Radio Shack®

TRS-XENIX™ Operations Guide

CUSTOM MANUFACTURED IN U.S.A. BY RADIO SHACK, A DIVISION OF TANDY CORPORATION

TERMS AND CONDITIONS OF SALE AND LICENSE OF RADIO SHACK COMPUTER EQUIPMENT AND SOFTWARE PURCHASED FROM A RADIO SHACK COMPANY-OWNED COMPUTER CENTER, RETAIL STORE OR FROM A RADIO SHACK FRANCHISEE OR DEALER AT ITS AUTHORIZED LOCATION

LIMITED WARRANTY

I. CUSTOMER OBLIGATIONS

A. CUSTOMER assumes full responsibility that this Radio Shack computer hardware purchased (the "Equipment"), and any copies of Radio Shack software included with the Equipment or licensed separately (the "Software") meets the specifications, capacity, capabilities, versatility, and other requirements of CUSTOMER

B CUSTOMER assumes full responsibility for the condition and effectiveness of the operating environment in which the Equipment and Software are to function, and for its installation

II. RADIO SHACK LIMITED WARRANTIES AND CONDITIONS OF SALE

A For a period of ninety (90) calendar days from the date of the Radio Shack sales document received upon purchase of the Equipment, RADIO SHACK warrants to the original CUSTOMER that the Equipment and the medium upon which the Software is stored is free from manufacturing defects THIS WARRANTY IS ONLY APPLICABLE TO PURCHASES OF RADIO SHACK EQUIPMENT BY THE ORIGINAL CUSTOMER FROM RADIO SHACK COMPANY-OWNED COMPUTER CENTERS, RETAIL STORES AND FROM RADIO SHACK FRANCHISEES AND DEALERS AT ITS AUTHORIZED LOCATION. The warranty is void if the Equipment's case or cabinet has been opened, or if the Equipment or Software has been subjected to improper or abnormal use If a manufacturing defect is discovered during the stated warranty period, the defective Equipment must be returned to a Radio Shack Computer Center, a Radio Shack retail store, participating Radio Shack franchisee or Radio Shack dealer for repair, along with a copy of the sales document or lease agreement The original CUSTOMER'S sole and exclusive remedy in the event of a defect is limited to the correction of the defect by repair, replacement, or refund of the purchase price, at RADIO SHACK'S election and sole expense RADIO SHACK has no obligation to replace or repair expendable items.

B RADIO SHACK makes no warranty as to the design, capability, capacity, or suitability for use of the Software, except as provided in this paragraph Software is licensed on an "AS IS" basis, without warranty. The original CUSTOMER'S exclusive remedy, in the event of a Software manufacturing defect, is its repair or replacement within thirty (30) calendar days of the date of the Radio Shack sales document received upon license of the Software. The defective Software shall be returned to a Radio Shack Computer Center, a Radio Shack retail store, participating Radio Shack franchisee or Radio Shack dealer along with the sales document

C. Except as provided herein no employee, agent, franchisee, dealer or other person is authorized to give any warranties of any nature on behalf of RADIO SHACK.

D Except as provided herein, **RADIO SHACK MAKES NO WARRANTIES, INCLUDING WARRANTIES OF MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE.**

E Some states do not allow limitations on how long an implied warranty lasts, so the above limitation(s) may not apply to CUSTOMER.

III. LIMITATION OF LIABILITY

A. EXCEPT AS PROVIDED HEREIN, RADIO SHACK SHALL HAVE NO LIABILITY OR RESPONSIBILITY TO CUSTOMER OR ANY OTHER PERSON OR ENTITY WITH RESPECT TO ANY LIABILITY, LOSS OR DAMAGE CAUSED OR ALLEGED TO BE CAUSED DIRECTLY OR INDIRECTLY BY "EQUIPMENT" OR "SOFTWARE" SOLD, LEASED, LICENSED OR FURNISHED BY RADIO SHACK, INCLUDING, BUT NOT LIMITED TO, ANY INTERRUPTION OF SERVICE, LOSS OF BUSINESS OR ANTICIPATORY PROFITS OR CONSEQUENTIAL DAMAGES RESULTING FROM THE USE OR OPERATION OF THE "EQUIPMENT" OR "SOFTWARE". IN NO EVENT SHALL RADIO SHACK BE LIABLE FOR LOSS OF PROFITS, OR ANY INDIRECT, SPECIAL, OR CONSEQUENTIAL DAMAGES ARISING OUT OF ANY BREACH OF THIS WARRANTY OR IN ANY MANNER ARISING OUT OF OR CONNECTED WITH THE SALE, LEASE, LICENSE, USE OR ANTICIPATED USE OF THE "EQUIPMENT" OR "SOFTWARE".

NOTWITHSTANDING THE ABOVE LIMITATIONS AND WARRANTIES, RADIO SHACK'S LIABILITY HEREUNDER FOR DAMAGES INCURRED BY CUSTOMER OR OTHERS SHALL NOT EXCEED THE AMOUNT PAID BY CUSTOMER FOR THE PARTICULAR "EQUIPMENT" OR "SOFTWARE" INVOLVED.

B RADIO SHACK shall not be liable for any damages caused by delay in delivering or furnishing Equipment and/or Software.

C No action arising out of any claimed breach of this Warranty or transactions under this Warranty may be brought more than two (2) years after the cause of action has accrued or more than four (4) years after the date of the Radio Shack sales document for the Equipment or Software, whichever first occurs.

D Some states do not allow the limitation or exclusion of incidental or consequential damages, so the above limitation(s) or exclusion(s) may not apply to CUSTOMER.

IV. RADIO SHACK SOFTWARE LICENSE

RADIO SHACK grants to CUSTOMER a non-exclusive, paid-up license to use the RADIO SHACK Software on **one** computer, subject to the following provisions:

A. Except as otherwise provided in this Software License, applicable copyright laws shall apply to the Software.

B. Title to the medium on which the Software is recorded (cassette and/or diskette) or stored (ROM) is transferred to CUSTOMER, but not title to the Software.

C CUSTOMER may use Software on one host computer and access that Software through one or more terminals if the Software permits this function.

D CUSTOMER shall not use, make, manufacture, or reproduce copies of Software except for use on **one** computer and as is specifically provided in this Software License. Customer is expressly prohibited from disassembling the Software.

E. CUSTOMER is permitted to make additional copies of the Software **only** for backup or archival purposes or if additional copies are required in the operation of **one** computer with the Software, but only to the extent the Software allows a backup copy to be made. However, for TRSDOS Software, CUSTOMER is permitted to make a limited number of additional copies for CUSTOMER'S own use.

F CUSTOMER may resell or distribute unmodified copies of the Software provided CUSTOMER has purchased one copy of the Software for each one sold or distributed. The provisions of this Software License shall also be applicable to third parties receiving copies of the Software from CUSTOMER.

G. All copyright notices shall be retained on all copies of the Software.

V. APPLICABILITY OF WARRANTY

A The terms and conditions of this Warranty are applicable as between RADIO SHACK and CUSTOMER to either a sale of the Equipment and/or Software License to CUSTOMER or to a transaction whereby RADIO SHACK sells or conveys such Equipment to a third party for lease to CUSTOMER.

B. The limitations of liability and Warranty provisions herein shall inure to the benefit of RADIO SHACK, the author, owner and/or licensor of the Software and any manufacturer of the Equipment sold by RADIO SHACK

VI. STATE LAW RIGHTS

The warranties granted herein give the **original** CUSTOMER specific legal rights, and the **original** CUSTOMER may have other rights which vary from state to state.

TRS-XENIX OPERATIONS GUIDE

IMPORTANT

TRSDOS™ Application Users: If you are currently running your Radio Shack Application under TRSDOS 2.0a (or 2.0b) and you plan to upgrade to TRS-XENIX™ Applications, you must transfer your data files to TRSDOS-II on your hard disk before upgrading to TRS-XENIX. If you have not initialized your hard disk with TRSDOS-II, follow the instructions in your Hard Disk Owner's Manual.

If you have been using TRSDOS-II on your hard disks prior to receiving your TRS-XENIX Runtime System, please note the following:

- TRS-XENIX overwrites any information already stored on your hard disk. Carefully follow the instructions given here to save important data and programs for later use.
- Be sure to SAVE your TRSDOS-II programs and data to floppy disks before installing TRS-XENIX.
- Only TRSDOS-II 4.0 (or later) files can be transferred to TRS-XENIX. If you have any TRSDOS 2.0a (or 2.0b) files that you want to transfer to TRS-XENIX, you must first convert them to TRSDOS-II using the FCOPY command.
- You can still run your TRSDOS and TRSDOS-II programs from floppy disk.

The following explains how to:

1. FCOPY any TRSDOS 2.0a (or 2.0b) programs and data you wish to save over to your hard disk.
2. SAVE all files on your hard disk to floppy disks.

Note that by FCOPYing your TRSDOS programs to hard disk first, all your files are located on set of floppy disks after the SAVE process.

FCOPY Your TRSDOS Programs to TRSDOS-II

The FCOPY command copies files from floppy disk TRSDOS 1.2, 1.2a, 2.0, 2.0a, or 2.0b to a disk formatted by TRSDOS-II (and vice versa except to TRSDOS 1.2 or 1.2a).

(For your information on FCOPY, see your TRSDOS-II Reference Manual.)

1. Power up your hard disk system under TRSDOS-II Version 4.1 (or later).
2. Insert your TRSDOS application disk in Drive 0.
3. Enter the FCOPY command at TRSDOS-II Ready.

   FCOPY 0 TO 4 {SYS,ALL} (ENTER)

   This copies all files, system and user, that are on the TRSDOS application disk in Drive 0 to Drive 4.

Alternate Method: FCOPYing A File. If you know the filespecs of the programs to be converted, you may want to use a different syntax form:

FCOPY filespec:drive TO drive

This is especially useful when you convert only a few programs. For example:

FCOPY SAMPLE/DAT:0 TO 4

copies the file SAMPLE/DAT from the disk in Drive 0 to Drive 4.

FCOPY Your Data Disks to TRSDOS-II

1. Insert the TRSDOS data disk (source disk) into Drive 0.
2. At TRSDOS-II Ready, type:

   FCOPY 0 TO 4 {SYS,ALL} ENTER
3. TRSDOS-II stores your data on Drive 4. When it finishes the conversion, TRSDOS-II displays the message ** FCOPY Complete **.

SAVE Your Hard Disk Files to Floppy Disks

You are now ready to save all of the programs and data stored on your hard disk.

> **Important:** Since this is a very important step in preparing to install TRS-XENIX on your hard disk, we strongly suggest making at least two SAVE Data sets.

Be sure to have a supply of blank disks ready. Do not format them; SAVE organizes the disks into its own special format.

Remember, these diskettes will be your archive file of all of your programs that were stored on your hard disk under TRSDOS-II. Take the time to put the date, volume and data set numbers on each diskette. You may also want to put descriptions on each diskette.

