creates the swap directory when configuring cluster nodes.

These files can be very large, up to 16.5 Mbytes apiece (16,500 Kbytes, as measured by the **ls** (1) and **df** (1) commands). When you configure the file systems for your network, you must allow enough space for these files in the */usr* partition.

Normally, the files are not that large. To see the actual size of the files, enter the command

```
ls -s /usr/spool/diskless/*.swap
```

The -s option to ls shows the size in blocks of the files. Do not use the -l option to look at file size; this gives a character count, an inaccurate measure in this case.

Enter the command

```
df
```

to see current disk usage. Check the "available" (avail) column for the */usr* file system. There should be at least 5000 Kbytes available for each cluster node on the network. Section 4.3, "Configuring Disks," tells you how to change disk configuration if you need to assign a larger partition for swap files. You can also create a "spanned disk" and use partitions from several disks to create a very large (or simply more efficient) file system for the swap file area.

If the */usr* file system is not large enough to hold the swap files, you can either use a symbolic link to put the swap files in another file system or create a spanned disk to enlarge the file system. For example, the command

```
ln -s /diskless/usr/spool/diskless /usr/spool/diskless
```

points to a directory in another area, a */diskless* file system that you can create with the **newfs** (8) command as described in Section 4.3.4, "Adding a New File System." Mount (**mount** (8)) this file system on a sufficiently large disk partition.

6.3.2 Choosing Names and Addresses for a Cluster Network

When you configure a cluster network, you assign a network address for the server node; **admin** (8) automatically creates addresses for cluster nodes based on this server address.

If you have not already done so, complete the procedure in Section 3.3, "Initializing Your System." This procedure assigns a name and internet address for the server node. You must complete this naming procedure before installing cluster or diskless nodes.

Choose hostnames for each of the cluster nodes and write them down. Use hostnames with less than twelve characters. Do not use spaces within hostnames; use underscores to separate words (for example, cn_name). The hostnames must be unique within the network.

In "Appendix C: Configuration Worksheet," in the area reserved for networks, enter this information:

- Server node's hostname
- Hostname of each cluster node

The procedure in Section 6.3.3, "Configuring the Network With **admin** (8)," will assign network addresses for the cluster nodes. Enter this information in Appendix C when you run the procedure.

6.3.3 Configuring the Network With admin (8)

Use the following Procedure to Add a Cluster Node to the Network to configure the cluster network or to add a cluster node to an existing network.

Procedure to Add a Cluster Node to the Network

	ACTION	WHAT IT DOES
Step 1.	If you have not yet done so, configure the server node with **admin** (8).	Sets up the server for cluster operations. See Section 3.3, "Initializing Your System."
Step 2.	Enter the command **/etc/admin**	Invokes the administration program.
Step 3.	Select option C at the top menu of the **admin** (8) program.	Selects operations dealing with local (diskless or cluster) networks.
Step 4.	Select option B from the next menu.	Selects adding a cluster node to the network. **admin** now prompts for the name of the cluster node, and whether or not this is a graphics node. See Table 6-1 for further details on **admin** operations.

End of procedure

eng gau16

y

Controller # 9 u60

89,0,0,200

u61

89.0.0.201

ex0

89,0.0,60

Table 6-1. **admin** Adding a Cluster Node

What **admin** (8) Does	Affected Files
Prints the top-level menu. Select item C, "Diskless/Cluster installation and deletion."	—
Prints the diskless/cluster menu. Select item B, "Add a cluster node."	—
Prompts for the name of the cluster node. Enter a name, with the same restrictions on length and characters as for the server node name.	—
Prompts for graphics configuration of the cluster node. If this is a Cluster WorkStation (graphics), answer y; if this is a Cluster Compute Node (non-graphics), answer **n**.	—
Prints the cluster node name and internet address, formatted as for the file */etc/hosts*. You should write this information in "Appendix C: Configuration Worksheet." **admin** chooses an internet address for this cluster node. **admin** chooses a unique address, based on the internet address for the vb0 VMEbus backplane network controller on the server node. If you accept the information printed by **admin**, press <return> to continue the procedure. If you want to abort admin, press ^C (CTRL-C) to prevent any changes to system files.	—
Writes the cluster node name and internet address to */etc/hosts*.	*/etc/hosts*
Creates the swap directory */usr/spool/diskless*, if it does not already exist.	*/usr/spool/diskless*
Creates various "hidden" files and directories for the cluster node. See Section 6.2, "Directory Links for Cluster Nodes."	*/dev* */etc/fstab* */etc/motd* */etc/psdatabase* */etc/rc* */etc/rc.local* */etc/ttys* */etc/utmp* */tmp* */usr/adm* */usr/lib/crontab* */usr/lib/mail* */usr/spool/mqueue* */usr/spool/rwho* */usr/tmp* */vmunix*

6.4 Booting Cluster Nodes

Boot the server node as any other Optimum V System or WorkStation using the procedures in "Section 2: Starting UNIX from Disk." Once the server node is running, boot the cluster nodes one at a time.

Booting a Cluster Workstation (a graphics node) requires graphics software on the server node. If the server is a WorkStation, this software is already in place. Graphics software is on the standard ISI UNIX 4.2/3BSD release tape from Release 3.05 onward. Contact ISI Customer Support if you do not have the necessary graphics software.

Use the following procedure, the Procedure to Boot a Cluster Node, to boot a cluster node. Figure 6-1 shows a sample of a normal cluster node boot.

Procedure to Boot a Cluster Node

	ACTION	WHAT IT DOES
Step 1.	Boot the server node, if it is not yet running. Continue the boot to multiuser UNIX.	You cannot boot cluster nodes without a functioning server node. Booting the server node also provides power to the cluster nodes. Use the procedures in "Section 2: Starting UNIX from Disk" to boot the server node.
Step 2.	Check that the server node has files of the name */dev/enet*[0-3]*a*	Checks for */dev* files to support remote booting. These files must exist in the */dev* directory. If they are not there, create them with the procedure in Section 4.6, "Making New Devices in */dev*."
Step 3.	Enter the command `ps -aux | grep bootd`	Checks that the boot daemon, */etc/bootd*, is running in the background. If it is not, enter the command /etc/bootd & and check that the file /etc/rc.local to see that it has a command to start the daemon.
Step 4.	Enter the command ps -aux | grep '*@-name*' where *name* is the hostname of the cluster node you want to boot, and '*@-name*' is enclosed by single quotes (apostrophes).	Checks for leftover processes from the last boot of this cluster node. You cannot boot if these processes exist. See Section 6.5, "Shutting Down Cluster Nodes," to remove these processes. These processes occur on server nodes that service requests from cluster nodes. They execute entirely in kernel mode, with no associated user.
Step 5.	Turn on power for the cluster node's console (or monitor if the cluster node is a WorkStation).	Powers up the console (monitor).
Step 6.	Press RETURN on the console keyboard. Wait for the PROM prompt (:).	Displays the PROM prompt for the cluster node. You can now enter commands to the cluster node's CPU PROMS.

(continued on next page)

Procedure to Boot a Cluster Node *(continued)*

	ACTION	WHAT IT DOES
Step 7.	Enter the character "@" (SHIFT-2 on most terminals) and press RETURN.	Boots the cluster node through the VMEbus backplane driver (vb) from the server node's disk. After a successful boot, the console shows the single-user UNIX prompt (#). If the "@" boot does not work, you can boot manually from the PROMs. Enter the command vb(0,0)*server:kernel* HOST=*hostname* SERVER=*server* all on one line, where *server* is the hostname of the server node. Note that this appears twice in the boot string. *kernel* is the name of the kernel for booting. This is usually /vmunix. *hostname* is the hostname of this cluster node. For example, this command boots a cluster node named cws1 from a server named v24: vb(0,0)v24:/vmunix HOST=cws1 SERVER=v24
Step 8.	Enter the command date *yymmddhhmm* where *yymmddhhmm* is today's date.	Tells UNIX what time it is. See date(1) for details on the entry format.
Step 9.	Enter a ^D (CTRL-D; hold down the CTRL key and press D).	Boots to multiuser UNIX using the shell script */etc/.rc.local.name*, where *name* is the hostname of the cluster node. After displaying several messages, the monitor shows this prompt: IS68K 4.3 BSD (*hostname*) login: where *hostname* is the hostname of this node. This is the UNIX login prompt.
Step 10.	Log in as the superuser by entering root as your login name. The root password is the same for all nodes on the local cluster network: you must use the same password as for the server node.	Logs in to UNIX, giving access to a working C shell (UNIX command processor). You can now issue UNIX commands to the shell prompt (#). Note that a Cluster WorkStation (graphics node) requires graphics software on the server node. See the comment at the beginning of this subsection.
	End of procedure	

You cannot turn off power to a cluster node without also turning off power to the server node, and vice versa. Therefore, if you want to leave the server node operating, you can only bring the cluster node down to a PROM prompt and turn off the console (or WorkStation monitor).

The shutdown procedure differs depending on your current condition:

- If you are running multiuser UNIX, begin with *Step 1* in the following procedure.
- If you are running single-user UNIX (the command line shows the single-user prompt (#)), skip to *Step 2* of the following procedure.
- If the cluster node has crashed, skip to *Step 4* of the following procedure before rebooting.

Use the following procedure, the Procedure to Shut Down a Cluster Node, to shut down a cluster node.

Procedure to Shut Down a Cluster Node

	ACTION	WHAT IT DOES
Step 1.	Enter this command: shutdown +*n* where *n* is a time in minutes.	Notifies users of shutdown, waits for the specified time, and then kills all processes. When shutdown is complete, the monitor shows the single-user prompt (#). CAUTION If you turn off your system at this point, you risk losing data. You must finish all steps of this procedure.
Step 2.	Enter the command reboot	Updates the superblock and brings the cluster node to the PROM prompt (:). Wait until the PROM prompt (:) appears.
Step 3.	Turn OFF power for the cluster console (or WorkStation monitor).	Powers down the console for this node. The cluster node is now "off," though the CPU is still active.
Step 4.	At the server node, enter the command ps -aux | grep '@-*name*' where *name* is the hostname of the cluster node you just shut down, and '@-*name*' is enclosed by single quotes (apostrophes).	Displays all processes running on the server node that are associated with the cluster node. These processes are now useless, and will interfere with any efforts to reboot the cluster node. If this command shows no processes on the server node you are now finished with this procedure.
Step 5.	Enter the command killpg -9 *pid* where *pid* is any one of the process IDs displayed in *Step 4*.	Kills the named process and all other processes in the process group. See killpg (8) for more details on this command.

End of procedure

```
Initialize all memory . .

Cluster Node 1

(Integrated Solutions logo appears here)

Integrated Solutions
VMEBUS 68020 4.4/1 boot:Wed Apr 22 19:15:44 PDT 1987
: @
327872+39808+108684 start 0x800

Integrated Solutions UNIX Release 4.0 #1 (root@UNKNOWN) Wed Apr 1 16:58:26 PST 1987
real      memory = 2.00M
available memory = 1.11M
detached  memory = 0.12M at 0x1c0000
using 90 buffers containing 0.17M of memory

 VB0  at address 0xf00000/074000000   ** no vector
    vb0    at VB0  slave 0        (60.123.0.0.0.0) 0.50M @ 0xf00000

 root on rd0a, dump on rd0b, args on rd0b, swap on rd1e

16.67 MC68020 CPU
Parity Enabled
*** WARNING: should run interleaved swap with >= 2 Mb ***
#
```

Figure 6-1. Sample of a Normal Cluster Node Boot

6.5 Shutting Down Cluster Nodes

To shut down a cluster network, first shut down the individual cluster nodes, then shut down the server node with the procedure in Section 2.4, "Shutting Down the System."

CAUTION

Failure to shut down the system in an orderly fashion may result in lost data. In particular, simply turning off power without an orderly shutdown will probably result in data loss.

Table 6-3. Moving Between UNIX Modes on Cluster Nodes

Want:	PROM	Single-User UNIX	Multiuser UNIX *system-name* #
Now:	:	#	(or $, %, login:)
PROM :		Enter the character "@", as described in Section 6.4, "Booting Cluster Nodes." If this does not work, use the command vb(0,0)*server*:*kernel* HOST=*hostname* SERVER=*server* all on one line, as described in the same section.	Follow the procedure in Section 6.4, "Booting Cluster Nodes."
Single-User UNIX #	Enter # reboot Wait for the PROM prompt to appear. CAUTION Pushing RESET crashes all nodes in the cabinet down to their PROM prompts, but this can write bad data to the server node's disks, even with a sync(8) first. Always run fsck(8) on the server node after a RESET.		Enter a ^D (CTRL-D; hold down the CTRL key and press D) to force a quick switch to multiuser. For a full reboot, enter the command: reboot and boot from the PROM prompt as described above in this table.
Multiuser UNIX *system-name* # or $ or % or login:	Use the procedure in Section 6.5, "Shutting Down Cluster Nodes," until the PROM prompt appears (after *Step* 2). See CAUTION above.	Use the shutdown(8) command as described in Section 6.5, "Shutting Down Cluster Nodes." This informs others of the shutdown, sends warning messages, then boots down to single-user.	

6.6 Booting Single-User on Cluster Nodes

Cluster nodes use the same UNIX modes as normal Systems or WorkStations. Boot the cluster node to multiuser UNIX for normal operation. Other modes are useful for

- Issuing PROM commands (Section 6.6.1)
- Running diagnostics (Section 6.6.2)

Table 6-2 shows the UNIX modes and their purposes.

Table 6-2. UNIX Operation Modes for Cluster Nodes

Prompt	Mode	Purpose
:	PROM	Communicating with the bootstrap PROMs on the CPU (see Section 1.2) for • booting UNIX (Section 6.4) • using PROM commands (Section 6.6.1) • stand-alone diagnostics (Section 6.6.2)
#	Single-user UNIX	Issuing commands as root with no other users logged in. Useful for changing hardware configurations.
system-name # or $ or % or login:	Multiuser UNIX	Multiuser UNIX operation. This is the normal mode for the system; normal booting tries to reach this level.

Table 6-3 shows how to move from any mode to any other mode. Find the mode you are in and read across to the column for the mode you want.

6.6.1 Using Commands at the Cluster Node PROM Prompt

The PROM prompt (:) gives access to the CPU PROMs. At this prompt you can use the commands in Table 6-4.

Table 6-4. Commands at the Cluster Node PROM Prompt

Command	What It Does
@	Boots the cluster node to single-user UNIX, using the kernel */.vmunix.$HOST* on the server node.
!	Toggles interrupt enabling by the monitor. Entering ! enables interrupts; entering ! again disables them.

vb(0,*m*)*source:filename*	Loads and starts execution of a file on a device, where
	vb is the VMEbus backplane driver, which actually accesses the boot disk on the server node. Commands at a cluster node PROM prompt can access only the boot disk.
	m is the device "minor" number, the partition number on the boot disk.
	source is the hostname of the node where the desired kernel resides.
	filename is the path to the file you want to execute on disk.
	For example, you could issue this command:
	`vb(0,0)v24server:stand/V20/mmu`
	to execute the file */stand/V20/mmu* on the *a* (first, numbered from zero) partition of the boot disk on a server node named v24server.
	If you want to go directly to a block instead of a partition, you can specify a number followed by "b". For example,
	`sd(0,10b)`
	In some applications, it may be desirable to load a program without automatically beginning its execution. Preceding the load string with "<" causes the file to be loaded and a starting address to be printed. Then control is returned to the PROM monitor. The program can then be started by typing a ">" followed by the specified starting address.