To save all files from your hard disk (programs, data, and system files), enter the following command at TRSDOS-II Ready:

SAVE :4 :0 {SYS,ALL,ABS} ENTER

If you have secondary hard disks, enter the following command for each:

SAVE : d :0 {ALL,ABS} ENTER

where d is the hard disk from which the files are being SAVEd. The files are stored on disks in Drive 0. As one disk becomes full, TRSDOS-II prompts you for the next disk.

When all of the files are saved, TRSDOS-II prompts you to insert Volume 0 of your Save Data Set into Drive 0. TRSDOS-II now updates that disk with housekeeping information.

When the save is finished, you are returned to TRSDOS-II Ready. Repeat the procedure again to create a Backup Save Data Set.

Radio Shack®
A DIVISION OF TANDY CORPORATION
FORT WORTH, TEXAS 76102

Notes to Users of TRS-XENIX

- This manual does not attempt to discuss every command and option included on your TRS-XENIX Runtime System Diskette. Only those commands that are necessary to effectively run applications under TRS-XENIX are described. For more information on TRS-XENIX commands, ask your Radio Shack dealer about the TRS-XENIX Development System.

- The RUNCOBOL program is included with TRS-XENIX Runtime so you can run your TRS-XENIX application programs.

- When running TRS-XENIX, the hard disks are numbered Ø-3, where Drive Ø is the primary hard disk.

- **Do not** turn on the write-protect switch on your hard disks or remove the write-enable tab from your floppy disks when running TRS-XENIX. Approximately every 3Ø seconds TRS-XENIX will access the drives to update its files and directories.

- TRS-XENIX uses the "Media Error Map" on the bottom of your hard disk to format the disk. If the map for your hard disk is missing, contact your Radio Shack dealer for assistance.

- The **mail** program is included in the TRS-XENIX Development System (26-64Ø1). Commands mentioned in this guide which use **mail** will operate normally, although the mail operation will not be invoked.

- TRS-XENIX Applications user's should look at Appendix E for instructions on installing their applications. Making backups of both the programs and the data are also described.

Table of Contents

Chapter page

Chapter page

CHAPTER 1

INTRODUCTION

This Operations Guide is designed to help you learn the basics of your new TRS-XENIX system, and to help you get the system up and running on your computer.

By the time you've read through the Guide and used your computer to duplicate the examples given in each chapter, you'll have a good, overall "hands-on" feeling for the way TRS-XENIX operates, and you'll be ready to use whichever of the TRS-XENIX applications packages is appropriate for your own day-to-day work. You'll also have learned in some detail about the important role of the TRS-XENIX System Manager.

Once you're familiar with the basics, you may want to learn more about TRS-XENIX. In that case, the TRS-XENIX Development System can give you more precise information about every aspect of the system, including programming instructions and a full list of TRS-XENIX commands, with their technical specifications.

Meanwhile, this Guide will get you started on your way to full and productive use of your TRS-XENIX system.

1.1 The TRS-XENIX System

TRS-XENIX is an operating system -- a master collection of programs, resident in the computer at all times, designed to control the computer's resources. An operating system, for instance:

- enables the central processor, input/output devices (like line printers and terminals), and storage devices (like hard and floppy disk drives) to communicate among themselves.
- controls user access to the system's files.
- determines what share of processing time each program will get.
- helps to ensure that processing time and disk space are fairly distributed among users.
- provides a way for user programs -- the software -- to communicate with the computer's hardware.

Most of the time, an operating system is invisible to the computer user, since it works internally and automatically to make certain the computer does what it should do. But, occasionally, the user does interact with the operating system -- for instance, when the system prompts the user by asking questions, or when it's time to use a command. That's why some basic knowledge about whatever operating system you're using is important.

TRS-XENIX is a thorough, flexible, and efficient system -- but what makes it especially attractive is its multi-tasking capability: TRS-XENIX is a system able to handle simultaneously the different demands of several users. For instance, if you have terminals connected to your computer, TRS-XENIX can perform separate and simultaneous tasks (or processes) for each terminal's user. If you have a single-terminal system, TRS-XENIX can still run several processes simultaneously for you. That's a recent breakthrough for micro-computers, and a real convenience for microcomputer users.

But, although in one way it's very new, TRS-XENIX also has the advantages of a system with a track record, because TRS-XENIX is derived from UNIX, a multi-tasking system developed by Bell Labs a number of years ago. UNIX is widely used on large computer systems, and its processes and programs -- and therefore TRS-XENIX's -- have time and again proved themselves workable and valuable.

1.2 The Role of the System Manager

If you're a veteran computer user, your operation may already have a System Manager -- one person with the overall responsibility for the smooth running of the computer system. If you don't have a System Manager, your first priority should be to designate someone, either yourself or another reliable person with administrative capabilities. (In a large operation, the System Manager's work can be full-time, and he or she may even require some assistance: in a smaller system, with only a few users, the work is less time-consuming, but no less critical.)

The System Manager's major responsibilities -- all of which are described in some detail in later chapters -- include:

- the initial installation of the TRS-XENIX system.
- adding and removing user accounts, passwords and file systems.

  ensuring that system resources are efficiently distributed.
- "backing up", or copying, all files on the system, to guard against the accidental loss of programs and data.

As System Manager, you may also be responsible for giving training and support to new users, and for dealing with hardware and software suppliers. You'll also set up and administer a library of floppy disks and other storage devices containing system backups, user files, and applications programs.

It's not an easy job. For instance, when your system is short of space, it can be frustrating to persuade users to take the time to identify and delete obsolete files from overcrowded disks -- nobody likes to do computer "housework".

But if you do the job well, you'll have what every System Manager wants: an efficient, productive system with efficient, productive -- and satisfied -- users.

> **HINT:** The System Manager logs in as the "root" of the TRS-XENIX system. The root is the most powerful user in TRS-XENIX and should only be used when absolutely necessary.

CHAPTER 2

BEFORE YOU START

Before you begin to install your TRS-XENIX system, reading this chapter will familiarize you with some basics that will help make both installation and operation easier for you. Included here is information about:

- how to use the computer's keyboard with TRS-XENIX.
- how TRS-XENIX files and directories are identified.
- how to use some of TRS-XENIX's commands.
- how to start and stop your system safely (including what to do if the system is accidentally halted).

2.1 Your Keyboard

In most respects, the keyboard of your terminal is exactly like that of a typewriter. Most of the letters, numerals, and punctuation marks are in the same place, and you will quickly discover that the "spacebar", "repeat", and "shift" keys behave much the way they do on any typewriter.

However, there are a few differences you should be aware of before you begin working. For instance, on a computer, the backspace key has a double function. In addition to backing up, one character at a time, it lets you correct typing errors, If you type s, for example, but meant to type d, just backspace and type the d over the s.

Another difference: on a computer, you may not interchangeably use keys that look alike -- the upper-case letter "OH" (O) and zero (Ø), for example, or the lower-case "el" (l) and the number one (1). All computers will recognize these as separate characters, so be sure you always type the correct one.

Some of the keys on your keyboard have a special meaning when you are using TRS-XENIX. These include control keys or sequences used to produce special TRS-XENIX characters that don't appear on your keyboard, and various "escape" sequences, used to exit from programs, terminate activities, log out, or stop the movement of text on your screen.

Angle brackets (< >) are used in this Guide to represent keys. Note that, whenever you are asked to type the "control" key along with another key, you should always press the <CTRL> key first, and hold it down while you type the second key, just as you would hold down the shift key on a typewriter while typing the letter you want to capitalize.

Here are a few of the most commonly-used TRS-XENIX characters and sequences:

<CTRL><S> Typing the "control" and "s" keys simultaneously will stop text from "scrolling", that is, moving up and off the screen. To start scrolling again, type <CTRL><Q>.

<CTRL><D> You will have several important uses for the <CTRL><D> sequence, all explained in detail later in this guide:

- to log in and out
- to bring the system up from maintenance mode
- to exit from writing a **wall** message

(Remember: for any "control" key sequence, hold the <CTRL> key down while typing the other character.)

<BREAK> In addition to special uses it may have in specific TRS-XENIX programs, the <BREAK> key will interrupt any command you have entered and return you immediately to the system "prompt", the sign that TRS-XENIX is ready to accept a command.

<ENTER>	In some documentation you may also see this key named <RETURN>, for "carriage return". You must type the <ENTER> or <RETURN> key after every command in order for TRS-XENIX to receive your instructions.
backslash (\)	The backslash (\) character is used in some advanced features of TRS-XENIX. It can be entered from the console by typing <CTRL></>.
pipe (|)	The pipe is also used in some advanced features of TRS-XENIX. It can be entered by typing <CTRL><1> from the console.

2.2 TRS-XENIX File and Directory Naming Conventions

The TRS-XENIX system comes with a number of files and directories already set up in the system itself; you may want to add others as you work with TRS-XENIX (Chapter 5 tells you how). But whether you're using built-in files and directories or creating your own, you should note TRS-XENIX's naming conventions.

All TRS-XENIX files and directories have names up to fourteen characters long, and may include any combination of upper and lower case letters and numbers. When naming files and directories, you shouldn't use the slash (/) character or any punctuation marks except the period (.). Such characters often have special meanings for TRS-XENIX, and, by using them, you could accidentally trigger commands and sequences you don't want.

TRS-XENIX recognizes two "wild card" characters, the question mark (?) and the asterisk (*); they're timesavers, conveniences you can use to replace filenames or parts of names. The question mark replaces a single character, and the asterisk replaces several characters, or even an entire name.

For example, you might refer to a file called chap2.s in any of these ways:

```
chap?.s        *2.s          *2*
```

If you want a command to apply to several files, you can use the "wild card" capability to process them all simultaneously. For instance, if you want to refer to all your files called "chapters" (chap1.s, chap2.s etc.), you can simply type:

```
*.s
```

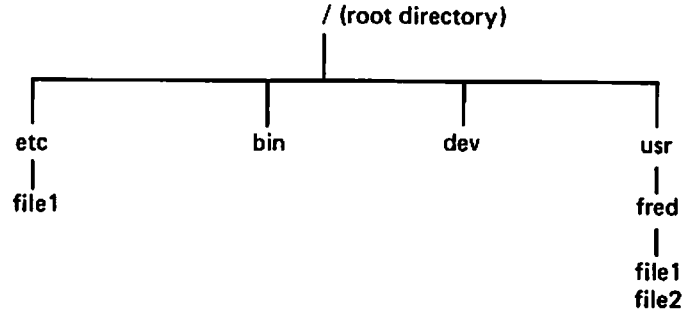
to tell TRS-XENIX that you mean "all the files ending in .s". You could even use:

```
*
```

alone to mean "every file". A cautionary note: be careful when you're using the "wild card" characters. If you're using a command like "remove files", mistaken use of the ? or the * could result in accidental removal of files you wanted to save.

2.3 The TRS-XENIX File Structure

TRS-XENIX files are grouped in directories and arranged hierarchically. That is, a directory, which contains a collection of files, may be a member of yet another directory, and so forth. This results in the formation of a tree with branches:



The "root" of this tree is the root directory, by convention called "slash" (/). TRS-XENIX identifies files by their "pathnames", the path you take along the branches of this tree to arrive at the named file. The steps along the way are denoted and separated by slashes (/). In the diagram, for example, there are two files called file1. One of these is, in fact, /etc/file1. The other file is /usr/fred/file1. The TRS-XENIX file structure is discussed at much greater length in Chapter 5. For now, however, it is sufficient that you understand the "pathname" concept when you work with directories and files.

2.4 How to Use TRS-XENIX Commands

A TRS-XENIX command may be typed whenever the prompt appears, indicating that you are in the "shell", the command level of the system. This prompt will be either a dollar sign ($) for users or a number sign (#) for root. TRS-XENIX commands generally consist of single lower-case words, which may be followed by one or more "switches" and "arguments". Most often the argument will be a filename. Every command must be followed by typing <ENTER>.

To take a simple example, you can use the command **cat** (catenate) to look at the contents of a file. If, while the prompt is on the screen, you type beside it:

```
cat file1  <ENTER>
```

the contents of file1 will appear on your screen.

To use another example, the command **lc** lists the contents of whatever directory you name. Therefore:

```
lc /usr/fred  <ENTER>
```

will give you a list of files and directories in the directory /usr/fred. Remember, when you type TRS-XENIX commands, be sure to include or omit spaces exactly as indicated.

2.5 Stopping the System Safely

You'll want to read and carefully follow the directions given in Chapter 3 for correct TRS-XENIX shutdown procedures. Meanwhile, though, there are some facts to keep in mind:

- While TRS-XENIX is powered up, you should not ordinarily press the reset or power switches, turn the hard disk key, or unplug your computer.

  If you should inadvertently stop your system, or if you experience a power failure, then the next time you start your system up, TRS-XENIX will remind you that its last shutdown was not properly done. It will ask you if you want it to "clean" the file system. Always answer yes (by typing <y>) to this question. TRS-XENIX will then proceed with an automatic program to locate and repair any system inconsistencies caused by the previous shutdown. (The program, called **fsck**, is explained in detail in Chapter 6.)

- At any time when you're using TRS-XENIX, if you make a mistake or simply want to start over, you shouldn't shut down the system. In most cases, however, you can press the <BREAK> key to return to the command level of TRS-XENIX. The prompt ($ or #) will reappear and you can begin again.

CHAPTER 3

STARTING UP

To get TRS-XENIX up and running on your system, you transfer TRS-XENIX system from the floppy disks you've received onto your own hard disk. This chapter tells you how to make that transfer. Basically, it involves three steps:

1. Formatting the hard disk by running the **diskutil** program from the TRS-XENIX Boot Disk.

2. Copying the contents of the same TRS-XENIX floppy onto the hard disk by using **hdinit**.

3. Copying the rest of TRS-XENIX from the other floppy disks in the package onto the hard disk by using **firsttime**.

Two notes to remember: first, ordinarily you have to do the transfer procedure only once, but it's wise to save the original TRS-XENIX floppy disks in case of any future mishap involving the hard disk. And second, if you make a mistake at any time during the installation process, just press the reset switch and repeat the procedure from the beginning.

3.1 Step One: The diskutil Program

You begin the TRS-XENIX installation by "booting", or starting, the system with the TRS-XENIX Boot Disk in Floppy Drive Ø and running a preliminary program called **diskutil**. **Diskutil** runs before TRS-XENIX is actually booted, and lets you accomplish two essential pre-TRS-XENIX tasks-- formatting your hard disk, and entering your "media error map", a necessary record of your hard disk's magnetic surface characteristics. (**Diskutil** has more uses, too; they're described in Chapter 7.)

Before you begin, make sure your hard disk is turned off, then remove your hard disk's media error map from the plastic envelope on the bottom of the disk drive.

The following instructions explain how to run **diskutil**.

1) Power up your computer and your hard disk as instructed in your owner's manuals. Make sure that all attached printers are turned on. Do **not** write-protect your hard disk.

2) Simultaneously press <REPEAT> and <BREAK> until the "Insert Diskette" message is displayed.

3) Insert the TRS-XENIX **Boot Disk** into floppy Drive Ø. As TRS-XENIX starts up, the copyright notice is displayed, and then you see the prompt:

```
XENIX Boot
:
```

4) Beside the colon, type:

```
diskutil <ENTER>
```

5) You see this message:

```
Diskutil: format or copy hard or floppy disks.
```

6) Then you're asked a series of questions:

```
Hard or floppy disk (h or f)?
```

Answer <h> for hard disk and then press <ENTER>.

```
Copy or format (c or f)?
```

Answer <f> to format the hard disk and then press <ENTER>.

```
Hard disk unit number (Ø...3)?
```

Answer <Ø> to indicate the first hard disk and then press <ENTER>.

7) To answer the next two questions, you need information from your Hard Disk Owner's Manual:

How many cylinders?

Answer with the number given in your Hard Disk Owner's Manual -- 256 for an 8-megabyte disk or 23Ø for a 12-megabyte disk, for instance. After entering the correct number, press <ENTER>.

How many heads?

Again, answer with the number given in your Hard Disk Owner's Manual -- 4 for an 8-meg disk, or 6 for a 12-meg, for instance. Then press <ENTER>.

8) Now you are prompted to enter the information from the TRACK and HEAD columns of your media error map. **Diskutil** repeatedly prompts you with:

enter next pair or "done":

If for instance, your list looks like this:

TRACK	HEAD	BYTE COUNT	LENGTH
133	ØØ	Ø1333	Ø2
174	Ø1	Ø9826	Ø5

then your completed entry would look like this:

enter next pair or "done": **133,Ø** <ENTER>
enter next pair or "done": **174,1** <ENTER>
enter next pair or "done": **done** <ENTER>

with you having typed in the numbers and pressed <ENTER>. If your list is blank, simply type:

done <ENTER>

9) You're then told that the hard disk is about to be formatted; you're also told the approximate length of time the formatting takes -- usually at least 15 to 30 minutes. The message looks like this:

```
About to format hard disk drive 0.
This will take about xx minutes.

Type <ENTER> to proceed or <BREAK> to abort:
```

You should press <ENTER> to proceed; if you made a mistake and want to start over, press <BREAK>.

10) When the formatting is complete, you see this:

```
Hard disk successfully formatted.
Drive parameters and MEDIA ERROR MAP successfully
written.  Your hard disk is ready for the TRS-XENIX
initialization.
```

11) Now, leaving both the boot floppy and the hard disk as they are, you're ready to proceed to Step Two.

3.2 Step Two: The hdinit Program

Hdinit accomplishes four basic tasks, three of which involve transferring material from the boot floppy to your hard disk:

A. It transfers the "boot track" -- the initial code that lets you call up TRS-XENIX on your computer from now on.

B. It transfers material which creates a TRS-XENIX file system on your hard disk.

C. It transfers the rest of the contents of the boot floppy.

D. It shuts itself down.

The following instructions explain how to run the **hdinit** program.

12) Press the reset button, and then press the <BREAK> key repeatedly until the boot prompt appears:

```
XENIX Boot
:
```

Press <ENTER> and the system responds as if you had typed the word "xenix" after the colon. You are then given the opportunity to change the root disk if you wish. However, you should simply press <ENTER> to proceed.

13) The **hdinit** program starts up automatically and displays a message telling you that you are running TRS-XENIX from the "Installation Floppy". The program asks:

Do you wish to initialize your hard disk?

Answer <y> for yes and press <ENTER>.

Has your hard disk been formatted with diskutil?

Naturally, if you have completed the formatting procedure described earlier, you answer <y> and press <ENTER>.

14) Once again, you are asked for the number of cylinders and heads on your hard disk. Respond with the same numbers you indicated during the formatting of the hard disk with diskutil (e.g. 256 cylinders and 4 heads for an 8-megabyte hard disk, or 23Ø cylinders and 6 heads for a 12-megabyte hard disk).

15) The system proceeds with the four steps outlined earlier -- installing the boot track, making a file system, copying TRS-XENIX files from the floppy disk to the hard disk, and finally shutting the system down. Don't touch the system until you see the message:

```
Halting System...
** Normal System Shutdown **
```

16) You're now ready to proceed to the next -- and final -- step in the installation process.

Note: If for some reason you are repeating this procedure, you may be warned, while the file system is being created, that "mkfs: /dev/hdØ contains data". You are asked whether to "overwrite". Answer <y> to finish installing the system.

3.3 Step Three: The firsttime Program

In this final phase of installing TRS-XENIX, you will for the first time boot -- start up -- from the hard disk (the last two boots have been from the floppy). Then **firsttime** copies the contents of the remaining floppies onto the hard disk. Here are instructions for running **firsttime**:

17) Press the reset button to boot TRS-XENIX from the hard disk. This time, do not press <BREAK>.

18) When you see:

```
XENIX Boot
:
```

press <ENTER> and the **firsttime** program starts up automatically.

19) Instructions for inserting and removing your floppy disks are displayed. You are prompted with the question:

```
First Floppy? [y,n]
```

Insert the **INSTALL 1** floppy disk into Drive Ø; press <y> and then <ENTER>.

2Ø) The system tells you that it is:

```
Extracting files from floppy. . .
```

and lists the files, byte length and number of blocks. When this is complete, you're prompted with:

```
Next floppy [y,n]
```

21) Insert the **INSTALL 2** floppy disk in Drive Ø and answer <y> and <ENTER>. The system continues to copy files to the hard disk.

22) Since you have only two floppy disks to "install", answer <n> for "no" when it asks for the next floppy. The following message is displayed:

```
Setting up directories and permissions
Installation complete.
```

Your hard disk now contains a complete TRS-XENIX operating system. The next message displayed is:

```
Type CONTROL-d to proceed with normal startup
(or give root password for system maintenance):
```

This is the normal boot you will see every time you start your TRS-XENIX system. (It's described in the next section.)

> **A Reminder:** the next time you power down your hard disk, return the media error map to its place on the disk drive.

3.4 Booting Your TRS-XENIX System

After your installation is complete, it's a simple matter to boot your TRS-XENIX system whenever you want. Of course, if your system isn't already on, turn on both the hard disk and the computer. Once you've powered up, TRS-XENIX starts up. The screen reads:

```
XENIX Boot
:
```

Then press <ENTER> and TRS-XENIX responds with "xenix".

Remember (as described in Chapter 2) that if your last system shutdown was not done correctly, TRS-XENIX at this point asks you if you want it to "clean" the file system; always answer yes (by typing <y>). When TRS-XENIX has cleaned the files, the boot procedure usually continues normally. Occasionally, however, you are asked to reboot; in that case, press the reset switch to begin again.