(continued on next page)

Table 6-4. Commands at the Cluster Node PROM Prompt *(continued)*

Command	What It Does
% address [value]	Opens/modifies location as a byte where *address* is the hexadecimal address for that memory location and *[value]* is the optional new hexadecimal value to write at address.
$ address [value]	Opens/modifies location as a word where *address* is the hexadecimal address for that memory location and *[value]* is the optional new hexadecimal value to write at address.
# address [value]	Opens/modifies location as a longword where *address* is the hexadecimal address for that memory location and *[value]* is the optional new hexadecimal value to write at address. You can use these commands in two ways: by entering the address in hex $fff520 or by entering the address in hex with a new value for that address $fff520 123456 If you specify the value, the location is written to without ever being read. When you type % and the hexadecimal address of a memory location, the system appends a vertical bar (|) followed by the current byte value at that address and waits for you to enter the hexadecimal address of the new location. When you type S and the hexadecimal address of a memory location, the system appends a colon (:) followed by the current word value at that address and waits for you to enter the hexadecimal address of the new location. When you type # and the hexadecimal address of a memory location, the system appends a semi-colon (;) followed by the current word value at that address and waits for you to enter the hexadecimal address of the new location. For example, entering $200 might display 200:68AC _ where "_" represents the cursor location. If you do not type in a new value and you press RETURN, the location remains unmodified.

(continued on next page)

Table 6-4. Commands at the Cluster Node PROM Prompt *(continued)*

Command	What It Does
	If you type in a new hex value following the displayed hex value 200:68AC 68AF and press RETURN, the new value is written to the current location, and the colon (:) prompt appears. If you type in a new hex value following the displayed value 200:68AC 68AF and press LINE FEED, the new value is written to the current location and the next location is opened, rather than returning to the PROM monitor prompt (:). If you type in a new hex value following the displayed value and follow it with a backslash (\\), 200:68AC 68AF \\ the new value is written to the current location and the previous location is opened, rather than returning to the PROM monitor prompt (:). Opening a non-existent memory location with interrupts enabled will cause the system to continuously attempt to access the non-existent location, causing trap errors.
> *address*	Begins execution at a hex address. If you enter the "greater than" sign (>) and an address in hex, execution immediately begins at that address. For example, entering >68AC begins execution at 68AC.
~	Puts the CPU into transparent mode. Characters from the console are transmitted to the second serial port and characters from the second serial port are directed to the console port.
~.	Exits transparent mode and returns you to the PROM monitor prompt (:).
~~	Sends one tilde (~) to the transparent side.

6.6.2 Running Diagnostics on Cluster Nodes

Two hardware diagnostic programs are useful for cluster nodes:

1. **mmu** to test the Memory Management Unit on the CPU
2. mem to test main memory

Use these diagnostic programs only if you suspect hardware failures. For example, run diagnostics in response to these symptoms:

- Cluster node will not boot
- Cluster node crashes frequently

Since cluster nodes cannot execute programs from a tape drive, you must install the diagnostic programs on the server node's boot disk. See Section B.2.2, "Running Diagnostics From UNIX," for instructions to install diagnostic programs on the server node.

Use the following Procedure to Load Diagnostics on a Cluster Node to run hardware diagnostics.

Procedure to Load Diagnostics on a Cluster Node

	ACTION	WHAT IT DOES
Step 1.	Check that the monitor displays the PROM prompt (:).	Checks for the right mode for diagnostics.
Step 2.	Enter this command: vb(0,0)*server*:stand/*SYS*/*test* where vb is the VMEbus backplane driver, which actually accesses the boot disk on the server node. *server* is the hostname of the server node *SYS* is your system type (CPU type): • V20 for the VME-68K20 • V10 for the VME-68K10 • Q10 for the IS-68K (Q-bus) *test* is the diagnostic test, either • mmu or • mem	Loads the program and begins execution. For example, to execute the **mmu** test enter **vb(0,0)v24server:stand/V20/mmu** where **v24server** is the hostname of the server node. This cluster node has a VME-68K20 CPU board. The (0,0) refers to the first partition of the first disk (numbered from 0, remember), also known as partition *a*, holding the root (/) file system. The console should now display this message: **Type RETURN to start at 0x*wwww*** If the command does not work, check that the directory */stand* on the server node contains diagnostic programs. If the directories */stand/V20*, */stand/V10*, and */stand/Q10* do not exist, you must install them from the diagnostic tape. See Section B.2.2, "Running Diagnostics From UNIX."
Step 3.	Press RETURN.	Starts the test.
Step 4.	Follow the instructions in "Appendix B: Using Diagnostic Programs" to execute the diagnostic test. See either Section B.3, "Using mmu" or Section B.4, "Using mem"	Runs the diagnostic test. The tests themselves are identical in procedure to those run on the server node or any other node.

End of procedure

SECTION 7: SETTING UP DISKLESS NODES

This section gives procedures to set up and operate a diskless node configuration. These special configurations require

- At least one server system (any Optimum System, Optimum V System, or Optimum V WorkStation) with an Integrated Solutions VME-ECX Ethernet controller or an Excelan Ethernet controller (Interlan will not work)
- One or more Diskless Nodes (special systems with an Ethernet controller and no disk drives)
- Identical UNIX release levels on the server and diskless nodes
- Graphics software installed on the server node (if any diskless node is a graphics WorkStation)

The *Optimum V Diskless Node Installation/Configuration Manual* gives details on how to set up the hardware of a diskless node.

This section contains the following subsections:

7.1 What Is a Diskless Configuration?

Section 1.2, "Introduction to Hardware," describes the physical structure of a diskless configuration. For system administration purposes, a diskless configuration is several nodes (individual systems) sharing common disk resources.

A server node operates as any normal Optimum or Optimum V system. The operating system is the same UNIX 4.3BSD used for other Systems and WorkStations. The kernel must contain the NW (Integrated Solutions Ethernet) or EX (Excelan Ethernet) driver, the RD (remote disk) driver, the TRFS (Transparent Remote File System) option, and the ENETFILTER option.

Each of the diskless nodes uses files on the server node. Each diskless node has its own CPU, booted from a UNIX kernel on the server node. The server node and its associated diskless nodes share a common disk and a common file system. Any file that can be accessed from the server node can also be accessed from any of the diskless nodes, subject to standard UNIX protection mechanisms.

A single network can accommodate several server nodes. The boot procedure in Section 7.4, "Booting Diskless Nodes," describes how to specify a particular server node when booting a diskless node.

7.2 Directory Links for Diskless Nodes

All diskless nodes and the server node share a common file system. When you access a file from one node, that is the same as accessing that file from any other node. However, some files are unique to each diskless node, and to the server node itself.

The unique, node-specific files fall into two categories:

- Configuration or data base files that contain node-specific information
- Temporary files and log files, where two or more processes running under different kernels might try to write simultaneously

The first case includes files such as */etc/ttys*, which usually are different for each node on the network. The second case includes the */tmp* and */usr/tmp* directories, where two processes on different kernels specify the same name for a temporary file. .

For example, programs like sort (1) create temporary files to hold the sorted information. These files are created in the */usr/tmp* directory and have a name of the form */usr/tmp/stm$$*, where *$$* is expanded to represent the current process ID.

In single-processor systems, attaching the process ID to the temporary file name prevents collisions between concurrent sorts because all process IDs are unique. On a diskless server's file system, two processes with the same ID might exist, which would create one temporary file rather than two separate temporary files. Therefore, separate processors must have separate temporary areas.

The UNIX kernel solves this problem by providing naming conventions for node-specific files and directories. The naming convention is based on the node hostname. The kernel provides unique names by expanding, or translating certain string variables when it evaluates the targets of symbolic links:

- $HOST translates to the hostname of the node from which the request originates, server or remote.
- $RHOST translates to the hostname of the remote node from which the request originates. If the request originates on the server node, $RHOST translates to a null string.
- $REMOTE translates to the string remote when interpreted on a remote node, and to a null string on the server node.
- $MACHINE translates to one of the following:
  - — Q10 for an Optimum (Q-bus) system
  - — V10 for an Optimum V system with a VME-68K10 CPU board
  - — V20 for an Optimum V system with a VME-68K20 CPU board

As an example, consider a symbolic link to the file *test.$RHOST*. When a command is issued on the server node that refers to this symbolic link, it tries to access a file "*test.*". A command issued from a diskless node named dws1 tries to access a file "*test.dws1*".

Through symbolic links, this process is transparent to the user. A file or directory name such as */etc/ttys* is actually a symbolic link to a file or directory name which has one of the string variables embedded. The kernel expands the string and accesses the correct file for the requesting node.

For example, the following command links the directory */tmp* to a directory name with the embedded string:

```
ln -s /.tmp.\$RHOST /tmp
```

(Note the backslash (\) precedes the dollar sign ($), which "escapes," or prevents shell interpretation.) On a network with a server node named central and diskless nodes dws1 and dws2, a pathname that includes */tmp* will translate to

- (on central) */.tmp.*
- (on dws1) */.tmp.dws1*
- (on dws2) */.tmp.dws2*

Each of these */.tmp* directories then contains files specific to its related node. The user simply types /tmp in pathnames; the kernel handles the interpretation invisibly.

Specifying a file by its full name (for example, */etc/.tmp.dws1*) will always access that particular file.

When the **admin** (8) program configures a server node for a diskless network, it makes node-specific links for these files and directories:

- */dev*
- */etc/fstab*
- */etc/motd*
- */etc/psdatabase*
- */etc/rc*
- */etc/rc.local*
- */etc/ttys*
- */etc/utmp*
- */tmp*
- */usr/adm*
- */usr/lib/crontab*
- */usr/lib/mail*
- */usr/spool/mqueue*
- */usr/spool/rwho*
- */usr/tmp*
- */vmunix*

Check your own UNIX installation for local programs that use temporary or log files. If any programs use temporary or log files in areas other than those listed above, and these programs can run on more than one processor simultaneously, you must create a symbolic link either for the files or for the directory in which they reside.

When you create node-specific symbolic links, you might want to make the "real" file or directory names begin with a period:

```
/tmp --> /.tmp.SRHOST
```

This way, the casual user will see only the link and not be confused by a profusion of names like

/tmp.dws1
/tmp.dws2
/tmp.dws3 . . .

since the **ls** (1) command does not normally show filenames that begin with a period (.).

7.3 Configuring Diskless Nodes

Configuring a diskless network involves four tasks:

1. Arranging for swap space.
2. Choosing a network address for the server node, which should already be done as shown in Section 3.3, "Initializing Your System."
3. Choosing a name and network address for each diskless node.
4. Configuring the network with **admin** (8).

7.3.1 Allowing Swap Space for Diskless Nodes

Diskless nodes require swap space on the server node. The server node uses assigned disk partitions for swap space; the diskless nodes, having no disks of their own, use files for swap space.

Diskless nodes use a file */usr/spool/diskless/*name*.swap* on the server node, where "name" is the hostname of the diskless node. **admin** (8) creates the swap directory when configuring diskless nodes.

These files can be very large, up to 16.5 Mbytes apiece (16,500 Kbytes, as measured by the **ls**(1) and **df**(1) commands). When you configure the file systems for your network, you must allow enough space for these files in the */usr* partition.

Normally, the files are not that large. To see the actual size of the files, enter the command

```
ls -s /usr/spool/diskless/*.swap
```

The **-s** option to **ls** shows the size in blocks of the files. Do not use the **-l** option to look at file size; this gives a character count, an inaccurate measure in this case.

Enter the command

```
df
```

to see current disk usage. Check the "available" (**avail**) column for the */usr* file system. There should be at least 5000 Kbytes available for each diskless node on the network. Section 4.3, "Configuring Disks," tells you how to change disk configuration if you need to assign a larger partition for swap files. You can also create a "spanned disk" and use partitions from several disks to create a very large (or simply more efficient) file system for the swap file area.

If the */usr* file system is not large enough to hold the swap files, you can either use a symbolic link to put the swap files in another file system or create a spanned disk to enlarge the file system. For example, the command

```
ln -s /diskless/usr/spool/diskless /usr/spool/diskless
```

points to a directory in another area, a */diskless* file system that you can create with the **newfs**(8) command as described in Section 4.3.4, "Adding a New File System." Mount (**mount**(8)) this file system on a sufficiently large disk partition.

7.3.2 Choosing Names and Addresses for a Diskless Network

When you configure a diskless network, you assign a network address for the server node during **admin**(8) initialization. Then, when adding each diskless node to the network, you must provide an internet address for the diskless node.

If you have not already done so, complete the procedure in Section 3.3, "Initializing Your System." This procedure assigns a name and internet address for the server node. You must complete this naming procedure before installing diskless nodes.

Choose hostnames for each of the diskless nodes and write them down. Use hostnames with less than twelve characters. Do not use spaces within hostnames; use underscores to separate words (for example, dl_name). The hostnames must be unique within the network.

Choose an internet address for this diskless node. The file */etc/hosts* defines internet addresses and hostnames; look in this file for examples of internet addresses. Network addresses have this format:

n.n.n.n

where each *n* is a decimal number **1-255**.

The first of the four numbers in an internet address is the "network number." You should use the same network number as the Ethernet controller of the server node.

In "Appendix C: Configuration Worksheet," in the area reserved for networks, enter this information:

- Server node's hostname and internet address
- Hostname and internet address of each diskless node

7.3.3 Configuring the Network With admin (8)

Use the following Procedure to Add a Diskless Node to the Network to configure the diskless network or to add a diskless node to an existing network.

Procedure to Add a Diskless Node to the Network

	ACTION	WHAT IT DOES
Step 1.	:f you have not yet done so, configure the server node with admin (8).	Sets up the server for diskless operations. See Section 3.3, "Initializing Your System."
Step 2.	Enter the command /etc/admin	Invokes the administration program.
Step 3.	Select option C at the top menu of the admin (8) program.	Selects operations dealing with local (diskless or cluster) networks.
Step 4.	Select option A from the next menu.	Selects adding a diskless node to the network. **admin** now prompts for the name and internet address of the diskless node, and whether or not this is a graphics node. See Table 7-1 for further details on admin operations.

End of procedure

Table 7-1. admin Adding a Diskless Node

What **admin** (8) Does	Affected Files
Prints the top-level menu. Select item C, "Diskless/Cluster installation and deletion."	—
Prints the diskless/cluster menu. Select item A, "Add a diskless node."	—
Prompts for the name of the diskless node. Enter a name, with the same restrictions on length and characters as for the server node name.	—
Prompts for the internet address of the diskless node. See Section 7.3.2, "Choosing Names and Addresses for a Diskless Network," to determine an internet address.	—
Prompts for graphics configuration of the diskless node. If this is a Diskless WorkStation (graphics), answer y; if this is a Diskless Compute Node (non-graphics), answer **n**.	—
Prints the diskless node name and internet address, formatted as for the file */etc/hosts*. You should write this information in "Appendix C: Configuration Worksheet." If you accept the information printed by **admin**, press <return> to continue the procedure. If you want to abort admin, press ^C (CTRL-C) to prevent any changes to system files.	—
Writes the diskless node name and internet address to */etc/hosts*.	*/etc/hosts*
Creates the swap directory */usr/spool/diskless*, if it does not already exist.	*/usr/spool/diskless*
Creates various "hidden" files and directories for the diskless node. See Section 7.2, "Directory Links for Diskless Nodes."	*/dev* */etc/fstab* */etc/motd* */etc/psdatabase* */etc/rc* */etc/rc.local* */etc/ttys* */etc/utmp* */tmp* */usr/adm* */usr/lib/crontab* */usr/lib/mail* */usr/spool/mqueue* */usr/spool/rwho* */usr/tmp* */vmunix*

7.4 Booting Diskless Nodes

Boot the server node as any other Optimum or Optimum V system using the procedures in "Section 2: Starting UNIX from Disk." Once the server node is running, boot the diskless nodes one at a time.