After TRS-XENIX has booted, the screen automatically displays information about availability of your system's memory space and other characteristics. That display is followed by the instructions:

```
Type CONTROL d to proceed with normal startup
(or give root password for system maintenance):
```

The vast majority of the time, you'll respond by typing <CTRL><D> to make the full TRS-XENIX system available to you and other users. If you should occasionally want to do some system maintenance work (see Chapter 6), respond with the password for the root (see Chapter 4), which puts you in "single-user" mode. Single-user mode should be used for maintenance duties, such as backups, restores, dumps and file checking.

After you type <CTRL><D>, you are asked for the time and date. (Note that **yy** means year, **mm** means month, **dd** means day, **hh** means hour, **mm** means minutes, and **ss** means seconds.) Although the system accepts an <ENTER>, it is recommended that you respond with the complete time/date information. This information is stored by TRS-XENIX and ultimately helps you with record-keeping, because you are able to check, for instance, on when files were created or last changed. If you press <ENTER> without entering a date, TRS-XENIX uses the last date as a default.

3.5 Login

Logging in -- login -- is the next step after booting. It's the procedure in which you identify yourself to TRS-XENIX, and get the system ready for your commands.

After you've entered time/date, you see the single word:

```
login:
```

If this is the first time you've used TRS-XENIX, and you're the System Manager, type:

```
root <ENTER>
```

and proceed with creating the "root" password and setting up user accounts. ("Root", passwords, and user accounts are all discussed in Chapter 4.)

If the system is up and running, complete with users, and this is a routine login, the user, even the system manager, should enter his login name, followed by his own password, (See Chapter 4 for information about login names and passwords.)

Once login is complete, the system responds with a welcoming message, and then you see the TRS-XENIX prompt-- the "go-ahead" sign:

- If you are logged in as "root", the prompt is a number sign -- (#).
- If you are logged in as a regular user, the prompt is a dollar sign -- ($).

When a prompt appears, you're ready to begin using TRS-XENIX.

3.6 Logging Out

When you've finished your work on TRS-XENIX and you want to log out and get the system ready for the next user, simply type:

<CTRL><D>

and TRS-XENIX responds with an invitation for the next user to log in.

3.7 Halting the System

Shutting down TRS-XENIX, either at the end of a work day, or for any other reason, is a straightforward procedure.

If you've logged in as "root", when you see the prompt #, type beside it:

/etc/haltsys <ENTER>

and you see the message:

** Normal System Shutdown **

If you're in a multi-user environment, you should warn other users before shutting down the system, so that they can close their files and log off the system. (This is **very important** for applications users.) An easy way to do this is to use the **shutdown** program to stop TRS-XENIX. At the root prompt, type:

shutdown <ENTER>

You're asked how many minutes there are until the shutdown. TRS-XENIX then sends a warning message to all terminals, and proceeds with the shutdown at the appropriate time.

> **Note:** TRS-XENIX applications users, see Appendix D for instructions on installing your applications.

CHAPTER 4

GETTING THE SYSTEM READY FOR YOUR USERS

One of TRS-XENIX's special features is its capacity to support many users, simultaneously on several terminals, or sequentially on one terminal. No matter how many people use TRS-XENIX on your system -- even if it's only one or two -- each user must have his or her own login name, password, and an assigned intra-computer "workspace".

As System Manager, you're responsible for adding users to the system, and this chapter tells you how to do so. It also tells you:

- more about "root" and the "root" login mentioned in Chapter 3.
- about removing a user account from the system.
- about a file called .profile.
- about using the system to communicate with other users.

4.1 "Root" -- the Super-User

To guard against mishaps and to protect confidentiality, TRS-XENIX automatically restricts access to many system files. It also provides a protection mechanism to allow users to restrict access to their own files (see Chapter 5).

There is one user, however, who has unlimited access to the system and everything on it. This person -- who is, in effect, a "super-user"-- automatically has the login name "root".

Root is unique -- for instance, only root can read, and change, anything on the system, whether or not the material is protected against other users; only root can perform certain vital functions, like adding or removing users; and only root can carry out many system maintenance duties.

Since the power given to root is considerable, and since misuse of that power could seriously damage the whole TRS-XENIX system, it's necessary for you, as System Manager, to make some early decisions about root. You should quickly do three things:

1. Determine who will be root on your system. (As System Manager, you're the logical choice, although on larger systems it's sometimes appropriate for more than one person to be able to function as root.)

2. Choose, and enter, a password for root so that unauthorized people won't have root's unlimited access. (The next section of this chapter tells you how to choose and enter the root password. Of course, if more than one person is to function as root, they'll share the password.)

3. Set up a regular, non-root user entry for the system manager to use when performing regular TRS-XENIX work that doesn't require root's special powers. With root's unlimited access, even minor mistakes in the root mode can cause major problems, so the rule of thumb for root to follow is this: log in as root only when it's absolutely necessary. Otherwise, log in as a regular user.

4.2 The Root Password

First, choose the root password. It should contain 6 to 14 characters, and should not be something easily guessed -- a birthdate, nickname, or license number for instance. Mix upper- and lower-case characters and numbers, and, as you select the password, be certain it's something you'll be able to remember.

After you've carefully chosen your password and have logged in as root by typing:

```
root <ENTER>
```

type beside the prompt (#):

passwd root <ENTER>

TRS-XENIX responds: New password:
Retype new password:

For privacy's sake, the password is not displayed while you're typing it. After you've entered the password twice, the system automatically records the entry in TRS-XENIX's /etc/passwd file.

Note that pressing <ENTER> as the new password tells TRS-XENIX to not change the old password.

> **Important: Do not forget the root password. There is no way to recover from a lost root password; if you forget the root password, you'll have to reinstall the TRS-XENIX system from the distribution floppy disks.**

Remember that the number of individuals who are given the root password should be extremely limited. Persons given access to the root password should log in as root only when absolutely necessary.

Also remember that even if you're the only user on your system, you should still create a regular user entry for yourself, and use it when you're doing work that doesn't require root's wide powers.

4.3 Creating Users: The mkuser Program

In order for a new user to have access to TRS-XENIX, he must have an individual user entry, and setting up these new user's is one of the jobs of the System Manager.

You need to be familiar with three terms before you begin:

Login name is the name by which the user is known to the system. Most people find it convenient to use a shortened form of the user's real name -- for instance, the login name of a user named John Doe might be a first name and last initial (johnd), a first initial and last name (jdoe), or three initials (jad).

Comment is a space to record information about a new user. The format to use is: name, department, phone extension. This information and format are used in some advance features of TRS-XENIX, so be complete and accurate in your entries. You're limited to 20 characters, including spaces.

Initial password is the password you assign to a new user. User passwords, like root's, are generally 6 to 14 characters long, and work best when they're a random mix of upper- and lower-case letters, digits, and special characters. (Pressing <ENTER> sets the password to a null password -- no password.) Each user can later change his or her own password to make it more personal and memorable. This is described later.

> **Note:** Spaces and tabs and special characters entered within passwords and login names are considered integral parts of the passwords and names, and must be included at each login.

Before you begin to run **mkuser**, you're wise to have a list of the login names, comments, and initial passwords you'll be entering as you work. And note that **mkuser** sets up one account at a time; to set up multiple user accounts, you must repeat the procedures detailed here.

To run **mkuser**, you must be logged in as root. When you see the root prompt -- it's (#) -- type beside it:

```
mkuser <ENTER>
```

The system responds like this:

```
        Mkuser
        ------
    Add a user to the system
Do you require detailed instructions? (y/n/q)
```

You should respond by typing one of these three letters after the colon and then pressing <ENTER>:

- <y> provides instructions for **mkuser**
- <n> does not display instructions
- <q> exits the **mkuser** program and returns you to the prompt.

Note: You may press the <BREAK> key to terminate the **mkuser** program at any time, unless you are specifically instructed NOT to use <BREAK>.

Now the program asks you to enter the new user's login name. Type in the name after the colon and press <ENTER>, as in:

login name: **johnd <ENTER>**

Then **mkuser** asks you for a password entry. Please note that the password does not appear on your screen as you type -- you should type carefully because you won't be able to see if you've made a mistake.

As a double-check, **mkuser** asks you again for the password, and won't accept it if your two entries are not identical. Of course you have to remember the password so your user can later log in for the first time. A password might look like this:

BigGirl2319

After you've typed in the password and pressed <ENTER>, **mkuser** asks you for a comment entry, and indicates the 20-character space limitation. Type your comment under the dashes, like this:

Please enter Comment>----------
>John Doe,Acct,#333 <ENTER>

If you don't want to enter a comment, just press <ENTER> instead.

Mkuser now automatically continues setting up the account. When **mkuser** has completed this phase of its work, it shows you the new user's complete entry, which might look like this:

johnd:j9jfLzK8hfdo:204:50:John Doe,Acct,#333:/usr/johnd:/bin/sh

johnd is the login name you assigned to the user.

j9djfLzK8hfdo is the password you chose.

2Ø4	is the "user ID" (UID) number TRS-XENIX automatically created for the **johnd** account. UID numbers for regular TRS-XENIX users begin at 2ØØ; numbers below 2ØØ are automatically reserved for special IDs like "root".
5Ø	is the "group ID" number TRS-XENIX automatically assigns to new users. Group ID allows joint access to the same files; in effect, all TRS-XENIX users initially belong to the same group, number 5Ø.
John Doe,Acct,#333	is the comment you entered.
/usr/johnd	indicates the "workspace" assigned to **johnd**. This is also known as his "home directory". /usr is the main directory of users. The user can store his files in the subdirectory johnd.
/bin/sh	indicates that, when **johnd** uses TRS-XENIX, he'll be using the normal range of TRS-XENIX commands, which are located in the command (or shell) level of the system.

When you've reviewed the entry, **mkuser** gives you an opportunity to change the login name, password, or comment. If you want to make changes, follow TRS-XENIX's instructions. If, for any reason, you want to cancel the entire procedure at this point, press <BREAK>. If you don't do either of these, then **mkuser** gives you a report like this:

```
Password file updated
Home directory /usr/johnd created
/usr/johnd . profile created
User johnd added to this system.
```

You now have created a new user account, the full record of which is stored in TRS-XENIX's /etc/passwd file.

Now John Doe can log in to the system using the password you assigned to him. At this point he can, if he chooses, change his password -- he may want to choose something with personal meaning for him, or something which he finds particularly easy to memorize.

If he does want to change his password, you should alert him about what should and shouldn't be part of a good password. (You needn't ask John Doe what his new password is, since, if you are root, you have automatic access to the entire TRS-XENIX system and need no password but your own.)

> **Important:** Users should also be cautioned not to type their passwords before the password prompt appears, when logging in. TRS-XENIX is not completely initialized until the password prompt appears and your password might be displayed on the video if you do not wait for the prompt.