Booting a Diskless WorkStation (a graphics node) requires graphics software on the server node. If the server is a WorkStation, this software is already in place. Graphics software is on the standard ISI UNIX 4.2/3BSD release tape from Release 3.05 onward. Contact ISI Customer Support if you do not have the necessary software release.

Use the following procedure, the Procedure to Boot a Diskless Node, to boot a diskless node. Figure 7-1 shows a sample of a normal diskless node boot.

Procedure to Boot a Diskless Node

	ACTION	WHAT IT DOES
Step 1.	Boot the server node, if it is not yet running. Continue the boot to multiuser UNIX.	You cannot boot diskless nodes without a functioning server node. Use the procedures in "Section 2: Starting UNIX from Disk" to boot the server node.
Step 2.	Check that the server node has files of the name */dev/enet[0-3]a*	Checks for */dev* files to support remote booting. These files must exist in the */dev* directory. If they are not there, create them with the procedure in Section 4.6, "Making New Devices in */dev.*"
Step 3.	Enter the command `ps -aux | grep bootd`	Checks that the boot daemon, */etc/bootd*, is running in the background. If it is not, enter the command `/etc/bootd &` and check that the file /etc/rc.local to see that it has a command to start the daemon.
Step 4.	Enter the command `ps -aux | grep '@-`*`name`*`'` where *name* is the hostname of the diskless node you want to boot, and '*@-name*' is enclosed by single quotes (apostrophes).	Checks for leftover processes from the last boot of this diskless node. You cannot boot if these processes exist. See Section 7.5, "Shutting Down Diskless Nodes," to remove these processes. These processes occur on server nodes that service requests from diskless nodes. They execute entirely in kernel mode, with no associated user.
Step 5.	Turn on power for the diskless node's console (or monitor if the diskless node is a WorkStation).	Powers up the console (monitor).
Step 6.	Press RETURN on the console keyboard. Wait for the PROM prompt (:).	Displays the PROM prompt for the diskless node. You can now enter commands to the diskless node's CPU PROMS.

(continued on next page)

Procedure to Boot a Diskless Node *(continued)*

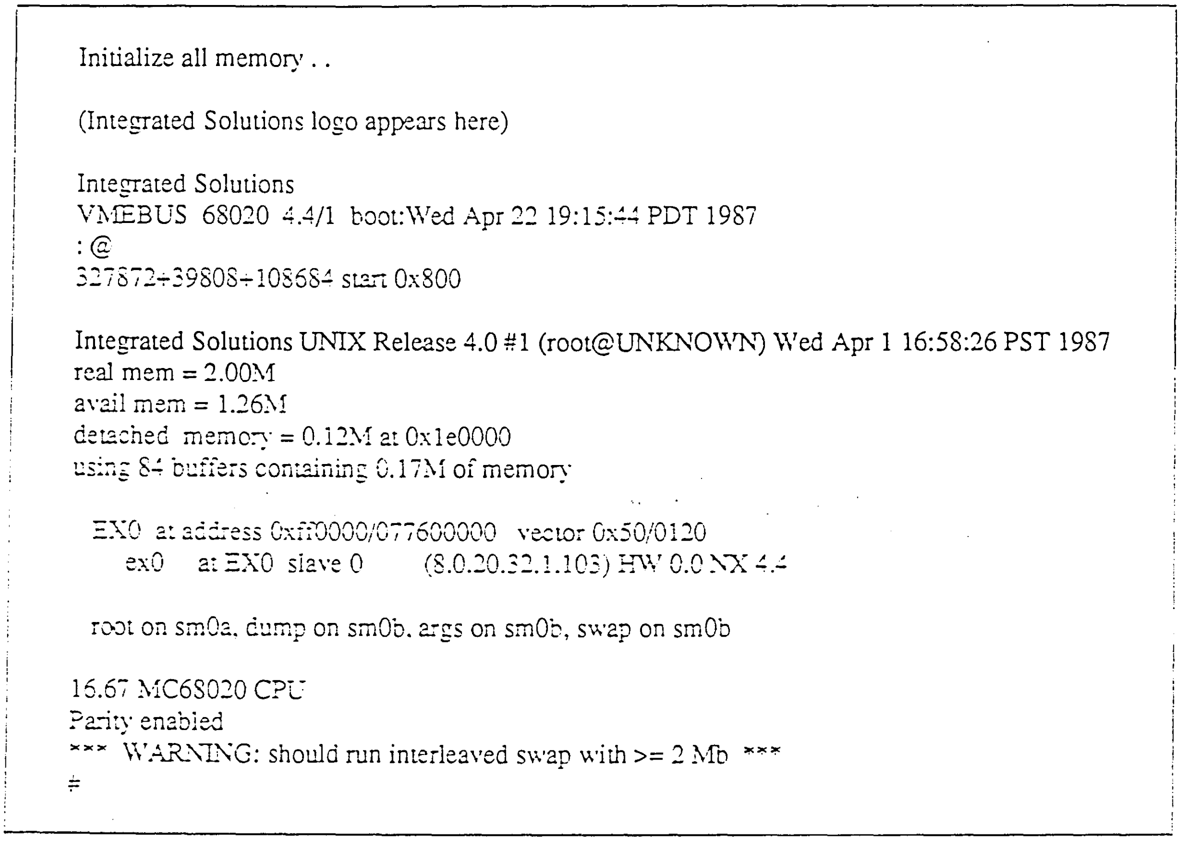
	ACTION	WHAT IT DOES
Step 7.	Enter the character "@" (SHIFT-2 on most terminals) and press RETURN.	Boots the diskless node through the Ethernet driver (ex or nw) from the server node's disk. After a successful boot, the console shows the single-user UNIX prompt (#). The "@" command broadcasts a request from the diskless node over the Ethernet; any server node on the network that is configured to support this diskless node can respond. If two or more servers respond, the first to respond becomes the server for this diskless node. If the "@" boot does not work, or if you want to name a specific server node, you can boot manually from the PROMs. Enter the command *dev*(0,0)*kernelhost*:*kernel* HOST=*hostname* SERVER=*server* all on one line, where *dev* is the Ethernet controller on the diskless node, either • nw for the VME-ECX board, or • ex for the Excelan Ethernet board *kernelhost* is the node from which you want to use the specified kernel. Note that this is usually, but not necessarily, the same hostname as *server*. *kernel* is the name of the kernel for booting. This is usually vmunix. *hostname* is the hostname of this diskless node. *server* is the hostname of the server node. For example, this command boots a diskless node named dws1, with a VME-ECX Ethernet controller, from a server node named central: nw(0,0)central:/vmunix HOST=dws1 SERVER=central
Step 8.	Enter the command date *yymmddhhmm* where *yymmddhhmm* is today's date.	Tells UNIX what time it is. See date(1) for details on the entry format.

(continued on next page)

Procedure to Boot a Diskless Node *(continued)*

	ACTION	WHAT IT DOES
Step 9.	Enter a ^D (CTRL-D; hold down the CTRL key and press D).	Boots to multiuser UNIX using the shell script */etc/.rc.local.name*, where *name* is the hostname of the diskless node. After displaying several messages, the monitor shows this prompt: **IS68K 4.3 BSD** (*hostname*) **login:** where *hostname* is the hostname of this node. This is the UNIX login prompt.
Step 10.	Log in as the superuser by entering root as your login name. The **root** password is the same for all nodes on the local diskless network; you must use the same password as for the server node.	Logs in to UNIX, giving access to a working C shell (UNIX command processor). You can now issue UNIX commands to the shell prompt (#). Note that a Diskless WorkStation (graphics node) requires graphics software on the server node. See the comment at the beginning of this subsection.

End of procedure



```
Initialize all memory . .

(Integrated Solutions logo appears here)

Integrated Solutions
VMEBUS  68020  4.4/1  boot:Wed Apr 22 19:15:44 PDT 1987
: @
327872+39808+108684 start 0x800

Integrated Solutions UNIX Release 4.0 #1 (root@UNKNOWN) Wed Apr 1 16:58:26 PST 1987
real mem = 2.00M
avail mem = 1.26M
detached  memory = 0.12M at 0x1e0000
using 84 buffers containing 0.17M of memory

  EX0  at address 0xff0000/077600000   vector 0x50/0120
     ex0    at EX0  slave 0        (8.0.20.32.1.103) HW 0.0 NX 4.4

  root on sm0a, dump on sm0b, args on sm0b, swap on sm0b

16.67 MC68020 CPU
Parity enabled
***  WARNING: should run interleaved swap with >= 2 Mb  ***
#
```

Figure 7-1. Sample of a Normal Diskless Node Boot

7.5 Shutting Down Diskless Nodes

To shut down a diskless network, first shut down the individual diskless nodes, then shut down the server node with the procedure in Section 2.4, "Shutting Down the System."

CAUTION

> Failure to shut down the system in an orderly fashion may result in lost data. In particular, simply turning off power without an orderly shutdown will probably result in data loss.

The shutdown procedure differs depending on your current condition:

- If you are running multiuser UNIX, begin with *Step 1* in the following procedure.
- If you are running single-user UNIX (the command line shows the single-user prompt (#)), skip to *Step 2* of the following procedure.
- If the diskless node has crashed, skip to *Step 4* of the following procedure before rebooting.

Use the following procedure, the Procedure to Shut Down a Diskless Node, to shut down a diskless node.

Procedure to Shut Down a Diskless Node

	ACTION	WHAT IT DOES
Step 1.	Enter this command: `shutdown +n` where *n* is a time in minutes.	Notifies users of shutdown, waits for the specified time, and then kills all processes. When shutdown is complete, the monitor shows the single-user prompt (#). CAUTION If you turn off your system at this point, you risk losing data. You must finish all steps of this procedure.
Step 2.	Enter the command `reboot`	Updates the superblock and brings the diskless node to the PROM prompt (:). Wait until the PROM prompt (:) appears.
Step 3.	Turn OFF power for the diskless node.	Removes power from the diskless node. The diskless node is now off. Turn off power for the console (or WorkStation monitor) as well.

(continued on next page)

Procedure to Shut Down a Diskless Node *(continued)*

ACTION		WHAT IT DOES
Step 4.	At the server node, enter the command ps -aux | grep '@-*name*' where *name* is the hostname of the diskless node you just shut down, and '@-*name*' is enclosed by single quotes (apostrophes).	Displays all processes running on the server node that are associated with the diskless node. These processes are now useless, and will interfere with any efforts to reboot the diskless node. If this command shows no processes on the server node you are now finished with this procedure.
Step 5.	Enter the command killpg -9 *pid* where *pid* is any one of the process IDs displayed in *Step 4*.	Kills the named process and all other processes in the process group. See killpg (8) for more details on this command.

End of procedure

7.6 Booting Single-User on Diskless Nodes

Diskless nodes use the same UNIX modes as normal Systems or WorkStations. Boot the diskless node to multiuser UNIX for normal operation. Other modes are useful for

- Issuing PROM commands (Section 7.6.1)
- Running diagnostics (Section 7.6.2)

Table 7-2 shows the UNIX modes and their purposes.

Table 7-2. UNIX Operation Modes for Diskless Nodes

Prompt	Mode	Purpose
:	PROM	Communicating with the UNIX bootstrap PROMs on the CPU (see Section 1.2) for • booting UNIX (Section 7.4) • using PROM commands (Section 7.6.1) • stand-alone diagnostics (Section 7.6.2)
#	Single-user UNIX	Issuing commands as root with no other users logged in. Useful for changing hardware configurations.
system-name # or $ or % or login:	Multiuser UNIX	Multiuser UNIX operation. This is the normal mode for the system: normal booting tries to reach this level.

Table 7-3 shows how to move from any mode to any other mode. Find the mode you are in and read across to the column for the mode you want.

Table 7-3. Moving Between UNIX Modes on Diskless Nodes

Want: Now:	PROM :	Single-User UNIX #	Multiuser UNIX *system-name #* (or $, %, login:)
PROM :		Enter the character "@", as described in Section 7.4, "Booting Diskless Nodes." If this does not work, or if you want to name a specific server node, use the command *dev*(0,0)*kernelhost*:*kernel* HOST=*hostname* SERVER=*server* all on one line, as described in the same section.	Follow the procedure in Section 7.4, "Booting Diskless Nodes."
Single-User UNIX #	Enter # reboot Wait for the PROM prompt to appear. CAUTION Pushing RESET can crash the system down to the PROM prompt, but this can write bad data to the server node's disks, even with a sync(8) first. Always run fsck(8) on the server node after a RESET.		Enter a ^D (CTRL-D; hold down the CTRL key and press D) to force a quick switch to multiuser. For a full reboot, enter the command reboot and boot from the PROM prompt as described above in this table.
Multiuser UNIX *system-name #* or $ or % or login:	Use the procedure in Section 7.5, "Shutting Down Diskless Nodes," until the PROM prompt appears (after *Step 2*). See CAUTION above.	Use the shutdown(8) command as described in Section 7.5, "Shutting Down Diskless Nodes." This informs others of the shutdown, sends warning messages, then boots down to single-user.	

7.6.1 Using Commands at the Diskless Node PROM Prompt

The PROM prompt (:) gives access to the CPU PROMs. At this prompt you can use the commands in Table 7-4.

Table 7-4. Commands at the Diskless Node PROM Prompt

Command	What It Does
@	Boots the diskless node to single-user UNIX, using the kernel */.vmunix.$HOST* on the server node. The "@" command broadcasts a request from the diskless node over the Ethernet; any server node on the network that is configured to support this diskless node can respond. If two or more servers respond, the first to respond becomes the server for this diskless node.
!	Toggles interrupt enabling. Entering ! enables monitor interrupts; entering ! again disables them.
dev(0,*m*)*source*:*filename*	Loads and starts execution of a file on a device, where *dev* is either • nw for the VME-ECX Ethernet board, or • ex for the Excelan Ethernet board which actually accesses the boot disk on the server node. Commands at a diskless node PROM prompt can access only the boot disk. *m* is the device "minor" number, the partition number on the boot disk. *source* is the hostname of the node where the desired kernel resides. *filename* is the path to the file you want to execute on disk. For example, you could issue this command: nw(0,0)v24server:stand/V20/mmu to execute the file */stand/V20/mmu* on the *a* (first, numbered from zero) partition of the boot disk on a server node named v24server with a VME-ECX Ethernet controller. If you want to go directly to a block instead of a partition, you can specify a number followed by "b". For example, sd(0.10b) In some applications, it may be desirable to load a program without automatically beginning its execution. Preceding the load string with "<" causes the file to be loaded and a starting address to be printed. Then control is returned to the PROM monitor. The program can then be started by typing a ">" followed by the specified starting address.

(continued on next page)

Table 7-4. Commands at the Diskless Node PROM Prompt *(continued)*

Command	What It Does
% address [value]	Opens/modifies location as a byte where *address* is the hexadecimal address for that memory location and *[value]* is the optional new hexadecimal value to write at address.
$ address [value]	Opens/modifies location as a word where *address* is the hexadecimal address for that memory location and *[value]* is the optional new hexadecimal value to write at address.
# address [value]	Opens/modifies location as a longword where *address* is the hexadecimal address for that memory location and *[value]* is the optional new hexadecimal value to write at address. You can use these commands in two ways: by entering the address in hex $fff520 or by entering the address in hex with a new value for that address $fff520 123456 If you specify the value, the location is written to without ever being read. When you type % and the hexadecimal address of a memory location, the system appends a bar (|) followed by the current byte value at that address and waits for you to enter the hexadecimal value of the new location. When you type $ and the hexadecimal address of a memory location, the system appends a colon (:) followed by the current word value at that address and waits for you to enter the hexadecimal value of the new location. When you type # and the hexadecimal address of a memory location, the system appends a semi-colon (;) followed by the current word value at that address and waits for you to enter the hexadecimal value of the new location. For example, entering $200 might display 200:68AC _ where "_" represents the cursor location. If you do not type in a new value and you press RETURN, the location remains unmodified.

(continued on next page)

Table 7-4. Commands at the Diskless Node PROM Prompt *(continued)*

Command	What It Does
	If you type in a new hex value following the displayed hex value 200:68AC 68AF and press RETURN, the new value is written in the current location, and the colon (:) prompt appears. . If you type in a new hex value following the displayed value 200:68AC 68AF and press LINE FEED, the new value is written to the current location and the next location is opened, rather than returning to the PROM prompt (:). If you type in a new hex value following the displayed value and follow it with a backslash (\), 200:68AC 68AF \ the new value is written to the current location and the previous location is opened, rather than returning to the PROM prompt (:). Opening a non-existent memory location with interrupts enabled will cause the system to continuously attempt to access the non-existent location, causing trap errors.
> *address*	Begins execution at a hex address. If you enter the "greater than" sign (>) and an address in hex, you will immediately begin execution at that address. For example, entering >68AC begins execution at 68AC.
~	Puts the CPU into transparent mode. Characters from the console are transmitted to the second serial port and characters from the second serial port are directed to the console port.
~.	Exits transparent mode and returns to the monitor prompt (:).
~~	Sends one tilde (~) to the transparent side.

7.6.2 Running Diagnostics on Diskless Nodes

Two hardware diagnostic programs are useful for diskless nodes:

1. **mmu** to test the Memory Management Unit on the CPU
2. **mem** to test main memory

Use these diagnostic programs only if you suspect hardware failures. For example, run diagnostics in response to these symptoms:

- Diskless node will not boot
- Diskless node crashes frequently

Since diskless nodes cannot execute programs from a tape drive, you must install the diagnostic programs on the server node's boot disk (or on the boot disk of any diskless server node on the local network). See Section B.2.2, "Running Diagnostics From UNIX," for instructions to install diagnostic programs on the server node.

Use the following Procedure to Load Diagnostics on a Diskless Node to run hardware diagnostics.

Procedure to Load Diagnostics on a Diskless Node

	ACTION	WHAT IT DOES
Step 1.	Check that the monitor displays the PROM prompt (:).	Checks for the right mode for diagnostics.
Step 2.	Enter this command: *dev*(0,0)*server*:stand/*SYS*/*test* where *dev* is either • nw for the VME-ECX Ethernet board, or • ex for the Excelan Ethernet board which actually accesses the boot disk on the server node. *server* is the hostname of the node where the desired test resides. *SYS* is your system type (CPU type): • V20 for the VME-68K20 • V10 for the VME-68K10 • Q10 for the IS-68K (Q-bus) *test* is the diagnostic test, either • mmu or • mem	Loads the program and begins execution. For example, to execute the mmu test through a VME-ECX Ethernet controller, on a diskless node with a VME-68K20 CPU board, enter **nw(0,0)v24server:stand/V20/mmu** where v24server is the hostname of the server node. This diskless node has a VME-68K20 CPU board. The (0,0) refers to the first partition of the first disk (numbered from 0, remember), also known as partition *a*, holding the root (/) file system. The console should now display this message: **Type RETURN to start at 0x*wwww*** If the command does not work, check that the directory */stand* on the server node contains diagnostic programs. If the directories */stand/V20*, */stand/V10*, and */stand/Q10* do not exist, you must install them from the diagnostic tape. See Section B.2.2, "Running Diagnostics From UNIX"
Step 3.	Press RETURN.	Starts the test.
Step 4.	Follow the instructions in "Appendix B: Using Diagnostic Programs" to execute the diagnostic test. See either Section B.3, "Using mmu" or Section B.4."Using mem"	Runs the diagnostic test. The tests themselves are identical in procedure to those run on the server node or any other node.

End of procedure

SECTION 8: DAY-TO-DAY OPERATIONS

This section gives procedures for daily operation and maintenance of the UNIX system.

This section contains the following subsections:

8.1 What Is UNIX Maintenance?

After booting the system and setting it up, the System Administrator's daily duties include:

- Troubleshooting—fixing problems such as system crashes and corrupted file systems
- Monitoring—checking various parts of UNIX to make sure that everything is running well
- Accounting—keeping track of users, making sure that they follow any rules for this site (such as disk quotas)
- Backing up—copying disk contents onto tape for safe storage, and restoring files from these backup tapes

8.2 Recovering From Crashes

If the system crashes, it will ordinarily reboot automatically to multiuser operation. If the boot halts and gives the single-user prompt (#), try entering **reboot** for a fresh start. If the boot halts and gives the PROM prompt (:), use the procedure in Section 2.3, "Booting the System."

Use fsck (S) to clean up the disks after a crash; see Section 2.6, "What to Do if the Autoboot Fails."

If a cluster node crashes, see Section 6.5, "Shutting Down Cluster Nodes." If a diskless node crashes, see Section 7.5, "Shutting Down Diskless Nodes."

Under some circumstances you may want to interrupt the autoboot in order to execute commands at the single-user UNIX prompt (#) or at the PROM prompt (:). For example, the diagnostic programs in Appendix B require the PROM prompt. See Section 8.3 for single-user and PROM boots.

If the system crashes and prints a terse hardware diagnostic message on the system console, look up the diagnostic message in the *UNIX Programmer's Reference Manual* (PRM), Section 4. For example, if the message

```
sdn: lost interrupt.
```

appears, look at sd (4).

UNIX saves these hardware diagnostic messages in a file */usr/adm/messages*. See Appendix B for diagnostic programs to correct hardware errors.

When the system crashes voluntarily it prints on the console a message of the form

panic: *message*

UNIX dumps a core image then triggers an automatic reboot, as described in "Section 2: Starting UNIX From Disk." If fsck (8) finds no faults, the system resumes multiuser operation.

One common cause of a **panic** crash is hardware failure. Table 8-1 shows some common **panic** crash messages with explanations.

Table 8-1. Panic Crash Messages

Crash Message	What It Means
panic: **IO err in push** **hard IO err in swap**	The system encountered an error trying to write to the paging device, or an error in reading critical information from a disk drive. Fix the disk if it is broken or unreliable.
panic: **timeout table overflow**	The timeout table ran out of entries. A minor problem; rebooting usually fixes it.
panic: **trap type** *n* where *n* is a number 0-13	The system encountered an unexpected trap. The trap types are 0 Reset 2 Bus error 3 Address error 4 Illegal instruction 5 Zero divide 6 CHK instruction 7 TRAPV instruction 8 Privilege violation 9 Trace 10 Line 1010 emulation 11 Line 1111 emulation
panic: init died	The system initialization program, init(8), has died, so no new users can log in. Rebooting is the only repair for this, so the system reboots.

8.3 When Do You Boot Single-User?

The system can operate in three modes, each with a different prompt. Table 8-2 shows these modes. For cluster nodes see Section 6.6, "Booting Single-User on Cluster Nodes." For diskless nodes see Section 7.6, "Booting Single-User on Diskless Nodes."

Table 8-2. UNIX Operation Modes

Prompt	Mode	Purpose
:	PROM	Communicating with the UNIX bootstrap PROMs on the CPU (see Section 1.2) for • installing UNIX from tape (Appendix A) • booting UNIX (Section 2) • stand-alone diagnostics (Appendix B) • using PROM commands (Section 8.4)
#	Single-user UNIX	Issuing commands as **root** with no other users logged in. Useful for • mounting and unmounting file systems (mount (8)) • checking the disks (fsck (8)) • backing up the file systems (Section 8.7) • changing hardware configurations (Section 4.5)
system-name # or $ or % or **login:**	Multiuser UNIX	Multiuser UNIX operation. This is the normal mode for the system; autobooting tries to reach this level.

Boot to the single-user UNIX prompt (#) to have the system to yourself. In this mode, no one else can log in. This is particularly useful for commands (fsck (8), umount (8), etc.) that require an inactive file system.

Boot to the PROM prompt (:) to use the stand-alone diagnostics in Appendix B, or to boot UNIX from some kernel source other than /*vmunix* (*dev*(0,0)vmunix to the PROM prompt).

You can also use the PROM prompt to issue commands in Section 8.4, "Using Commands at the PROM Prompt."

Table 8-3 shows how to move from any mode to any other mode. Find the mode you are in and read across to the column for the mode you want.

Table 8-3. Moving Between UNIX Modes

Want: Now:	PROM :	Single-User UNIX #	Multiuser UNIX *system-name #* **(or $, %, login:)**
PROM :		Enter **: *dev*(0,0)vmunix** where *dev* is the name of the system's boot disk.	Enter the "@" character for a full autoboot. Or shortcut by booting to single-user (see this table) and from single-user to multiuser (see this table). CAUTION This method bypasses disk checks; do this only after a graceful shutdown.
Single-User UNIX #	Enter **# umount -a; reboot** Wait for the PROM prompt to appear. After ten seconds with no entry, the system will autoboot. CAUTION Pushing RESET can crash the system down to the PROM prompt, but this can thrash the disks, even with a sync(8) first. Always run fsck(8) after a RESET.		Enter **# reboot** to perform a full reboot. Entering a ^D (CTRL-D; hold down the CTRL key and press D) forces a quick switch to multiuser. CAUTION This method bypasses disk checks; do this only after a graceful shutdown.
Multiuser UNIX *system-name #* or $ or % or login:	Use the procedure in Section 2.4, "Shutting Down the System," until the PROM prompt appears (after *Step 2*). See CAUTION above.	Use the shutdown(8) command as described in Section 2.4, "Shutting Down the System." This informs others of the shutdown, sends warning messages, then boots down to single-user.	

8.4 Using Commands at the PROM Prompt

The PROM prompt (:) gives access to the CPU PROMs. At this prompt you can use the commands in Table 8-4. For PROM commands on cluster nodes, see Section 6.6, "Booting Single-User on Cluster Nodes." For PROM commands on diskless nodes, see Section 7.6, "Booting Single-User on Diskless Nodes."

Table 8-4. Commands at the PROM Prompt

Command	What It Does
@	Boots the system to multiuser UNIX, using the kernel */vmunix* on the default boot disk drive.
!	Toggles interrupt enabling. Entering ! enables interrupts; entering ! again disables.
dev(M,m)filename	Loads and starts execution of a file on a device, where *dev* is the device name (e.g., sd, ts) *M* is the device number (e.g., 0 for the first such device, 1 for the second) *m* is the device "minor" number, usually • for disks, the partition number • for tapes, the file number on the tape *filename* is the path to the file you want to execute on disk. There is no *filename* when you specify a tape device. For example, you could issue these commands: ts(0.8) sd(0.6)stand/bad144 to execute the eighth file on a tape or to execute the file *stand/bad144* on the *g* (7th, numbered from zero) partition of an SD disk. If you want to go directly to a block instead of a partition, you can specify a number followed by "b". For example, sd(0.10b) In some applications, it may be desirable to load a program without automatically beginning its execution. Preceding the load string with "<" causes the file to be loaded and a starting address to be printed. Then control is returned to the PROM monitor. The program can then be started by typing a ">" followed by the specified starting address.

(continued on next page)

Table 8-4. Commands at the PROM Prompt *(continued)*

Command	What It Does
% address [value]	Opens/modifies location as a byte where *address* is the hexadecimal address for that memory location and *[value]* is the optional new hexadecimal value to write at address. Table 8-5 shows memory locations for Optimum V system board addresses. Table 8-6 shows memory locations for Optimum system board addresses.
$ address [value]	Opens/modifies location as a word where *address* is the hexadecimal address for that memory location and *[value]* is the optional new hexadecimal value to write at address.
# address [value]	Opens/modifies location as a longword where *address* is the hexadecimal address for that memory location and *[value]* is the optional new hexadecimal value to write at address. You can use these commands in two ways: by entering the address in hex $fff520 or by entering the address in hex with a new value for that address $fff520 123456 If you specify the value, the location is written to without ever being read. When you type % and the hexadecimal address of a memory location, the system appends a bar (|) followed by the current byte value at that address and waits for you to enter the hexadecimal value of the new location. When you type $ and the hexadecimal address of a memory location, the system appends a colon (:) followed by the current word value at that address and waits for you to enter the hexadecimal value of the new location. When you type # and the hexadecimal address of a memory location, the system appends a semi-colon (;) followed by the current word value at that address and waits for you to enter the hexadecimal value of the new location. For example, entering $200 might display 200:68AC _ where "_" represents the cursor location. If you do not type in a new value and you press RETURN, the location remains unmodified.

(continued on next page)

Table 8-4. Commands at the PROM Prompt *(continued)*

Command	What It Does
	If you type in a new hex value following the displayed hex value 200:68AC 68AF and press RETURN, the new value is written to the current location, and the colon (:) prompt appears. If you type in a new hex value following the displayed value 200:68AC 68AF and press LINE FEED, the new value is written to the current location and the next location is opened, rather than returning to the PROM prompt (:). If you type in a new hex value following the displayed value and follow it with a backslash (\), 200:68AC 68AF \ the new value is written to the current location and the previous location is opened, rather than returning to the PROM prompt (:). Opening a non-existent memory location with interrupts enabled will cause the system to continuously attempt to access the non-existent location, causing trap errors.
> *address*	Begins execution at a hex address. If you enter the "greater than" sign (>) and an address in hex, execution begins immediately at that address. For example, entering >68AC execution begins immediately at 68AC.
~	Puts the CPU into transparent mode. Characters from the console are transmitted to the second serial port and characters from the second serial port are directed to the console port.
~.	Exits transparent mode and returns to the monitor prompt.
~~	Sends one tilde (~) to the transparent side.