When John Doe has chosen a new password, he should type beside the prompt the command:

```
passwd  <ENTER>
```

TRS-XENIX responds:

```
Changing password for johnd
Old password:
New password:
Reenter for check:
```

When John Doe enters his old password and his new password twice, TRS-XENIX automatically changes his entry in the etc/passwd file.

As System Manager, you may occasionally need to alter entries in /etc/passwd. Chapters 5, 8, and 9 all contain important information to help you do so easily and correctly.

4.4 The .profile File

When any user --including root-- first logs in, TRS-XENIX file called .profile, built in to the user's home directory, is automatically executed. It's an internal file which, at subsequent logins, provides the system with information including:

- where the system should look for programs the user runs.
- where the user's electronic mail "box" is located.
- what certain characters the user types from the keyboard should mean to TRS-XENIX.

Ordinarily, the user needn't be concerned about his .profile file, since it's used only by TRS-XENIX. But there are times when it's necessary to edit .profile. In that case, follow the directions for multi-terminal systems (Appendix B) and for regular file editing (Appendix X).

4.5 Removing a User: The rmuser Program

As System Manager, you might occasionally want to delete a user from the system-- for instance, if John Doe leaves your company, naturally you won't want him to be able to continue to access your company's confidential files.

Deleting a user entry involves two steps: removing the user's files from the system (you don't have to be logged in as root to do this), and then using a program called **rmuser** to remove the user's login name and all its associated information. For **rmuser**, you must be logged in as root.

When you're dealing with John Doe's files, you're wise not to delete them without careful review; you might need them later. Look them over, transfer appropriate ones to other users' workspaces, and use floppy disks to save ones you want to store (all procedures described in Chapters 5 and 7). Then, when you want to delete the remaining material, simply type beside the prompt:

```
cd /usr/johnd <ENTER>
rm -fr *     <ENTER>
```

and all of John Doe's files are deleted.

The next step is to run the **rmuser** program, which can be used only by root. First, if **mail** is part of your system, make sure there's nothing in John Doe's electronic mailbox.

Then, beside the prompt, simply type:

rmuser <ENTER>

The system asks you for the name of the user you want to remove. Then it deletes the user's entry in the /etc/password file, and removes his or her mailbox, .profile file, and home directory.

Rmuser has several built-in safeguards-- for instance, it won't permit removal of root from the system, and it won't remove a user if there's mail waiting (if **mail** is part of your TRS-XENIX system). But it's still a good idea to use **rmuser** sparingly, since it's a powerful and permanent command.

4.6 Communicating with Your Users

There may be many times that you, as System Manager, will want to communicate with one or more of the users on your system, and when other users may want to communicate with you and among themselves. TRS-XENIX provides three major ways for such communication.

The Message of the Day: A built-in message-of-the-day file-- /etc/motd -- can be used as a system-wide "bulletin board", with daily reminders about topics like disk space shortages, meetings, scheduled maintenance, or anything else that's appropriate. Each time a user logs in, the message of the day is automatically displayed. Use **ed** to change the /etc/motd file (ed is described in Appendix X).

Write All: The command **wall** is an abbreviation for "write all". It's an effective emergency communications tool, which connects you immediately with all users on the system. By the same token, it's also disruptive, and should therefore be used only when it's necessary. To use wall, type the command **wall** beside the prompt. Then type your message, and end with <CTRL><D> (at the beginning of a line) instead of the usual <ENTER>. A sample **wall** message might look like this:

```
# wall  <ENTER>
Critical shortage of available diskspace.  <ENTER>
Please clean up files fast.  <ENTER>
-- from root  <ENTER>
<CTRL><D>
```

CHAPTER 5

THE TRS-XENIX FILE STRUCTURE

In order for you, as System Manager, and other TRS-XENIX users to get the greatest benefit from the system, you need to understand how TRS-XENIX uses directories and files to organize information.

This chapter:

- describes TRS-XENIX's "hierarchical" file and directory structure.
- familiarizes you with some basic TRS-XENIX commands which let you access and make changes in files and directories.
- discusses "permission settings" -- safeguards to protect files and directories, and giving privacy where you want it.

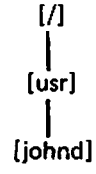
5.1 Files and Directories

TRS-XENIX stores information in disk "files", much as you might store a memo or other information in a file folder. (The disk files, however, in addition to containing ordinary information like memos, lists, lines of data, and the texts of documents, can also contain complete programs.)

As an aid to organization, TRS-XENIX also gives you the option of collecting groups of files in "directories", just as in an office you might organize files into categories, and group them together in labeled file cabinet drawers.

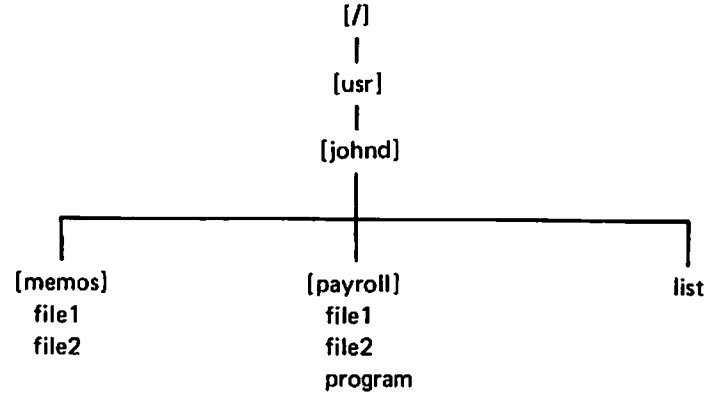
When you're working with TRS-XENIX files and directories -- both the ones that are built in to the system and the ones you create -- it's important to remember that TRS-XENIX can organize material "hierarchically". That means that each TRS-XENIX directory can contain other directories. In effect, you can build a hierarchy -- a pyramid -- of directories.

For instance, when John Doe is added to the system as a new user, the **mkuser** program (see Chapter 4) automatically creates a workspace for him -- a "home directory" called /usr/johnd. /usr is a major TRS-XENIX directory that contains the workspaces of all system users, and johnd is a directory -- actually a subdirectory -- within /usr. Diagrammed hierarchically, /usr/johnd looks like this:



On the same line with johnd would be the home directories of other users -- freds, maryd etc.

As soon as John Doe begins working with the system, he may want to create directories and files in his workspace. He may want: a directory called memos in which he can store two files; a payroll directory where he can keep two more files and a computer program; and a simple, separate file called list, which he can use to remind himself of work to be done. Diagrammed hierarchically -- with directories indicated in brackets to distinguish them from files -- the results would look like this:



It's important to understand the concept of hierarchical organization. You need to keep it in mind, because, in TRS-XENIX, each file and directory is identified by a "pathname" -- a step-by-step tracing of the "path" from the top of the pyramid to the particular file or directory with which you're working.

For example, if you want to give TRS-XENIX a command involving the file called program in John Doe's payroll directory (see diagram), you determine the file's pathname by reading down to the file from the top of its directory. Therefore, the pathname of John Doe's program file, beginning with the / at the top of the diagram, would be /usr/johnd/payroll/program. (Always separate different parts of a pathname with the slash character.)

5.2 File and Directory Commands

In your work with TRS-XENIX files and directories, you need to know at least a few basic commands. The ones you will very probably use most are outlined here.

To Create A Directory

Suppose John Doe has just logged in, and wants to create a directory called memos. To do so, he types beside the prompt:

```
mkdir memos <ENTER>
```

and TRS-XENIX automatically records the existence of the new memos directory in John Doe's current directory.

To Change Directories

You may find it easier to be positioned in a directory when working on it, so it's important to know how to move from one directory to another. Each time you log in, TRS-XENIX automatically positions you in your home directory -- when John Doe logs in, he's immediately in the /usr/johnd directory (see

diagram given earlier). From there, he can create the memos directory -- but if he wants to work on it, he has to reposition himself in it. To do that, he uses a command called cd (change directory) and the pathname of the memos directory, typing this beside the prompt:

```
cd /usr/johnd/memos  <ENTER>
```

He's now in the memos directory, and can work on it.

When you're using TRS-XENIX, should you ever become confused about exactly where you are, just type beside the prompt the command:

```
pwd  <ENTER>
```

pwd means "print working directory", and TRS-XENIX starts at the top of whatever directory you're in, and prints out the path for you, ending with the directory in which you're positioned.

To Create A File

Creating a file is as straightforward as creating a directory; just be sure you're positioned in the directory where you want the file to be. For example, if John Doe is in his new memos directory, and wants to create in it a file called newproject, he simply types beside the prompt the character ">" and the name of the new file, like this:

```
>newproject  <ENTER>
```

(Please see Chapter 9 for suggestions about naming files.)

To List The Contents of a Directory

The lc command lists the contents of the directory in which you're positioned. For instance, if John Doe is still in memos and wants to check to make sure that his newproject file was entered properly, he types lc beside the prompt, and a complete listing of files is displayed, like this:

```
lc   <ENTER>
file2     file1     newproject
```

If you want to check the contents of a directory other than the one you're in, just follow the lc command with a pathname, as in:

```
lc /usr/freds/schedules  <ENTER>
```

and TRS-XENIX shows you a list of the contents of Fred S.'s schedules directory.

To Move a File

Suppose John Doe wants to move the newproject file he's created out of the memos directory and into the payroll directory. He makes sure he's positioned in memos, and then uses the mv (move) command, followed by the name of the file and the pathname of the directory to which he wants it moved. He types beside the prompt:

```
mv newproject /usr/johnd/payroll  <ENTER>
```

and TRS-XENIX transfers newproject from memos into payroll.

If for any reason John Doe wanted to put newproject into payroll but still keep a copy in memos (thus making newproject a file in both directories), he'd use the cp (copy) command instead of mv, like this:

```
cp newproject /usr/johnd/payroll  <ENTER>
```

Two notes: first, TRS-XENIX does allow you to have two files with the same name, as long as they're in different directories. And second, if you do use the cp command to copy a file, remember to change and update that file in both locations -- making a change on one file doesn't automatically change the other, so if you want them to stay identical, you have to make every change twice.

To Remove/Delete a File

You use a command called **rm** (remove), and the name of the file. For instance, if John Doe no longer needs newproject in the payroll directory, he could move to the directory payroll, and then type beside the prompt:

```
rm newproject  <ENTER>
```

and TRS-XENIX immediately deletes the file.

To Remove/Delete a Directory

Removing a directory from TRS-XENIX involves two steps: deleting all files in the directory, and then deleting the directory itself.