Table 8-5. Optimum V System Board Addresses in Memory[†]

Function	Device	Board	Default	Alternate
Tape	ts	VME-QIC2/R	fff550	fff520
Disk	sd	VME-SCSI	ffffe0	fffff0
	sm	VME-SMD	0 in short I/O space	0200, 0400, 0600
Serial Ports	cp	VME-ICP16/X	fff520	fff560, fff580, fff5a0, fff5c0
Ethernet	ex	Excelan	ff0000	fe0000,ff0100, fe0100
	nw	VME-ECX	f80000	
Other	gp	Graphics—GIP[††]	ffc000	ff8000, ff4000, ff0000, fec000
		Graphics—Display Memory[††]	e00000	d00000, c00000, b00000, a00000
	sky	VME-FFP	ff8100	ff8110, ff8120, ff8130, ff8140

Table 8-6. Optimum System Board Addresses in Memory[†]

Function	Device	Board	Default	Alternate
Tape	ts	IS-QIC2	3ff550	3ff520
Disk	cl	RL101	3ff900	3ff910
	hp	RL101	3ffdc0	3ffcc0
	rx	RX02	3ffc78	3ffc68
Serial Ports	dh	ISDH	3fe010	3fe020, 3fe030, 3fe040, 3fe050, 3fe060, 3fe070, 3fe080
	dm	ISDH	3ff140	3ff148, 3ff150, 3ff158, 3ff160 3ff168, 3ff170, 3ff178
	dl	ISDH (DL emulation)	3ffd40	3ffd48, 3ffd50. 3ffd58, 3ffd60. 3ffd68, 3ffd70. 3ffd78
	dz	ISDZ	3fe008	3fe010, 3fe018, 3fe020, 3fe028. 3fe030. 3fe038. 3fe040
Ethernet	ex	Excelan	3fe800	3fe810, 3fe820, 3fe830
	il	Interlan	3fe800	3fe810, 3fe820
Other	lp	ISLP	3fff4c	
	sky	IS-FFP	3fe280	

[†] All memory addresses are hexadecimal numbers. On Optimum V systems with 68010 CPUs memory addresses start with "7" instead of "f."

[††] Addresses for GIP and Display Memory are matched pairs.

8.5 Monitoring Your System

Table 8-7 shows the tasks to perform at various intervals.

Table 8-7. Maintenance Tasks

Interval	Task	Why
Daily	**dump w**	Shows the file systems that should be dumped. **dump** (8) uses fstab (5) and */etc/dumpdates* to keep track of this. See Section 8.7, "Backing Up Your System," for a more thorough system of daily backups (the ''Towers of Hanoi'' schedule).
Weekly	df	df (1) shows the current capacity of each file system. Check that none are getting too full. For a cluster or diskless network, check that the */usr* file system has enough room for swap files.
	quot	quot (8) checks the amount of disk space taken up by each user. Use this command only when using system accounting (see Section 8.6, "Accounting").
	fsck -p	Checks the file systems for trash. Boot down to single-user to do this (see Section 8.3). If fsck shows problems, run fsck manually (see Section 2.6).
	ls -rt /tmp	Check the */tmp* directory with **ls -rt** and remove any ancient files that are just taking up disk space. Autobooting does this automatically.
	dump 1un	Runs a ''level 1'' dump (8) of the selected file system (see Section 8.7). This saves all files changed since the last level 1 dump. Do this for all file systems.
Monthly	dump 0un	Runs a level 0 dump (8) of the selected file system (see Section 8.7). Do this for all file systems.
	uuclean	Cleans the */usr/spool/uucp* area.

8.5.1 Monitoring System Files

Table 8-8 shows system files that require periodic attention or that contain definitions specific to your site.

Table 8-8. Files to Be Maintained

File	Function
/etc/fstab	how disk partitions are used
/etc/disktab	disk partition sizes
/etc/sptab	spanned-disk configurations
/etc/printcap	printer data base
/etc/gettytab	terminal type definitions
/etc/remote	hostnames and phone numbers of remote systems for **tip**(1)
/etc/group	group memberships
/etc/motd	message of the day
/etc/passwd	password file; each account has a line
/etc/rc.local	local system restart script; runs reboot; starts daemons
/etc/hosts	hostname data base
/etc/networks	network name data base
/etc/services	network services data base
/etc/hosts.equiv	"trusted" hosts for remote logins
/.rhosts	"trusted" hosts for remote **root** access
/etc/host.lpd	hosts with remote access to local printers
/etc/ttys	defines terminal lines
/usr/lib/crontab	commands that are run periodically
/usr/lib/mail/aliases	mail forwarding and distribution groups
/usr/adm/acct	raw process account data
/usr/adm/messages	system error log
/usr/adm/shutdownlog	log of system reboots
/usr/adm/wtmp	login session accounting

8.5.2 Monitoring for Performance

In addition to the regimen in Table 8-7, you can monitor the system's activity with an eye to improving performance.

Use the **vmstat** (1) and **ps** (1) commands to show CPU and memory activity. Ideally, there should be

- few blocked (b) jobs
- little paging or swapping activity
- available bandwidth on disks
- high user CPU use (us)

If the system is busy, then the count of active jobs may be large, and several of these jobs may often be blocked (b). If the virtual memory is active, then the paging daemon will be running (sr will be non-zero). It is healthy for the paging daemon to free pages when the virtual memory gets active; it is triggered by the amount of free memory dropping below a threshold and increases its pace as free memory goes to zero.

Running **vmstat** (1) when the system is busy (type **vmstat 1** for a running display of everything) can show imbalances by noting abnormal job distributions. If many processes are blocked (b), then the disk subsystem is overloaded or imbalanced. If several non-DMA devices or open teletype lines are "ringing," or user programs are doing high-speed non-buffered input/output, then the system time may be high (60-70 percent or higher).

You can often pin down the cause of high system time by checking for excessive context switching (cs), interrupt activity (in), or system call activity (sy). Large systems may average about 60 context switches and interrupts per second and about 90 system calls per second.

If the system has a heavy load, or if you have little memory for your load (2M is little in most any case), then the system may be forced to page/swap. This causes a noticeable reduction in system performance and lengthy pauses (sometimes more than 5 seconds) when interactive jobs such as editors swap out. If you expect to be in a memory-poor environment for an extended period you might consider administratively limiting system load.

8.6 Accounting

UNIX optionally records two kinds of accounting information: connect time accounting and process resource accounting. The file */usr/adm/wtmp* holds connect time information; process this information with ac (8). The file */usr/adm/acct* holds process time data; use sa (8) to look at this information.

When charging for computing time, use the information provided by these commands. A convenient way to do this is to give commands to the clock daemon **cron** (8) to be executed every day at a specified time. Do this by adding lines to */usr/lib/crontab;* see **cron** (8) for details.

You can also set disk quotas for individual users with the **quotaon** (8) command. See **quotaon** (8) and **edquota** (8) for details.

8.7 Backing Up Your System

"Backing up" means putting copies of disk files onto magnetic tape for safe storage. Do this periodically by the schedule described in Section 8.5.

The *-n* option of **dump** (8) notifies people from the group operator when it needs attention. Add users to operator in the file */etc/group* (see group (5)). root should be a member of operator.

In a cluster or diskless network only the server node can perform dumps.

One good method for keeping safe backups on tape is the "Towers of Hanoi" dump sequence. This is a series of daily, weekly, and monthly dumps that ensures recent backups for all files.

Keep these backup tapes as long as you can before reusing them. Table 8-9 shows the Towers of Hanoi dump schedule. Table 8-10 shows a sample dump log with this schedule.

Table 8-9. Towers of Hanoi Dump Schedule

Period	Level	Explanation
Daily	3, 2, 5, 4, 7, 6, 9, 8, 9, 9, . . .	Each day, run a **dump** (8) of all file systems. The first day, run a "level 3" dump, the next day a level 2 dump, and so forth following the sequence at left. This saves the files changed each day. Stop the daily sequence when you do a weekly (level 1) dump, and restart on the next day at level 3 again.
Weekly	1	Each week, run a level 1 dump of all file systems. This saves all files changed since the last level 1 dump. Restart the daily sequence at level 3.
Monthly	0	Each month, run a level 0 dump of all file systems. This makes a complete copy of every file on the system.

Table 8-10. Sample Dump Log for Towers of Hanoi Schedule

Tape Label	Level	Date	Who	Size
FULL	0	Nov 24, 1986	cf	137K
Daily1	3	Nov 28, 1986	cf	29K
Daily2	2	Nov 29, 1986	km	34K
Daily3	5	Nov 30, 1986	km	19K
Daily4	4	Dec 1, 1986	km	22K
WEEK1	1	Dec 2, 1986	kh	40K
Daily5	3	Dec 4, 1986	km	15K
Daily6	2	Dec 5, 1986	cf	25K
Daily7	5	Dec 6, 1986	cf	15K
Daily8	4	Dec 7, 1986	km	19K
WEEK2	1	Dec 9, 1986	kh	118K
Daily9	3	Dec 11, 1986	km	15K
Daily10	2	Dec 12, 1986	km	26K
Daily1	5	Dec 15, 1986	km	14K
WEEK3	1	Dec 17, 1986	kh	71K
Daily2	3	Dec 18, 1986	kh	13K
FULL	0	Dec 22, 1986	kh	135K

The Procedure to Dump a File System shows how to run a dump (8). The Procedure to Restore Files From a Dump Tape shows how to retrieve files from a dump tape.

Procedure to Dump a File System

	ACTION	WHAT IT DOES
Step 1.	Boot down to single-user UNIX (the # prompt). See Section 8.3.	Forces all other users off the system. Running dumps while people are modifying the file system can cause all sorts of errors. Dumps are safest while in single-user mode.
Step 2.	Prepare a tape for the dump. Use either a blank tape or one with unneeded contents. Label the tape with today's date, the level of dump, and the file system you are dumping. If the dump will take more than one tape, number the tapes.	Prepares the tape for dumping and for archival storage. Make sure the label is legible; dump tapes do no good if you cannot find the one you need. The dump may require more than one tape. If so, the dump program will print this information when you begin the dump.
Step 3.	Load the tape in the tape drive. If necessary, use the command `mt rewind` to rewind the tape.	Readies the tape for the dump. If the system has more than one tape drive, use the **-f** option for **mt**(1) and **dump**(8) to specify another tape drive. The default is drive 0. For example, to rewind a tape on drive 1, enter `mt -f /dev/rmt13 rewind`
Step 4.	"Retension" the tape with the command `mt reten`	Winds the tape all the way forward and all the way back. This smooths out reel tension for the tape, lowering the chances of tape failure during the dump. If the dump requires more than one tape, retension all of the tapes now.
Step 5.	Enter the command dump *n*un /dev/r*dev* where *n* is the level number of the dump. *dev* is the disk partition holding the file system for dumping. Do not forget the /dev/r before *dev*.	Executes the dump. **dump** will print many messages on the console, including notification when the dump is complete. For example, to dump (at level 0) the */u* file system on the *g* partition of a CDC Wren 86 disk on an Optimum V system, enter `dump 0un /dev/rsd0g`
Step 6.	When the dump is finished, remove the tape and store it.	Archives the dump for future restores (**restore** (8)) if needed.

End of procedure

Procedure to Restore Files From a Dump Tape

	ACTION	WHAT IT DOES
Step 1.	Load the tape in the tape drive. If necessary, use the command **mt rewind** to rewind the tape.	Readies the tape for restoring. If the system has more than one tape drive, use the **-f** option for **mt**(1) and **restore**(8) to specify another tape drive. The default is drive 0. For example, to rewind a tape on drive 1, enter **mt -f /dev/rmt13 rewind**
Step 2.	Enter the command **restore xv** *file1 file2 ...* where *file1 file2 ...* is a space-separated list of the files you want to extract from the dump tape. If you want to restore a complete directory and all files and subdirectories under it, enter the command **restore xvh** *directory* where *directory* is the pathname of the directory. If you want to restore the entire file system, enter the command **restore rv**	Extracts the named files or directories from the dump tape. **restore**(8) should print the name of each file as it restores it. See **restore**(8) for details on this command. Use **restore rv** with caution, and only under these circumstances: • Writing to a new file system (created with **newfs**(8)) • Making incremental restores after a full level 0 restore
Step 3.	After finishing all restore commands enter the command **rm /restoresymtab**	Removes the system file used for incremental restore passes.
Step 4.	Enter the command **mt rewind**	Rewinds the tape.
Step 5.	Remove the tape and store it.	Saves the tape for later restores if necessary.

End of procedure

APPENDIX A: LOADING UNIX FROM TAPE

This appendix tells how to install UNIX onto the system from a release tape.

If you are using a system shipped complete from Integrated Solutions, do not use these procedures. The system is fully installed and ready to boot from disk. Turn to "Section 2: Starting UNIX From Disk."

If you want to install UNIX on a cluster or diskless network use these procedures to install the server node. After fully installing the server node turn to "Section 6: Setting Up Cluster Nodes" or "Section 7: Setting Up Diskless Nodes."

This appendix contains the following subsections:

A.1 Why Should You Boot From Tape?

''Bootstrapping,'' usually abbreviated to ''booting,'' means loading and executing a system program (usually the operating system, UNIX). With UNIX, full operation means that the system has a fully installed file system and a working shell (command processor).

With UNIX installed on disk, the system will be able to ''autoboot,'' to boot itself without help. Your task is to install UNIX on disk so that it can do so.

Use the procedures in this appendix to install UNIX under these conditions:

- Installing UNIX on a fresh disk.
- Some accident, such as a power failure, destroyed data on the disk and UNIX will not boot.

NOTE

> You can repair most disk damage without re-installing UNIX. See Section 2.6 and Appendix B.

A.2 Do You Have Everything You Need?

You should have a fully installed Optimum or Optimum V system, as described in the appropriate *Installation Manual*:

- All boards plugged into the card cage
- At least one disk connected
- A connected tape drive
- A terminal (or workstation monitor) connected to the console line
- System plugged into an AC wall outlet

You should know the revision level of the PROMs on your CPU board. In *Step 4* of the Procedure to Load and Boot the Miniroot File System, you will need to know if your PROM revision level is above or below Revision 4.0. The PROM displays its revision level in a message to the console (or WorkStation monitor) during a boot. Look for a line of this form:

VMEBUS*(QBUS) 680XX prom-level* boot:*date*

If you are in doubt, look on the board and read the revision level from a sticker on the PROM.

Find the name of the system's boot disk in Table A-1.

Table A-1. UNIX Disk Names

Product Name	Device	Controller Name
	Optimum V Systems and WorkStations	
VDS106 VED106	5 ¼-inch disks (SD-style) 106 Mbytes formatted	sd
VDS139 VED139	5 ¼-inch disks (embedded SCSI) 139 Mbytes formatted	gd
VDS269 VED269	8-inch disks (SMD-style), 269 Mbytes formatted	sm
VDS418 VED418	9-inch disks (SMD-style), 418 Mbytes formatted	sm
VDS494 VED494	8-inch disks (SMD-style), 494 Mbytes formatted	sm
	Optimum Systems	
DS36 ED36	5 ¼-inch disks, 36 Mbytes formatted	el
DS67 ED67	5 ¼-inch disks. 67 Mbytes formatted	el
DS85 ED85	5 ¼-inch disks, 85 Mbytes formatted	el
DS140 ED140	5 ¼-inch disks, 140 Mbytes formatted	el
DS330 ED330	SMD-style disks, 330 Mbytes formatted	hp
DS474 ED474	SMD-style disks, 474 Mbytes formatted	hp

Find the name of the system's tape drive in Table A-2.

Table A-2. UNIX Tape Drive Names

Device	Name
Optimum V Systems and WorkStations:	
All tape drives (except embedded SCSI)	ts
Embedded SCSI	gt
Optimum Systems:	
TS-11 tape drives	ts

Enter the PROM revision level, disk name(s), and tape drive name(s) in "Appendix C: Configuration Worksheet." Fill in the rest of Appendix C to the best of your knowledge. This will serve as an easy reference in later procedures.

Have these materials on hand:

- The appropriate Installation Manual (for example, the *Optimum V24 System/WorkStation Installation Manual*)
- The UNIX 4.3BSD Reference Set, seven volumes
- The ISI UNIX Release 4.0 release tapes (two tapes)

Table A-3 shows the format of the release tapes for Integrated Solutions Release 4.0 of UNIX.

All release tapes contain the files necessary to boot either an Optimum System or an Optimum V System or WorkStation.

The release tapes includes graphics software (files 14 and 15, tape 2, in Table A-3). If you do not have a graphics workstation, you do not need this software. The server node for a Cluster or Diskless WorkStation does need this software.

The release consists of two 1/4-inch cartridge tapes or two 1/2-inch reel-to-reel tapes. Note that this does not include the diagnostic tape, which is a separate tape for diagnostic purposes only.

Table A-3. Format of the Release 4.0 Release Tapes

File			
1/4-Inch Tape	1/2-Inch Tape	Contents	Description
(tape 1)0-8	(tape 1)0-8	miniroot	the "miniroot" file system, with install software
9	9	root	dump (8) of root (/) file system
10	10	kernels	tar (1) of kernels for the root file system
11	11	bin	tar of */usr/bin*
12	12	ucb	tar of */usr/ucb*
13	13	doc	tar of */usr/doc*
14	14	man	tar of */usr/man*
15	15	games	tar of */usr/games*
16	16	new	tar of */usr/new*
17	17	old	tar of */usr/old*
(tape 2)0	(tape 2)0	sys	tar of */usr/sys*
1	1	lib	tar of */usr/lib*
2	2	etc	tar of */usr/etc*
3	3	dict	tar of */usr/dict*
4	4	include	tar of */usr/include*
5	5	bench	tar of */usr/bench*
6	6	adm	tar of */usr/adm*
7	7	spool	tar of */usr/spool*
8	8	guest	tar of */usr/guest*
9	9	hosts	tar of */usr/hosts*
10	10	local	tar of */usr/local*
11	11	msgs	tar of */usr/msgs*
12	12	preserve	tar of */usr/preserve*
13	13	pub	tar of */usr/pub*
14	14	tmp	tar of */usr/tmp*
15	15	graph1	tar of graphics files in root file system
16	16	graph2	tar of graphics files in */usr* file system
17	17	X	tar of X window system files
18	18	UPDATE	tar of UPDATE script

A.3 Booting the Miniroot File System

Booting UNIX is a two-stage process:

1. Boot UNIX from the "miniroot". The miniroot is a subset of the root file system. Boot this rudimentary UNIX and use it to load the full root (/) file system and /usr file system.
2. Boot UNIX from the root file system. This installs the tools for a full boot to UNIX.

Use the following Procedure to Load and Boot the Miniroot File System to start the system, load the miniroot from tape, and boot the miniroot.

Procedure to Load and Boot the Miniroot File System

	ACTION	WHAT IT DOES
Step 1.	Turn ON power for the terminal connected as the system console (or the WorkStation monitor).	Prepares the terminal to print boot messages.
Step 2.	Turn the system power ON. Check that the DC ok indicator lights up.	Provides power to the system. After a short wait, this message should appear on the console: **Integrated Solutions** VMEBUS*(QBUS) 680XX prom-level* boot:*date* : The colon (:) is a prompt from the UNIX PROMs (see Section 1.2, "Introduction to Hardware").
Step 3.	Put the first of the two release tapes into the tape drive.	Prepares the tape.
Step 4.	Enter this command: ts@ exactly as shown. The "@" notation will not work with older PROM levels on the CPU board. See the information at right.	Begins installation of the miniroot from the release tape. For example, on an Optimum V system with a VME-68K20 CPU, enter ts@ This invokes a program called the Miniroot Installer. The console should print *xxx+xxx+xxx* start at 0x*wwww* --- *System-type PROM-level* Miniroot Installer --- PROMs previous to PROM Revision 4.0 will not execute the *ts@* command. See Section A.2, "Do You Have Everything You Need?" to determine your PROM revision. If you have a pre-4.0 PROM, you must issue a different command, based on your CPU board: VME-68K20: *ts*(0,0) VME-68K10: *ts*(0,3) IS-68K (Q-bus) *ts*(0,2)

(continued on next page)

Procedure to Load and Boot the Miniroot File System *(continued)*

	ACTION	WHAT IT DOES
Step 5.	Accept the defaults by pressing return or answer no to specify another partition.	
Step 6.	Enter the following response to the Miniroot Installer's prompt: *dev*(*n*,1) where *dev* is the name of the disk, from Table A-1 *n* is the number of the disk drive (0 for the first disk drive, 1 for the second disk drive, and so on) NOTE SMD-type (“sm”) disks are numbered 0 for the first one, 2 for the second SMD-type device, 4 for the third SMD-type device, and so on.	Copies the miniroot to disk and boots the miniroot. This command copies the miniroot to partition 1, also known as partition b, of the chosen disk. Remember that partition numbers, like file numbers on the release tape, start at 0. For example, for an Optimum V system with one SMD disk, enter sm(0,1) During a brief wait, you should see and hear the tape moving. The system now boots on the miniroot, ending with the single-user UNIX prompt (#): *xxx+xxx+xxx* **start at 0x*www*** **Integrated Solutions UNIX Release 4.0 #*n*** (root@*system-name*) *date* real mem = *xxx* avail mem = *yyy* *(information about devices)* . . . CTC System Clock *(or LTC)* # You are now in UNIX, booted from miniroot.
Step 7.	Continue to the next section, and execute the UNIX install script.	Completes the installation of UNIX, as described in the next section.

End of procedure

A.4 Booting the Complete Root File System

After booting from miniroot, the install script is automatically started.

Table A-4 shows the disk types for all disks used with Integrated Solutions VMEbus Systems. Do not confuse this with the disk "names" in Table A-1. "RLL encoded" means that the Optimum V systm uses Run Length Limit encoding to yield a higher formatted capacity.

Table A-4. Disk Types for VMEbus Systems

Disk Name	Disk Type
CDC Swallow PA5N2	vds418
	vcd418
	swallow
CDC Wren 36	wren36
RLL encoded	wren36L
CDC Wren 86	vds85
	vcd85
	wren86
	wren86n
RLL encoded	vds106
	vcd106
	wren86L
CDC 94161-9	cdc148
Fujitsu M2243	m2243
RLL encoded	m2243L
Fujitsu M2312	m2312
Fujitsu M2246SA	fuj134
Fujitsu M2284	m2284
Maxtor 1065	max1065
RLL encoded	max1065L
Maxtor 1085	max1085
RLL encoded	max1085L
Maxtor 1140	max1140
RLL encoded	max1140L
Micropolis 1375	mic139
NEC 2352	nec
Northern Telecom Mercury 8212X/33	vds269
	vcd269
	northtel
Northern Telecom Mercury 8312/56	vds494
	vcd494
Priam 807	priam
Tecstor 332	tecstor
Toshiba MK56F	tos86
RLL encoded	tos86L
Toshiba MK156FB	tos140
Vertex 170	vertex170
RLL encoded	vertex170L
Vertex 185	vertex185
RLL encoded	vertex185L

Use the following Procedure to Boot from the Root File System to install the root (/) and */usr* file systems, and to reboot from the new root kernel.

Procedure to Boot from the Root File System

	ACTION	WHAT IT DOES
Step 1.	After completing the previous procedure, you should now be in the install script.	The screen should show a message about the installation, beginning with the line ===4.3BSD UNIX INSTALL== and concluding with this prompt: **Do you want to continue with the installation? [y]:**
Step 2.	Provide information to the install script prompts.	Assists the install script in making the correct installation for your system. The install script follows the steps in the remainder of this procedure. When the install script is finished, you can reboot the system automatically, or exit the script and remain in the miniroot shell. Unless you have special configuration concerns, you should perform the reboot.
Step 3.	The install script prints this prompt: Do you want to continue with the installation? [y]:	Allows you to immediately exit the install script if desired. Press RETURN to continue with the installation.
Step 4.	The install script prints this prompt: Please enter the disk device name [*devN*]: where *devN* is a default value.	Prompts for the disk device name for the boot disk. UNIX provides a default value from autoconfiguration information. If this value is correct, simply press RETURN. If you want to supply another value, from the choices in Table A-1, enter the value and press RETURN.
Step 5.	The install script prints this prompt: Please enter the tape drive name [*devN*]: where *devN* is a default value.	Prompts for the tape device name of the tape drive. UNIX provides the default value from autoconfiguration information.
Step 6.	The install script prints this prompt: Do you want to set up any spandisks? [y]:	Prompts for setting up spandisks. Answering "yes" allows you to set up as many as four separate spandisks. If you answer "no", skip to Step 10.

(continued on next page)

Procedure to Boot from the Root File System *(continued)*

	ACTION	WHAT IT DOES
Step 7.	The install script prints the prompt: **Which spandisk partition do you want to set up? [*sp0c*]:**	Prompts for the first spandisk partition to be created. Allowable partitions are *sp0c*, *sp1c*, *sp2c*, and *sp3c*. You may enter any of these partitions or press RETURN to accept the default.
Step 8.	The install script prints this prompt: **Available partitions and sizes:** *partition size partition size* *partition size partition size* **Remember that partitions d, e, and f make up partitioning so they are mutually exclusive on any particular disk.** **Please enter all desired partitions on a single line...** where *partition* is *devNpart*, *dev* is disk device, *N* is disk number and *part* is the partition letter.	Prompts for the list of disk partitions to be used in creating the designated spandisk. Available partitions and their sizes are listed along with the prompt. Partition g is actually the combination of partitions d, e, and f so you cannot use partition g on any disk where partitions d, e, and f are already used. To use an entire disk, specify partitions *devNa*, *devNb*, *devNg*. Include partition *devNh* if applicable.
Step 9.	The install script prints this prompt: **Which spandisk partition do you want to set up? [none]:**	Allows you to set up other spandisks. Press RETURN to continue with the installation process. Steps 8 and 9 may be repeated for the three additional spandisks if desired.
Step 10.	The install script prints this prompt: **Will you be installing the /usr file system now? [y]:**	Prompts for installation of the */usr* file system. You should answer "yes" (y) if you have a disk partition ready for the */usr* file system. Answer "no" (n) if you must prepare a disk partition before installing the */usr* file system. See the procedure in Step 30 for instructions to use the */installusr* and */installgraphics* scripts.
Step 11.	The install script prints this prompt: **Do you want to remove the kernels and kernel configuration directories for alternate configurations? [y]:**	Prompts for removal of possibly extraneous directories and kernels. The release tapes includes configuration files for all ISI systems. You may only need the configuration files for your own system. If so, answer "yes" (y) to this question. Servers for diskless or cluster nodes should retain the configuration files, since they may need to provide kernels for the individual nodes.

(continued on next page)

Procedure to Boot from the Root File System *(continued)*

	ACTION	WHAT IT DOES
Step 12.	The install script prints this prompt: Do you want /usr/doc (approx. 6450 kbytes) installed? [y]:	The default is to install the files.
Step 13.	The install script prints this prompt: Do you want /usr/games (approx. 2150 kbytes) installed? [y]:	The default is to install the files.
Step 14.	The install script prints this prompt: Do you want /usr/man (approx. 3300 kbytes) installed? [y]:	The default is to install the files.
Step 15.	The install script prints this prompt: Do you want the graphics files extracted from the tape? [y]:	Prompts for graphics software extraction. Answering "yes" (y) to this question means that the install script will install graphics software onto your system. All WorkStations and servers for graphics cluster or diskless nodes should do this.
Step 16.	The install script prints this prompt: Is the console a graphics workstation? [y]:	This question appears only if you answered "yes" (y) to the question above. Answering "yes" (y) to this question causes the install script to alter the file */etc/ttys* to accommodate the workstation monitor.
Step 17.	The install script prints this prompt: Do you want to install the X windows files? [y]:	The default is to install the files.
Step 18.	The install script prints the prompt: Usr files are about *N* kbytes. Available disk partitions are: *partition size* *partition size* Enter the disk partition to install /usr onto [*devNpart*]:	Prompts for placement of the */usr* file system. The normal placement, the default here, is on the first partition of the list of partitions large enough for the files.

(continued on next page)

Procedure to Boot from the Root File System *(continued)*

	ACTION	WHAT IT DOES
Step 19.	The install script prints the message: System configuration is *SYS*. (This is not alterable.) Disk Drive name is *diskdev*. Tape drive name is *tapedev*. Making spandisks: partition ... Installing /usr file system on *devNpart*. Removing alternate kernels and kernel configurations: *response* Installing /usr/doc: *response* Installing /usr/games: *response* Installing /usr/man: *response* Installing graphics files: *response* Console is graphics workstation: *response* Installing X windows: *response* Is this setup correct? [y]:	*response* is the information you entered to earlier prompts. If you answered no to the installing */usr* prompt, the remaining message lines are not printed.
Step 20.	The install script makes tape devices on the miniroot.	Makes devices required by the remainder of the scripts.
Step 21.	The install script creates a root directory.	Provides an empty root file system for the contents of the release tape. The install script creates the file system and checks it with fsck (8).
Step 22.	The install script installs the root file system.	Extracts the root file system from the release tape and installs it on your system. The install script checks the new file system again with fsck (8).
Step 23.	The install script creates devices in the */dev* directory.	Provides disk and tape devices. For workstations, this step also provides the graphics console device.

(continued on next page)

Procedure to Boot from the Root File System *(continued)*

	ACTION	WHAT IT DOES
Step 24.	The install script edits the file */etc/fstab* to match this configuration.	Configures */etc/fstab* for your system. See fstab (5) for detail on */etc/fstab* entries. If you see an error message at this point, you should edit */etc/fstab* manually after the end of this procedure.
Step 25.	The install script creates a */usr* directory.	Provides an empty */usr* file system for the contents of the release tapes. The install script creates the file system and checks it with fsck (8).
Step 26.	The install script installs the */usr* file system.	Extracts the */usr* file system from the release tapes and installs it on your system. The */usr* file system is split across the two release tapes. When the script has extracted everything from the first tape, it will prompt you to change tapes.
Step 27.	The install script installs graphics software, if you requested it in *Step 8*.	Extracts graphics files from the release tape.
Step 28.	The install script prints this prompt: Set root password New password: Type a new password in response to the prompts.	Assigns the root password. Write it down somewhere safe. Because the root login account has unlimited access to the entire system, you must protect that access by assigning a root password and by keeping knowledge of the password limited.
Step 29.	The install script finishes by rewinding the release tape and printing a prompt to reboot the system.	Finishes the installation and prepares for reboot. Unless you have special configuration requirements that would prevent a reboot, you should now reboot the system. Simply press RETURN at the final install script prompt. If you need to make a special disk configuration for */usr*, do it here and run */installusr* and */installgraphics*.