For instance, if John Doe wants to delete his memos directory, he first positions himself there. Then he should use the **lc** command to list the files he's about to delete; it's a double-check to help make sure he doesn't remove files he'd rather keep. Then, if he does want to delete all the files, he uses the **rm** command and the (*) wild card character (see Chapter 2) like this:

```
rm * <ENTER>
```

When all the files are removed, then TRS-XENIX lets you remove the directory itself. But note that you cannot be positioned in a directory you're removing; this is the major exception to the "be in it to work on it" rule. You must be positioned in a directory other than the one you are removing. You can do this either by using the **cd** command and the a pathname, or by simply typing two periods after the prompt, as in:

```
cd .. <ENTER>
```

That's a signal for TRS-XENIX to back you up by one directory. Once you're correctly positioned, just type beside the prompt the **rmdir** (remove directory) command, as in:

```
rmdir memos <ENTER>
```

and the directory no longer exists.

Note: When you're using the **rmdir** command, you may occasionally receive a message saying that the directory still contains files, even though you've just used the **lc** and **rm *** commands. The message usually means that there are files beginning with (.) -- like .profile -- left in the directory; **lc** doesn't list them, and the wild card (*) doesn't touch them. To get a listing of the (.) files, use a command called **lc -a**. If after seeing the list of files, you want to delete them, use this variant of the **rm** wild card command:

rm .* <ENTER>

That deletes all the files beginning with (.) and lets you go on to removing the directory. (You may, when listing with **lc -a**, see files names simply "." and ".."; those are system files which don't have to be deleted in order to remove a directory.)

To Read A File

To read a file -- to look at all the pieces of information in it -- you use a command called **cat** (catenate). If you're already positioned in the directory which contains the file you want to read, just type beside the prompt the **cat** command and the name of the file, like this:

cat file2 <ENTER>

and TRS-XENIX shows you everything in file2. If you're not positioned in the directory where the file is located, you can do one of two things -- either change to that directory by using the **cd** command, and then use **cat** to call up the file, or simply stay where you are and use the **cat** command and the file's pathname.

For instance, if John Doe is in the memos directory, and wants to read file2 in the payroll directory, he has a choice. He can type beside the prompt:

cd /usr/johnd/payroll; cat file2 <ENTER>

or

cd ../payroll; cat file2 <ENTER>

That puts him into payroll and then tells TRS-XENIX that he wants to read file2. His other alternative -- a simpler one -- is to type:

```
cat /usr/johnd/payroll/file2  <ENTER>
```

or

```
cat ../payroll/file2  <ENTER>
```

and read payroll's file2 while he is in memos.

Note that TRS-XENIX -- as shown in John Doe's first option -- allows you to give two commands at the same time, if you separate them with a semi-colon.

Please remember three important notes about TRS-XENIX files and directories:

- It's worth spending time to plan exactly how you'll set up and maintain your files and directories. Random placement of information on the system, and failure to keep that information current and orderly, can be as much of a problem in a computer as it can be in an office.
- For information on how to edit information within individual files, refer to the Appendices of this manual.
- All of the commands that involve activity with files and directories -- reading them, adding to and deleting them, moving them, etc. -- are subject to the user's permissions for those files and directories (see the next section).

5.3 Permissions: What and Who

As System Manager, you very probably won't want every user to be able to read -- and change -- every file on your system. TRS-XENIX's permission settings can help you to control access to each individual file and directory -- giving or denying access as you and your users think appropriate.

TRS-XENIX deals with three kinds of permissions: read (r), write (w), and execute (x).

READ permission has slightly different meanings for files and for directories:

- To read a file is to access the pieces of information in it; therefore, for instance, a user needs read permission to use the **cat** command on a given file.

- To read a directory -- which contains not individual pieces of information but a group of files and perhaps other directories -- is to look at its "table of contents". A user therefore needs read permission to use the **lc** command.

WRITE permission allows the user to change a file or directory -- to move it, to add to it, to remove it (or remove information from it), or to alter it in any way. A user also needs write permission to create a subdirectory, or a file within a directory -- that is, to use the **mkdir** and **>** commands.

EXECUTE permission, like read, has two different meanings. It's used in relation to a file only when that file is a program; in that case, it gives the user permission to run -- to execute -- the program. When execute refers to a directory, it means permission for the user to move into that directory -- to use the **cd** command and also gives permission to search the directory for a file to execute.

In addition to providing three permission options, TRS-XENIX provides three choices about the people to whom the three kinds of permissions can be given: an individual user (u), a group (g), and others (o).

Individual User

As TRS-XENIX understands the term, the individual user is the owner of a particular file or directory. Individual users should ordinarily have full access to files and directories located in their own home directories.

Group

When TRS-XENIX is dealing with permission settings, it understands "group" to mean all the members of the same group as the owner of a particular file. Occasionally you won't want every user on the system to be able to access a certain file or directory, but you may well want all the members of one subgroup of users --for instance, people working together on a project-- to share access. Rather than setting individual permissions for each person in the group, you can save time by setting a group permission. (Remember that, when users are added to the system, TRS-XENIX automatically makes them all members of the same group, number 5Ø. If you want to create subgroups and then grant them group permission, you can simply change the "group ID" number of the members of the subgroups and proceed to make your permission settings.)

Others

In the context of permissions, "other" is anybody --and everybody -- who is neither the owner of a particular file or directory, nor a member of the owner's group.

Any of these three "user categories" (user, group and others) can be given any of the three types of permission (read, write and execute) on a particular file or directory.

5.4 Setting Permissions

TRS-XENIX permissions are set in two ways -- by the system, and by the users, including root. The system comes with some permissions set for its own files and directories. And, whenever a new file or directory is created, TRS-XENIX automatically gives these permissions: read (r) and write (w) for the user, and read (r) only for the user's group and for others on the system. (It doesn't give the execute permission automatically because it's needed so infrequently -- only files which are programs are executable, and most TRS-XENIX files aren't programs.)

Users of the system can also set and change permissions on the files and directories which belong to them. And root -- the "super-user" -- can set and change permission settings anywhere on the system. (Section 5.6 tells you how.)

A note about root: of course root has total access, regardless of permission settings, to every file and directory on the TRS-XENIX system, and that makes logging in as root a tempting convenience. But the root rule of thumb ("log in as root only when it's absolutely necessary") holds true, because an error while you're logged in as root can do major damage to all the files on the system.

5.5 Reading Permission Settings

To look at the permission settings on a particular file or directory, you use a variant of the **lc** (list contents) command, **lc -l**, plus the pathname of the file or directory. For instance, if you want to look at the permissions on John Doe's payroll directory, type beside the prompt:

```
lc -l /usr/johnd/payroll <ENTER>
```

and you see something like this:

```
-rw-rw---- 1 johnd 11515  Nov 17  14:21 file1
-rw-rw---- 1 johnd 12337  Nov 16  1Ø:15 file2
-rwxrwx--x 1 johnd  7712  Oct 1Ø  Ø9:Ø2 program
```

The 1Ø-character letter-and-dash grouping on the left represents the permission settings. The other information is helpful, too -- in line one, for instance, the 1 is internal TRS-XENIX information referring to the number of links for that file†; johnd is the name of the file's creator, 11515 is file size in bytes; Nov 17 and 14:21 are the date and time the file was created or last changed; and file1 is the file's name.

Now look at the permission settings -- the first character tells you whether you're looking at settings for a file or a directory: a dash signifies a file; a "d" would indicate a directory. The next nine characters are divided into groups of three-- the first group for the individual user's permissions, the second for his or her group's, and the third for the permissions of others on the system. In each case, a letter signifies that permission is allowed, and a dash means that permission is denied.

Looking at line one, you see that:

- the first character (-) means that you're looking at the permission settings of a file, not a directory.

  the next three characters (rw-) mean that the individual user -- the file's owner, John Doe -- has permission to read and write on the file, and doesn't have permission to execute. (Perhaps the file isn't a program and therefore isn't executable.)

- the next three characters (rw-) mean that the members of John Doe's group have the same permissions he has.

- the last three characters (---) mean that others on the system have no access to the file: they may not read, write, or execute.

---

† Links represent a connection between disk blocks.

5.6 Changing Permission Settings

To change permission settings, make sure you're positioned in the directory where you want to make the change, and then use a command called **chmod** (change mode). As part of the command, you give TRS-XENIX information about:

- whose permission is to change: user(u), group(g), other(o), or all(a).
- whether you're adding (+), removing (-) or setting (=) permission.
- the type of permission you're changing (r, w or x).
- the name of the file or directory on which you're changing permission.

In the last line of the John Doe example, for instance, the permission settings looked like this:

```
-rwxrwx--x 1 johnd   7712  Oct 1Ø  Ø9:Ø2 program
```

Therefore, John Doe has given himself and his group permission to read, write and execute a file called program. He's also given others permission to execute the program, although not to read or write. If he decides that only he and his group should be permitted to execute the program file, he has to remove the execute permission from others. So, making sure he's positioned in the correct directory (in this case, it's payroll), he types beside the prompt:

```
chmod o-x program <ENTER>
```

If, instead, he wants to add read and write permission for others, he types:

```
chmod o+rw program <ENTER>
```

The order, or syntax, of the information after the **chmod** command is important: it's [group], [add or remove], [type of permission], and name of file. If you get the syntax wrong, TRS-XENIX helps with a reminder like this:

```
Usage: chmod [ugoa] [+-=] [rwxtugo] file
```

The items in brackets are your options. Three notes: first, the <=> indication is an option you can use to save time. If you know exactly what you want a particular file's or directory's permission settings to be, but don't want to take the time to find out what they currently are, you can use the <=>. TRS-XENIX understands it to mean "No matter what the settings are, this is what they should be", and changes them accordingly, for example:

```
chmod a=rwx file1
```

changes the permissions for all users (user, group and others) to read, write and execute.

Second, the <tugo> in the third set of brackets is for TRS-XENIX's internal use; your three basic options are still the r, w, and x. And third, after you've changed a setting, it's wise to double-check the results by using the lc -l command. If you find you've made a mistake, just reset the permissions.

5.7 Some Often-Used TRS-XENIX Directories and Files

You and your users will encounter some built-in TRS-XENIX files and directories more often than others; the ones you'll probably see most frequently are listed, and briefly described, here. For more information, see Appendix A. (The information there is particularly important if you're considering making any changes to built-in system directories and files.)

Some major TRS-XENIX directories:

/bin	TRS-XENIX commands.
/dev	Files concerning computer-related devices--for instance, disks, printers and terminals.
/etc	System maintenance programs and data files.
/lib	Library routines used in C programming. C is the programming language in which TRS-XENIX is written.
/tmp	Temporary files created by programs, editing, and mail.
/usr	User home directories, and also some commands.

Some important TRS-XENIX files:

/etc/passwd	The login name, password, group and user ID of each user on the system.
/etc/motd	The message-of-the-day file.
/etc/termcap	A list of terminals and their characteristics which are used to support certain TRS-XENIX functions.
/etc/rc	A shell -- or command-level -- script designed to set certain system functions going when TRS-XENIX is booted.