(continued on next page)

Procedure to Boot from the Root File System *(continued)*

ACTION		WHAT IT DOES
Step 30.	If you did not install the */usr* file system from tape, you must prepare a disk partition now, and extract the */usr* file system.	Installs the */usr* files system from the release tapes. To install the */usr* file system, use the */installusr* script: 1. Create or configure a disk partition for the */usr* file system. 2. **mount** (8) the file system. 3. Insert the first of the two release tapes into the tape drive. 4. Enter the command **/installusr** 5. Follow the instructions of the script. The instructions are the same as previously described in this procedure. If you want graphics files as well, use the */installgraphics* script: 1. Check that the */usr* file system is installed and mounted. 2. Insert the second of the two release tapes into the tape drive. 3. Enter the command /installgraphics 4. Follow the instructions of the script. The instructions are the same as previously described in this procedure.

(continued on next page)

Procedure to Boot from the Root File System *(continued)*

	ACTION	WHAT IT DOES
Step 31.	After running the install script, reboot the system. Enter this command at the PROM prompt (:): *dev*(*n*,0)vmunix where *dev* is the name of the disk, from Table A-1 *n* is the number of the disk	Boots UNIX from the kernel file */vmunix* in the root file system. For example, for an Optimum V WorkStation with a VDS269 disk subsystem, enter sm(0,0)vmunix These messages should appear: *xxx+xxx+xxx* start at 0x*wwww* Integrated Solutions UNIX Release 4.0 #*n* (root@*system-name*) *date* real mem= *(physical memory)* avail mem= *(memory for user files)* Using *xx* buffers containing . . . (information about devices) . . . root on *dev*0a, dump on *dev*0b, args on *dev*0b, swap on *dev*0b CTC System Clock *(or LTC)* # You are now running single-user UNIX off the installed root file system. Continue to Section A.5 to finish installing the software.

End of procedure

A.5 What Now?

Use the following procedure, the Procedure to Clean Up After Installing UNIX, to clean up loose ends from the installation procedure in the previous section.

After booting from the root file system (*dev*(0,0)**vmunix**) the console now shows the single-user UNIX prompt (#).

Procedure to Clean Up After Installing UNIX

	ACTION	WHAT IT DOES
Step 1.	Enter the command `chmod 755 / /usr`	**chmod** (1) sets protections and permissions for the root (/) and */usr* directories. The mode 755 denies write permission to everyone but **root**, but opens the directories for reading and searching by anyone.
Step 2.	Check that the */dev* directory contains special files for all of the devices attached to the system.	Checks for proper configuration. See Section 4.6, "Making New Devices in */dev*," to add devices.
Step 3.	Turn to "Appendix C: Configuration Worksheet" and write down any changes.	Updates the reference sheet.
	End of procedure	

You have now installed UNIX. Reboot the system with **reboot** (8) and continue to "Section 3: Setting Up Your System."

APPENDIX B: USING DIAGNOSTIC PROGRAMS

Use these diagnostic programs to test the CPU, the memory board, and the disks.

For more information on repairing disk blocks, see ISI Technical Bulletin 001, "Disk Bad Tracks and Blocks."

This appendix contains the following subsections:

B.1 When to Run Diagnostics
B.2 Running the Diagnostics
B.3 Using **copy**
B.4 Using **mmu**
B.5 Using **mem**
B.6 Using **dma**
B.7 Using **bad144**
B.8 Using **diskformat**
B.9 Using **gdbad**

B.1 When to Run Diagnostics

Use these diagnostic programs only if you suspect hardware failures. For example, run diagnostics in response to these symptoms:

- System will not boot
- Disk errors that fsck (8) cannot correct
- System crashes frequently

B.2 Running the Diagnostics

The diagnostic tests are on the Integrated Solutions diagnostic release tape, separate from the standard UNIX release tape. Thus, you can load them directly from the tape if, for example, you have been unable to load UNIX on the system and you suspect a hardware problem.

Always run the diagnostics in "stand-alone" mode. This means that the console displays the PROM prompt (:). See Section 8.3, "When Do You Boot Single-User?" to reach the PROM prompt.

To run diagnostics on a cluster or diskless node, you must first install the diagnostic programs on the server's UNIX file system. See Section B.2.2, "Running Diagnostics from UNIX." To run diagnostics on a cluster node see Section 6.6, "Booting Single-User on Cluster Nodes." To run diagnostics on a diskless node see Section 7.6, "Booting Single-User on Diskless Nodes." The only diagnostic tests appropriate to these nodes are **mmu** and **mem**.

B.2.1 Running Diagnostics From Tape

Use the Procedure to Run Diagnostic Programs From Tape to load a diagnostic program from the diagnostic tape and execute the program.

Procedure to Load Diagnostic Programs From Tape

	ACTION	WHAT IT DOES
Step 1.	Check that the console displays the PROM prompt (:).	Checks for the right mode for diagnostics.
Step 2.	Enter this command: ts@ exactly as shown.	Loads the diagnostic interface from the tape. This "Diagnostics Executive" is a menu from which you choose the program you need. The "@" notation works with CPU boards with PROM Revision 4.0 or higher. See Section A.2, "Do You Have Everything You Need?" to determine your PROM revision. If the "@" notation does not work, or you have pre-4.0 PROMs, use one of the following commands, based on your CPU board: • VME-68K20: ts(0,0) • VME-68K10: ts(0,3) • IS-68K (Q-bus): ts(0,2) The screen should now show a menu of diagnostic programs. Standalone Diagnostic Main Menu --- <SYS TYPE> <VERSION> Standalone Diagnostics Executive --- a) copy b) bad144 c) diskformat d) mmu e) mem f) dma g) gdbad Enter letter of desired test [a-g]: <CNTL>-D exits to boot prompt from the menu level <CNTL>-D exits to the menu from a diagnostic
Step 3.	Select a program from the menu. Enter the letter for the program and press RETURN.	Loads the program from tape and begins execution.

End of procedure

B.2.2 Running Diagnostics From UNIX

You can load diagnostic programs from the diagnostic tape to the UNIX file system. The diagnostics can be booted from disk rather than tape. Use this method under either of these circumstances:

- You will run diagnostics on a cluster or diskless node. These nodes cannot load diagnostics from the tape drive.
- You will be running the diagnostics often.

Use the following Procedure to Install Diagnostics in UNIX to extract diagnostic programs from the diagnostic tape and install them in the UNIX file system. Perform this procedure on the server node of a cluster or diskless network.

Procedure to Install Diagnostics in UNIX

	ACTION	WHAT IT DOES
Step 1.	Boot the system to single-user or multiuser UNIX.	Prepares the system for tape transfer. You cannot install diagnostic programs in the UNIX file system from the PROM prompt.

Step 2.	Enter these commands:
	`mt rew`
	`mt fsf 10`
		Positions the tape for extraction of the programs. The **mt fsf** command winds the tape past the PROM-executable programs at the beginning of the tape.

Step 3.	Enter the commands:
	`cd /usr/stand`
	`tar xv`
		Extracts the diagnostic programs from the tape, installing them in the directory */usr/stand.* This command extracts programs for all ISI products; in the next step, you can remove those programs that do not apply to your system.

Step 4.	Enter the command: `ls`	Lists the contents of the */usr/stand* directory. These subdirectories contain the diagnostic programs appropriate to different ISI systems, where */usr/stand/V20* is for the VME-68K20 CPU */usr/stand/V10* is for the VME-68K10 CPU */usr/stand/Q10* is for the IS-68K CPU (Q-bus) NOTE For the server node of a cluster or diskless network, you should leave these extra directories in place, so that other nodes can access the diagnostics for their own CPUs. At this time, if you choose to do so, you can remove the directories that do not apply to your system. Use the command **rm -r** *dir1 dir2* where *dir1* and *dir2* are the unneeded directories.

(continued on next page)

Procedure to Install Diagnostics in UNIX *(continued)*

	ACTION	WHAT IT DOES
Step 5.	To execute a program in this directory, boot to the PROM prompt and enter the command *dev*(0,6)stand/*SYS*/*test* where *dev* is the name of the disk drive, from Table A-1 *SYS* is your system type, one of • V20 for the VME-68K20 • V10 for the VME-68K10 • Q10 for the IS-68K (Q-bus) *test* is the name of the diagnostic test	Loads a selected program and begins execution. To execute these programs from cluster or diskless nodes, see Section 6.6, "Booting Single-User on Cluster Nodes," and Section 7.6, "Booting Single-User on Diskless Nodes." CAUTION Execute diagnostic programs in this directory from the PROM prompt only. DO NOT execute these programs from single-user nor multiuser UNIX. For example, to execute the **diskformat** program on an Optimum V WorkStation with a VME-68K20 board and a VDS418 disk drive, enter : sm(0,6)stand/V20/diskformat (The “:” is the PROM prompt.) The (0,0) refers to the first partition of the first SMD-style disk (numbered from 0, remember), also known as partition *a*, holding the root (/) file system.

End of procedure

B.3 Using copy

The standalone copy program is typically used to copy files from tape to disk. Copy prompts for the FROM device and the TO device and copies all blocks until EOF is reached.

After loading this test and beginning execution, the screen displays

```
--- <SYS TYPE> <VERSION> Standalone Copy ---

From: <Source Device>
TO: <Destination Device>
```

B.4 Using mmu

Test the CPU before testing the memory, as CPU errors may cause random errors in the memory tests. **mmu** is a memory management unit test that tests only the CPU.

After loading this test and beginning execution, the screen displays

```
--- <SYS TYPE> <VERSION> Standalone MMU Diagnostics ---

Test Functionality
a) PTP
b) TBUF_SYS
c) TBUF_SYS
d) ALL
(Enter letter of functionality):
```

Enter c. **mmu** now displays

```
Tests available:
a) Patterns
b) Walk
c) Ping-pong
d) Uniqueness
z) All tests
Enter letter of desired test (H for Help, X for region redefinition):
```

Select Option z, which will prompt

```
Repeat count in decimal (0 forever :
Verbose (no):
```

Enter 1 (only one lap is necessary). The system will then display

```
Lap 1
Pattern test - segment register
Walking test - segment register
Ping-pong test - segment register
Uniqueness test - segment register
Pattern test - page register
Walking test - page register
Ping-pong test - page register
Uniqueness test - page register
```

This takes about four minutes. You can return to the stand-alone PROM prompt by pressing x twice. If you see errors, please try the test again, record all messages on the screen, and contact your distributor or Customer Support at Integrated Solutions.

B.5 Using mem

If **mmu** shows no errors, run **mem** to check the memory. When this test has been loaded, the screen displays

```
       --- <SYS TYPE> <VERSION> Standalone Memory Diagnostics ---

  Test region definition:
    Test all memory (default 'yes'):
```

Answer y, and the system will respond

```
  Enter number of memory banks in decimal (1 to 2047; 0 for autosizing):
```

Entering an incorrect number generates a TRAP ERROR; reboot and restart the program. Select automatic sizing; the system will then tell how many banks it found and display

```
  Found X memory banks

  Verbose? (default no):
  Test as a Byte? (default no):
  Test as a Short? (default yes):
  Test as a Long? (default no):
  Test with parity? (default no):
```

Use the default (press RETURN).

```
  Tests Available
  a) Patterns
  b) Walk
  c) Ping-pong
  d) Exec
  e) Uniqueness
  z) All tests
  Enter letter of desired test ('H' for Help, 'X' for region redefinition):
```

Enter e for the uniqueness option. It takes about one-and-one-half minutes per lap and should be run for three laps. To stop the test, hit the x key once; this returns to the "test all memory" section of the program. Another x returns to the PROM prompt (:). Report errors as described above.

The other options are for lengthy Integrated Solutions testing only; ignore them.

B.6 Using dma

This test generates random length DMA transfers through any memory range desired.

CAUTION

> This is a data destructive test! It will overwrite data in the specified test region.

When the dma diagnostic has been loaded, the console displays

```
          --- <SYS TYPE> <VERSION> Standalone DMA Test ---
device:
```

Enter the device name, *dev*(*m*,0) where *dev* is the name of the device you want to test and *m* is the device number (starting from 0).

```
Block size (default 512 bytes):
```

Select the default (press RETURN).

```
Number of blocks (default 1000):
```

Select the default (enter RETURN).

```
Start of buffer area (NO DEFAULT):
```

Enter 0x20000(h) for the beginning test area.

```
End of buffer area (NO DEFAULT):
```

Enter the hex value for the high end of memory. For example, enter 0x200000 for 2 Mbytes of total memory. The buffer must be at least two times the number of bytes to be read.

```
Number of laps (0 means forever: 1 default):
```

Enter the default (press RETURN).

```
Done DMA Test: <results are given>
```

B.7 Using bad144

The **bad144** program builds a table of disk bad blocks and their replacements for use by the remaining stand-alone programs and by UNIX 4.3BSD.

bad144 implements a *software* remapping of bad blocks and is independent of the *hardware* remapping of bad tracks implemented on any disk controller. A maximum of 126 bad blocks per logical disk can be mapped out by **bad144. bad144** is not a disk formatting program; it assumes that the disk has already been formatted.

Unlike the other diagnostic programs, **bad144** is available at the UNIX level as described in **bad144**(8). The UNIX level version of the program is designed for looking at the bad block table on the disk and not remapping bad blocks. Use the stand-alone version, as described here, for that.

If you are using a system shipped complete from Integrated Solutions, you should not run **bad144** (or any of the other stand-alone programs) now unless you have encountered disk problems.

After loading **bad144** the console displays

```
             --- <SYS TYPE> <VERSION> Standalone Bad144 Program ---
device:
```

Enter the device name, *dev*(*m*,0) where *dev* is the name of the disk and *m* is the disk number (starting from 0).

bad144 now prompts with

```
Options are
    1) Display current bad block information
    2) Add bad blocks
    3) Do read of disk
    4) Do write/read/verify of disk
        (this option destroys disk contents)
    5) Clear all bad block information
        (this option may cause bad blocks to 'appear' on disk)
    6) Do read/write/read/sector of disk
Enter number of desired option or (CR) to stop:
```

Select Option 1 to display the current bad block information. This produces output like

```
sn = 16570, cn = 139, tn = 1, sn = 12
```

The numbers are

• sn	Physical sector number
• cn	Cylinder number
• tn	Track number
• sn	Sector number within the cylinder/track

Record this information.

After formatting a disk, run Option 4. Running Option 4 on a disk that has not been freshly formatted is fine with **bad144**, but *will destroy the data on that disk!* Be very careful about using Option 4. Run Option 3 if you do not want to risk losing data.

Once you have decided to run the write/read verify, the dialogue might proceed as follows:

```
WARNING: this test destroys disk contents
Do you want to proceed? (y or n) y

Starting block (default 1):
Ending block (default end of disk):
Number of laps (0 or CR means forever): 6
begin lap 1
<running count of disk blocks>
end lap 1
checked blocks 1 through 131630
0 bad blocks this lap
0 bad blocks on disk
...
begin lap 6
<running count of disk blocks>
end lap 6
checked blocks 1 through 131630
0 bad blocks this lap
2 bad blocks on disk
```

The lap on which **bad144** found the two bad blocks is not shown; in the process of finding bad blocks, **bad144** produces a multitude of disk error messages, which would be tiresome to reproduce here.

In this example we have run six laps of **bad144**. That is because **bad144** writes six different patterns to the disk. Six laps can take a long time on a large disk; depending on the type of disk, **bad144** typically can write and verify from 7000 to 14000 blocks per minute. (Speeds can exceed 30000 blocks per minute on the read-only test.) Verify the entire disk, unless you have reason to suspect trouble in a particular area.