CHAPTER 6

SYSTEM MAINTENANCE

Two of the ways in which you, as System Manager, can help to keep your TRS-XENIX System running smoothly are:

- by ensuring that there's adequate disk space available to your users, and
- by making certain that your files are "clean" -- that they're consistent in both the office and computer senses of the word.

This chapter introduces you to a number of TRS-XENIX tools which can help you to keep your system running well.

6.1 The Importance of Disk Space

Available disk space quickly becomes a precious commodity on any computer system, as users compile programs, create and edit directories and files, and perform other jobs. When a system runs out of disk space, that system is essentially paralyzed; no new files can be created on it, and existing files can't expand.

One of the key jobs of the System Manager is monitoring space availability and usage. You're wise to plan periodic "disk space inventories" -- regular surveys of what space is available, where it is, and which files and directories might be using disproportionate amounts of it. (How often you should make the inventories depends, of course, on how busy your system is.)

Most successful System Managers find inventories a real administrative aid: only when you know exactly where you stand in terms of disk space can you make intelligent decisions about deleting files vs. reallocating space vs. reorganizations of materials on the system vs. buying new hard or floppy disks.

TRS-XENIX gives you three principal tools to use in monitoring disk space: commands called **df** (disk free), **du** (disk usage), and **quot** (disk quota). As you use the commands to inventory disk space, you need to remember that TRS-XENIX expresses disk space in blocks.

HINT: A disk "block" is equal to 512 bytes.

6.2 The df Command

The **df** command is both simple and useful: it reports to you the exact number of available -- free -- blocks on a disk, or disks.

If you type beside the prompt:

```
df <ENTER>
```

TRS-XENIX reports free blocks on the root file system.

If you'd rather have information about blocks available on one particular disk -- for instance, hard disk Ø -- you simply type the **df** command, /dev to indicate that you're asking about a particular "device" -- the disk -- and the name of the device itself, in this case hdØ:

```
df /dev/hdØ  <ENTER>
```

TRS-XENIX's response looks something like this:

```
/dev/hdØ  7ØØ8
```

That means that hard disk Ø has 7ØØ8 available blocks. (Of course, in order for the information to be meaningful, you have to know the block capacity of whatever disks you're dealing with. That information is in the Appendices.) With experience, you'll come to

know what figure represents a comfortable margin for you and your users, and what figure means that you're facing a potential space shortage. In general, you're wise to keep available free space at about 15% of total capacity --more if your system usage fluctuates, perhaps less if usage is relatively stable.

6.3 The quot Command

If, through using the **df** command, you discover that you have a space problem, the TRS-XENIX **quot** command can give you specific information about how many blocks are owned by each user represented on a particular disk. When you type beside the prompt, for instance:

```
quot -f /dev/hdØ <ENTER>
```

TRS-XENIX inventories hard disk Ø and produces a list of every file on the disk, plus the name of the file's owner and the file's size in blocks.

6.4 The du Command

The **du** (disk usage) command is another useful space reporting tool: it tells you the number of blocks that are currently used by individual directories, subdirectories, and files.

If you use **du** without specifying the name of a file or directory, TRS-XENIX reports the size in blocks of everything below you in the directory hierarchy.

If you specify a directory, TRS-XENIX gives you the block sizes of everything in that particular directory. The /usr directory, since it contains all the subdirectories and files of system users,

is a logical place to begin looking when you're faced with a space shortage. To look at the space situation in /usr, type beside the prompt:

```
du /usr <ENTER>
```

You might see something like this:

```
2Ø8     /usr/anthonys/admin
378     /usr/anthonys/mp
999     /usr/anthonys/junk
1585    /usr/anthonys
26      /usr/johnd/memos
235     /usr/johnd/payroll
261     /usr/johnd
```

In that example,

- 2Ø8, 378, and 999 are the number of blocks for each of the three files in Anthony Smith's home directory.
- 1585 is the sum of the 2Ø8, 378, and 999: it's the total number of blocks Anthony Smith has in use in his home directory.

Similarly, 26 and 235 are the number of blocks for each of John Doe's files, and the sum of those two numbers --261-- is the total number of blocks in John Doe's home directory.

In this instance, the System Manager would very probably confer with Anthony Smith (anthonys) about whether he genuinely needs all the space his files are using.

6.5 File System Integrity

It's important that you, as System Manager, encourage your users to create, organize, and maintain their files methodically and logically-- it's both more convenient and more efficient to work in a system where files are well maintained than in one where they're disorganized, outdated, or hard to find.

It's as important to be on the lookout for another kind of potential file problem: the internal, electronic problem called an "inconsistency". Inconsistencies -- which are said to make file systems "dirty"-- can be caused either by human error (like improper shutdown of the system) or by hardware difficulties (like faulty disk drives), but, whatever the cause, inconsistent, "dirty" file systems are more than inconveniences -- they can be serious enough to put all your files at risk.

A TRS-XENIX program called **fsck** (file system check) helps you deal with "dirty" file systems. If there's been an improper system shutdown, the five-phase **fsck** program runs automatically the next time you start up the system. And if you suspect you have a problem with files, you can run **fsck** yourself, simply by typing beside the prompt the **fsck** command, the letters /dev to tell TRS-XENIX you're dealing with a device, and the name of the device -- the disk -- on which you suspect there's a problem.

> **Important:** You should only run **fsck** from system maintenance mode (single-user). This way you are sure that there is no one using the system or accessing files.

If, for instance, you suspect a difficulty with hard disk Ø, you type:

```
fsck -y /dev/hdØ  <ENTER>
```

A report of the five checking phases of the program is displayed, plus a report on the number of files, blocks used, and blocks free on the disk. It might look like this:

```
**  Phase 1 - Check Blocks
**  Phase 2 - Check Pathnames
**  Phase 3 - Check Connectivity
**  Phase 4 - Check Reference Counts
**  Phase 5 - Check Free List
426 files 6753 blocks 7ØØ8 free
```

If **fsck** finds a problem, it reports the problem to you, and asks you if you want a correction made. Occasionally, a damaged or inconsistent file has to be deleted by TRS-XENIX in order to make the correction. In that case, the loss of information is usually minor, and TRS-XENIX tells you what was deleted. Remember that any minor deletions which TRS-XENIX has to perform are invariably less damaging to the system than a file inconsistency which is allowed to continue.

6.6 Monitoring Processes

All functions running on the system -- including system and user programs, editing, and so forth -- are referred to as "processes". Several users may each have several processes running simultaneously, and you, as System Manager, may want to check and see what your system is doing at a particular time.

A command called **ps** (process status) lists information about processes currently running at the terminal from which you type the command. A variant of the **ps** command -- **ps -a** -- gives you a list of all the processes currently running on a multi-terminal system. When you type beside the prompt:

```
ps -a  <ENTER>
```

you might see something like this:

```
PID  TTY  TIME  CMD
 2   co   1:Ø7  -sh
73   Ø1   2:Ø9  tsh
74   Ø1   Ø:1Ø  sh -c ps-a
77   co   Ø:59  ps -a
```

PID — is the "process ID" -- a number TRS-XENIX generates and uses to identify the process.

TTY — indicates the number of the terminal from which the process is running. (It's possible, on TRS-XENIX, to run several processes simultaneously from a single terminal.)

TIME — indicates the total amount of time for which the process has been running.

CMD — is the name of the process's command or program.

6.7 Daemon Processes

There are in TRS-XENIX a number of programs that run automatically whenever you use the system. Called "daemons" (pronounced like "demons"), these programs periodically check the system and perform basic system functions. For example:

- The **update** daemon is a safeguard. It helps to keep your disk current by automatically writing information from your computer's memory onto your disk every thirty seconds. In the event of a problem -- a system "crash" or an abnormal shutdown -- the information that's recorded on your disk is still very current.

- The **lpd** daemon superintends the operation of the line printer.

- The **cron** daemon acts like an internal alarm clock for TRS-XENIX. It allows you to execute commands and jobs at times you specify in advance. It repeatedly looks in a file called /usr/lib/crontab for instructions to perform these functions.

Ordinarily, the commands which start the daemons running are stored in the /etc/rc file, a program which runs automatically each time you boot TRS-XENIX. The /etc/rc file also tells the system to perform other functions. It may, for instance, contain a message that greets you when you log in, or it may direct the system to ask you for the time. (If you should want to edit the /etc/rc file, you must log in as root. Check the Appendices if you want editing instructions.)

CHAPTER 7

BACKUPS

Backup copies of the contents of your system are like good insurance policies: you hope you never need them -- but they're necessary protections you can't afford not to have.

As System Manager, you're responsible for making a backup schedule, and then carrying it out. No matter how busy you are, you should resolve never to skip a scheduled backup, because a complete and current backup is the only protection your users have against time-consuming and costly losses of their programs and data. Inevitable user errors (like writing over a file), in addition to less likely mishaps (for instance, hardware failure or accidental damage to software), can have serious -- sometimes permanent -- negative consequences if your system isn't properly covered by backups.

For reasons of economy and efficiency, most people choose to make their backups -- and other kinds of copies -- on floppy disks; if that's the way you choose to handle backups, you should remember that, before you can write anything onto floppy disks, you have to format them. This chapter tells you how to do so, and also discusses:

- how to make individual disk copies for users who request them.

  how you might schedule backups.

- how to use two TRS-XENIX programs -- **sysadmin** and **tar** -- to make several kinds of backups.

- how to obtain listings of backups.

- how to restore backups and other kinds of copies to your system.

- how you might archive your backup materials.

7.1 Formatting and Copying Floppy Disks

TRS-XENIX can't write material onto a disk that hasn't been properly formatted, so, before you make backups - or any other copies -- you have to format the floppies on which you want to put data on.

The program you use in all your formatting, and also in making individual disk copies, is the same one you used to install TRS-XENIX -- **diskutil**. Remember that **diskutil** is a standalone program, one which is independent of, and separate from, TRS-XENIX.

That means you must shut down your system (using one of the methods described in Chapter 3) to inactivate TRS-XENIX before you can use **diskutil**. You should schedule formatting and copying well in advance, so that your users can plan not to be on the system at the scheduled times, and also so that you can make sure that your supplier has time to provide as many floppies as you need.

> **Important notes:** make sure that each floppy you work with has a write-enable tab. Also, **diskutil** can format or copy both single-sided and double-sided floppies. Also,

Formatting With diskutil

> HINT: Since you must exit TRS-XENIX to format floppy disks, you may want to format them at the end of the day, after the system is shut down (or at the beginning of the day, before it's booted.)

Have your floppies ready beside you and shut down TRS-XENIX. Then press the reset button. When the TRS-XENIX Boot prompt is displayed, type **diskutil** beside the colon, like this:

```
XENIX Boot
:diskutil <ENTER>
```

The following message and prompt is displayed:

> At any time you may type <break> or press reset to abort the procedure. The backspace key may be used to correct single characters of your input. End each line you type by typing <enter>

Diskutil: format or copy hard or floppy disks? (h or f)?
Type <f> and then press <ENTER>.