Sometimes it will seem obvious from the number of soft errors on a block that the block is marginal, but not quite bad enough for **bad144** to identify. It is possible to force the program to accept this block as bad: use Option 2. The program will respond

```
Enter bad block number, or (CR) if all have been entered
```

Enter the number of the marginal block. The program will keep asking for more blocks until it receives a bare carriage return.

The final option is primarily useful if you have made an error in using Option 2; it wipes all the bad block information from the disk and allows you to start over.

CAUTION

> If you have had bad blocks on the disk, have been running UNIX, and then use Option 5 to clear the bad block information and Option 2 to re-enter the bad block numbers, you will not necessarily get the same replacement blocks; the data, therefore, may be garbled.

B.8 Using diskformat

Be sure to use the proper version of **diskformat** for your system. The **diskformat** for Optimum systems is not the same as the **diskformat** for Optimum V systems.

Use this program to

- reformat your disk
- display parameter and defect lists

CAUTION

> Try to correct problems with **bad144** before reformatting your disk. Reformatting wipes out all data on the disk.

If you must reformat, first save users' files on tape. You can then restore these files after reformatting and rebuilding UNIX on the disk.

Table B-1 shows characteristics for EL disks (Optimum systems). Table B-2 shows characteristics for SD disks, while Table B-3 shows characteristics for SMD disks.

Table B-1. Parameters for EL Disk Drives with **diskformat**

Disk	Number of Heads	Number of Cylinders	Formatting Constant
DS140/ED140	15	918	7395
DS418/ED418	24	711	—
DS474/ED474	20	842	—
Fujitsu M2243	11	754	52F1
CDC Wren 36	5	697	22B8
CDC Wren 86	9	918	4395
Maxtor 1065	7	918	3395
Vertex 170	7	987	33DA
Vertex 185	7	1166	348D

Table B-2. Parameters for SD Disk Drives with **diskformat**

Disk	Number of Heads	Number of Cylinders	Formatting Constant
VDS85	9	918	4395
VDS106	9	918	4395
Fujitsu M2243, RLL encoded	11	754	52F1
CDC Wren 36	5	697	22B8
RLL encoded	5	697	22B8
Maxtor 1065	7	918	3395
RLL encoded	7	918	3395
Maxtor 1140	15	918	7395
RLL encoded	15	918	7395
Vertex 170	7	987	33DA
RLL encoded	7	987	33DA
Vertex 185	7	1166	348D
RLL encoded	7	1166	348D

Table B-3. Parameters for SMD Disk Drives with **diskformat**

SMD-type	Number of	Number of	Sectors	Cylinder	Size of		Interleave
Disk	Heads	Cylinders	/Track	Skew	Gap 1	Gap 2	Factor
VDS269	12	1332	33	0	16	16	1
VDS418	24	711	48	0	16	16	1
VDS494	12	1439	56	0	12	12	1
CDC PA1A3	16	1024	80	0	16	16	1
CDC 9720	10	1217	48	0	16	16	1
Fujitsu Eagle	20	842	44	0	16	16	1
Fujitsu M2312	7	589	32	0	16	16	1
Fujitsu M2333	10	823	64	0	16	32	1
Priam 807	11	1552	32	0	16	16	1
NEC 2352	19	760	56	0	16	16	1

Use these procedures to reformat disks with **diskformat**:

- Procedure to Reformat EL Disks with **diskformat**
- Procedure to Reformat SD Disks with **diskformat**
- Procedure to Reformat SMD Disks with **diskformat**

Procedure to Reformat EL Disks with diskformat

	ACTION	WHAT IT DOES
Step 1.	Shut down the system and turn power OFF.	Turns off the system. See Section 2.4, "Shutting Down The System."
Step 2.	Remove the RL101 Disk Controller board. Set DIP SW 1 to the ON position. Replace the RL101 Disk Controller board.	Enables disk formatting. This hardware switch prevents accidental formatting of the disk.
Step 3.	Turn ON power to the system, and press RESET to get the PROM prompt (:).	Displays the PROM prompt for loading diagnostic programs.

(continued on next page)

Procedure to Reformat EL Disks with diskformat *(continued)*

ACTION	WHAT IT DOES
Step 4. Load the **diskformat** program.	Starts the program. See Section B.2 for ways to load **diskformat**. Check that you are using the **diskformat** version for Optimum systems. The console now displays **--- <SYS TYPE> <VERSION> Standalone Diskformat ---** **Functions available:** **a) Format Disk** **b) Display Paramater and Defect list** **Enter letter of desired function ('H' for Help):**
Step 6. Enter the letter **a** and press RETURN.	Selects **Option a** from the menu, to format the disk. The console now displays Entering H simply reprints the menu. To exit the program press ^D (CNTL-D). **diskformat** now prompts for a description of the disk.
Step 7. Enter this information to **diskformat**'s prompts: • Controller number (0 or 1) • Drive number (remember that the first disk is 0) • Number of heads (see Table B-1) • Number of cylinders (see Table B-1)	Describes the disk for **diskformat**. **diskformat** now displays the estimated time it will take to complete the formatting. It also displays a "formatting constant" for your approval: **formatting constant =** *xxx* **OK?** where *xxx* is the formatting constant computed by **diskformat** based on the information you just entered.
Step 8. Check that **diskformat**'s formatting constant matches the one in Table B-1 for the disk. If it does, enter the letter **y** and press RETURN.	Begins reformatting the disk. **diskformat** displays the current cylinder being formatted. When **diskformat** finishes with the disk, it displays the initial menu again. Choose an option, or press ^D to quit.
Step 9. After formatting the disks, turn OFF power to the system. Remove the RL101 Disk Controller board. Set DIP SW 1 to the OFF position. Replace the RL101 Disk Controller board.	Disables disk formatting. This hardware switch prevents accidental reformatting of the disk.
Step 10. Turn ON power to the system. You can now use the reformatted disk.	Brings the system back up with the clean disk(s). If you reformatted the disk that contains the UNIX bootstrap code, you must now install UNIX from a release tape. See Appendix A. Restore user files saved on tape before running **diskformat**.

End of procedure

Procedure to Reformat SD Disks with diskformat

	ACTION	WHAT IT DOES
Step 1.	Shut down the system and turn power OFF.	Turns off the system. See Section 2.4, "Shutting Down The System."
Step 2.	Remove the VME-SCSI Host Adapter board. Set Bit 4 of SW1 to the ON position. Replace the VME-SCSI Host Adapter board.	Enables disk formatting. This hardware switch prevents accidental formatting of the disk.
Step 3.	Turn ON power to the system, and press RESET to get the PROM prompt (:).	Displays the PROM prompt for loading diagnostic programs.
Step 4.	Load the **diskformat** program.	Starts the program. See Section B.2 for ways to load **diskformat.** The console now displays **--- <SYS TYPE> <VERSION> Standalone Diskformat ---** **Select drive type to be formatted:** **a) sd** **b) sm** **c) gd** **d) rf** NOTE: VME **diskformat** offers options for **gd** (embedded SCSI disk drives) and **rf** (ESDI disk drives) that are currently unsupported.
Step 5.	Enter the number 1 and press RETURN.	Selects formatting for an SD-type drive. The console now shows **Functions available:** **a) Format Disk** **b) Display Parameter and Defect List** **Enter letter of desired function ('H' for help):**
Step 6.	Enter the letter a and press RETURN.	Selects Option a from the menu, to format the disk. Entering H simply reprints the menu. To exit the program press ^D (CNTL-D). **diskformat** now prompts for a description of the disk.

(continued on next page)

Procedure to Reformat SD Disks with diskformat *(continued)*

	ACTION	WHAT IT DOES
Step 7.	Enter this information to **diskformat**'s prompts: • Drive number (remember that the first disk is 0) • Number of heads (see Table B-2) • Number of cylinders (see Table B-2)	Describes the disk for **diskformat**. **diskformat** now displays the estimated time it will take to complete the formatting. It also displays a "formatting constant" for your approval: **formatting constant** = *xxx* OK? where *xxx* is the formatting constant computed by **diskformat** based on the information you just entered.
Step 8.	Check that **diskformat**'s formatting constant matches the one in Table B-2 for the disk. If it does, enter the letter y and press RETURN.	Begins reformatting the disk. **diskformat** displays the current cylinder being formatted. When **diskformat** finishes with the disk, it displays the initial menu again. Choose an option, or press ^D to quit.
Step 9.	After formatting the disks, turn OFF power to the system. Remove the VME-SCSI Host Adapter board. Set Bit 4 of SW1 to the OFF position. Replace the VME-SCSI Host Adapter board.	Disables disk formatting. This hardware switch prevents accidental reformatting of the disk.
Step 10.	Turn ON power to the system. You can now use the reformatted disk.	Brings the system back up with the clean disk(s). If you reformatted the disk that contains the UNIX bootstrap code, you must now install UNIX from a release tape. See Appendix A. Restore user files saved on tape before running **diskformat**.

End of procedure

Procedure to Reformat SMD Disks with diskformat

ACTION	WHAT IT DOES
Step 1. Boot to the PROM prompt (:).	Displays the PROM prompt for loading diagnostic programs. See Section 8.3, "When Do You Boot Single-User," for instructions to move between UNIX modes.
Step 2. Load the **diskformat** program.	Starts the program. See Section B.2 for ways to load **diskformat.** The console now displays --- <SYS TYPE> <VERSION> Standalone Diskformat --- Select drive type to be formatted: 1) sd 2) sm c) gd d) rf NOTE: VME **diskformat** offers options for **gd** (embedded SCSI disk drives) and **rf** (ESDI disk drives) that are currently unsupported.
Step 3. Enter the number 2 and press RETURN.	Selects formatting for an SMD-type drive. The console now displays Functions available: a) Format Disk b) Read UIB c) Modify UIB Enter letter of desired function ('H' for help):
Step 4. Enter the letter a and press RETURN.	Selects Option a from the menu, to format the disk. Entering H simply reprints the menu. Entering b prints the User Interface Block, containing the current disk parameters. To exit the program press ^D (CNTL-D). **diskformat** now prompts for a description of the disk.

(continued on next page)

Procedure to Reformat SMD Disks with diskformat *(continued)*

	ACTION	WHAT IT DOES
Step 5.	Enter this information to **diskformat**'s prompts: • Drive number (remember that the first disk is 0) • Start head volume 0 (enter 0) • Number of heads volume 0 (see Table B-3) • Start head volume 1 (enter 0) • Number of heads volume 1 (enter 0) • Number of cylinders (see Table B-3) • Number of sectors per track (see Table B-3) • Amount of cylinder skew (see Table B-3) • Size of gap 1 (see Table B-3) • Size of gap 2 (see Table B-3) • Sector Interleave Factor (see Table B-3)	Describes the disk for **diskformat. diskformat** now displays the estimated time it will take to complete the formatting. "Volumes" refer to splitting your physical disk into two logical disks. If you have not done this, answer 0 to all questions concerning "volume 1." **diskformat** displays the current cylinder being formatted. When **diskformat** finishes with the disk, it displays the initial menu again. Choose an option, or press ^D to quit. If you reformatted the disk that contains the UNIX bootstrap code, you must now install UNIX from a release tape. See Appendix A. Restore user files saved on tape before running **diskformat.**

End of procedure

B.9 Using gdbad

The gdbad program maps bad tracks to the bad tracks table. It is used for SCSI disks only. After you select this diagnostic, you are prompted to enter the controller number, target device number, and logical unit number (lun).

APPENDIX C: CONFIGURATION WORKSHEET

Use this worksheet to note your system configuration and any changes you make.

Item		Information			Example
System name		enter:	hostname:		central24
Model		enter:	type of system:		Optimum V24S
PROMs		enter:	revision level:		Revision 4.0
Disks	#1	enter:	model:	________	VDS106
		enter:	name (Table A-1):	________	sd0
		enter:	type (Table A-4):	________	vds106
		enter:	kernel driver:		SD0
	#2	enter:	model:	________	VED106
		enter:	name (Table A-1):	________	sd1
		enter:	type (Table A-4):	________	ved106
		enter:	kernel driver:		SD0
	other		under Comments		
Tape	#1	enter:	model:	________	¼" Flashback
		enter:	name (Table A-2):	________	ts0
		enter:	kernel driver:		TS0
	#2	enter:	model:	________	½" CDC 6250
		enter:	name (Table A-2):	________	ts1
		enter:	kernel driver:		TS1
	other		under Comments		
Serial ports	#1	enter:	model:	________	VME-ICP16/X
		enter:	name (Table 4-3):	________	cp0
		enter:	number of ports:	________	16
		enter:	kernel driver:		CP0
	#2	enter:	model:	________	VME-ICP8/X
		enter:	name (Table 4-3):	________	cp1
		enter:	number of ports:	________	8
		enter:	kernel driver:		CP1
	other		under Comments		

(continued on next page)

Configuration Worksheet *(continued)*

Item		Information		Example
Modems	#1	enter:	model: ________________	Racal-Vadic
		enter:	baud rate: ________________	1200
		enter:	port:	ttyda
	#2	enter:	model: ________________	
		enter:	baud rate: ________________	
		enter:	port:	
	other		under Comments	
Software release			ISI release 4.0	
Released on		(circle)	disk 1/4" tape 1/2" tape	¼" tape
Installed		enter:	date: ________________	(first boot)
		enter:	who:	(your name)
Ethernet		(circle)	none VME-ECX Interlan Excelan serial	VME-ECX
Networks		describe:	List networks connected; give hostname and address of your system. For diskless or cluster networks list hostnames and addresses of all nodes.	Diskless network: central24 89.0.1.96 ws1 89.0.1.97 ws2 89.0.1.98

Comments:

APPENDIX D: RELEASE NOTES

Please insert behind this page any release notes that you receive for your system.

Integrated Solutions

AN NBI COMPANY

DOCUMENTATION COMMENTS

Please take a minute to comment on the accuracy and completeness of this manual. Your assistance will help us to better identify and respond to specific documentation issues. If necessary, you may attach an additional page with comments. Thank you in advance for your cooperation.

Manual Title: *4.3BSD System Administrator's Guide* Part Number: *490197 Rev. C*

Name: ______________________ Title: ______________________

Company: ______________________ Phone: ( ) ______________________

Address: ____

City: ______________________ State: ______ Zip Code: __________

1. Please rate this manual for the following:

	Poor	Fair	Good	Excellent
Clarity	□	□	□	□
Completeness	□	□	□	□
Organization	□	□	□	□
Technical Content/Accuracy	□	□	□	□
Readability	□	□	□	□

Please comment: ____

____

2. Does this manual contain enough examples and figures?
Yes □ No □

Please comment: ____

____

3. Is any information missing from this manual?
Yes □ No □

Please comment: ____

____

4. Is this manual adequate for your purposes?
Yes □ No □

Please comment on how this manual can be improved: ____

____

____

____

____

____

____

Fold Down First

NO POSTAGE
NECESSARY
IF MAILED
IN THE
UNITED STATI

BUSINESS REPLY MAIL

First-Class Mail Permit No. 7628 San Jose, California 95131

Postage will be paid by addressee

An NG Company

Integrated Solutions

ATTN: Technical Publications Manager
1140 Ringwood Court
San Jose, CA 95131

Fold Up Second

Staple Here