Then you're asked a series of questions:

Copy or format (c or f)?
Type <f> and then press <ENTER>.

Floppy drive number (Ø..3)?
Answer with the number of the drive into which you put the floppy to be formatted. Then press <ENTER>.

TRS-XENIX format, IBM single-density
or IBM double-density format (x or i)? <x> <ENTER>
(Occasionally, if you're formatting a floppy to be used with a mainframe computer, you use the other option.

Then the following is displayed:

About to format floppy disk in TRS-XENIX format.
Insert disk in drive x.
type <ENTER> to proceed or <BREAK> to abort:

If you see the message and you want to start the formatting, press <ENTER>; if you wish to stop at this point, press <BREAK>. If you press <ENTER>, diskutil proceeds with the formatting, and, when it's done, it shows you this message:

Floppy disk in drive x successfully formatted.

IBM™ is a trademark of International Business Machines, Inc.

Copying With Diskutil

Occasionally, your users will ask you for a copy of a particular floppy disk, and you'll use **diskutil** to make such a copy. (Remember that you can make copies only on formatted disks.)

With **diskutil**, the copying procedure is essentially the same as that used in formatting, although naturally you respond to the questions with answers appropriate for copying. The major differences are that you are asked for a "source drive number" and a "destination drive number" -- those are the numbers of the drives into which you put, respectively, the disk to be copied (the source) and the blank disk on which you want the copy made (the destination). And, at the end of the process, instead of reporting a successful formatting, **diskutil** says:

    Disk copy and verifying complete

If, after you've finished formatting and/or copying a disk, you want to do more work using **diskutil**, press the reset switch, and, when you see the "XENIX Boot" message, again type **diskutil** beside the colon and proceed with the work. If, instead, you want to bring TRS-XENIX back up on your system, you still press the reset button, but when you see the "XENIX Boot" message, you simply press <ENTER> and continue with normal booting and login procedures as described in Chapter 3.

Note: As you work with the system, you'll come to know roughly how many floppy disks you routinely use for backups and individual copies over a particular period of time. Many system managers find it efficient and convenient to regularly format batches of floppies so they're ready when they're needed.

7.2 Scheduling Backups

It's best to schedule two kinds of backups for your system: periodic full backups in which you copy everything on the system, and, in the intervals between full backups, incremental backups to record the changes made to your files since the last full backup.

Naturally, the backup schedule you set depends on the size, and degree of activity, of your own system. Many systems warrant a periodic backup once a week, with incremental backups every day, but, no matter what the size of your system, even if it's quite small, you should schedule periodic backups at least once a month.

Whatever schedule is appropriate for you, keep in mind that it's best to schedule backups when there are few, if any, users on the system; you don't want to risk having files changed just as they're being copied.

7.3 The sysadmin Command: Four Backup Options

No matter how often you do it, backup-making is a major job, and instead of using **diskutil** to copy one disk at a time, you can use another TRS-XENIX tool especially designed for backup work: a program called **sysadmin**. **Sysadmin** offers you four major options, programs for:

- making full-system backups (**sysadmin** calls them periodic backups).
- making incremental backups (**sysadmin** calls them daily backups; they're copies of only files which have been changed since the last full backup).
- obtaining a list of what's been backed up.
- restoring backup material from floppies to the system.

You must be logged in as root to run **sysadmin**. Before you begin, remember to have at hand an adequate number of blank (or reusable) single- or double-sided floppy disks with write-enable tabs. (A TRS-XENIX system with an eight-megabyte hard disk can use seven or eight double-sided floppies in a full, periodic backup.)

To run **sysadmin**, type beside the root prompt:

```
/etc/sysadmin <ENTER>
```

and TRS-XENIX produces for you a menu of "File System Maintenance" tasks:

```
File System Maintenance
-----------------------

Type 1 to do a daily backup
     2 to do a periodic backup
     3 to get a backup listing
     4 to restore a file
     5 to quit
```

If at this point you want to return to the root prompt, just type:

```
5 <ENTER>
```

If you want to proceed, just type the appropriate number, and press <ENTER>.

Options 1 and 2: MAKING BACKUPS

Whichever option you choose, you're asked whether you're putting your backups onto single- or double-sided floppies. You are prompted to insert your first disk; after you do so (remembering to press <ENTER>), the system responds with the current date, followed by the date of the last backup.

If the system has never before been backed up, you see on your screen the phrase "the epoch", and **sysadmin** then automatically does a full (periodic) backup, even if you chose the "daily" option from the task menu.

As the backup proceeds, you see the red disk drive light and hear the floppy disk drive. On your screen, a series of messages is displayed; the only one to which you need to respond is the "change volumes" instruction. When you see it, remove the first disk and insert the next, being sure to press <ENTER> when the disk is in place. You might be prompted to change volumes -- disks -- several times before the backup is complete; just remember not to change a disk until and unless you're prompted to do so.

If you make a mistake at any time during the process, just press <BREAK> and begin again. Occasionally, **sysadmin** encounters a problem -- for instance, an incorrectly formatted disk, or a malfunctioning floppy disk drive. In that case, a message like this is displayed:

```
dump: write failure on /dev/rfdØ
```

and you're returned to the system prompt. After you've investigated the problem (checking your disk and drive, and perhaps substituting another blank disk and/or another drive), just reactivate the program by typing beside the prompt:

```
/etc/sysadmin  <ENTER>
```

OPTION 3: BACKUP LISTINGS

You may find it useful to have a record of the files you've backed up, and **sysadmin** option 3 can provide such a record. After you've finished whatever backups you're doing, return to **sysadmin** by typing beside the prompt:

```
/etc/sysadmin <ENTER>
```

and choose option 3, and press <ENTER>. You're prompted to reinsert the backup disks in the same order you used when making them. **Sysadmin** reads the names of files which are on the disks, and record the names in a TRS-XENIX file called /temp/backup.list. Then you can print the list of files on your line printer, using the **lpr** command and the name of the storage file, like this:

```
lpr /temp/backup.list  <ENTER>
```

OPTION 4: RESTORING A BACKUP FILE

If you want to restore a file to the system from a backup floppy, go to **sysadmin** -- using the usual /etc/sysadmin command -- and choosing option 4.

The program **sysadmin** uses a pathname to find a file; however as each file is extracted, it is assigned a unique number. Each file is then stored in the current directory with its filename being the number that was assigned to it. You can then move the file to the proper directory and rename it to its original name.

> Note to applications users: TRS-XENIX provides a special program for applications backups. For information, see Appendix B.

7.4 The tar Command

TRS-XENIX's **tar** program is a convenient way to copy a small number of files or directories from your hard disk to floppy disks, and then to restore them to your hard disk. You must be logged in as root to use **tar**.

To use **tar** to copy, insert a formatted floppy into a floppy drive, and then specify: the "copy" command variant (which is <cvf>); the floppy drive number; and the file/files or directory/directories you want to copy. Your command might look like this:

```
tar cvf /dev/rfdØ file1 file2 file3  <ENTER>
```

It indicates that file1, file2, and file3 are to be copied onto the floppy diskette in floppy drive Ø. The "r" in the device specification denotes "raw data" which is a faster form of data transfer.

If you specify directories instead of files, be aware that **tar** copies everything that's below the specified directory/directories in the directory hierarchy.

To restore **tar**-made copies to your hard disk, insert the floppy with the copies on it, again make sure you're logged in as root, and then specify: the "restore" command variant (which is <xvf>); the floppy drive number; and the file/files or directory/directories you want restored. Your command might look like this:

```
tar xvf  /dev/rfdØ file1 file2  <ENTER>
```

7.5 Archiving Your Disks

You'll find that, over time, you'll accumulate a great number of floppy disks, including the floppies you used to install TRS-XENIX, disks from any applications packages you may have, copies you've made for your users, and all the backup copies you've produced.

Right at the beginning, you're wise to develop a simple, logical way to label, organize, and store your floppies. Since they contain valuable -- and sometimes irreplaceable -- data, you should give a high priority to safety and security. You may even want to consider making double backups of irreplaceable material, and storing those backups offsite, so that, should there be damage to your computer area, you'll still have the materials needed to help get your system up and running again.

You're also wise to develop a consistent policy for saving backups, and to resist the temptation to recycle backup disks immediately; there'll probably be many times when your users ask to have a relatively old file restored. One feasible approach is to save the full backups indefinitely, and the incremental backups for at least several weeks, rotating the incremental floppies back into use for future copy-making.

CHAPTER 8

TRS-XENIX SYSTEM SECURITY

Although security needs vary greatly among computer systems, every System Manager should give serious thought to protecting data and programs from unauthorized access. This chapter discusses ways to safeguard your system by making good initial decisions, by regularly using certain TRS-XENIX tools, and by adopting physical security precautions.

8.1 Initial Decision-Making

Some of the early decisions you make as System Manager have a continuing impact on the security of your system. Set up new users only as they're needed, and only for people who really need them. If your new users decide to change their passwords at initial login time, make sure they understand the importance of choosing unusual, difficult-to-guess combinations of letters and numbers.

Think hard, too, about the ways in which you assign permissions to files and directories: you need to strike a balance between convenience for those who need access and protection against those who don't.

8.2 Ongoing Use of the System

TRS-XENIX gives you certain tools to help you maintain system security.

The system's standard logging out procedure helps control unauthorized access to your files. Remind your users that they

must log out properly each time they've finished their work with TRS-XENIX; if they leave without logging out -- without "locking the door" behind them -- all their files and directories are open and vulnerable to unauthorized use.

Continued password security is a must, and TRS-XENIX helps by never displaying a password as it's being typed in. But if a user's password should somehow become known (or is suspected to be known), it should be changed immediately. The procedure to change a password is the same as that used for a forgotten password; see Chapter 9.

An error in the root mode can do major harm to your system; therefore, for security's sake, be careful how you assign and use root's wide powers. Limit knowledge of the root password to only those who really need it, and always remember the root rule of thumb: log in as root only when it's absolutely necessary.

8.3 Options for Physical Security

You may want to consider taking some precautions which relate to the physical security of your system. Some options include:

- removing the key from the hard disk when the system is shut down.

  locking up important floppies (backups, for instance, or floppy-based file systems you're not using at the time).

  storing important floppies (particularly full-system backups) offsite.

  instituting a system to keep unauthorized people out of the computer work area.

  adopting rules to protect your system from physical damage -- from cigarette smoke, drink spills, ballpoint pens used on disk labels, and so forth.

CHAPTER 9

TROUBLESHOOTING

If you follow the instructions in this Guide, your TRS-XENIX system should run smoothly and well. But if you run into the occasional problems inevitable in anything as complex as a computer system, you'll want to know how to deal with them. This chapter gives you some suggestions for troubleshooting.

9.1 Jammed Line Printer

If your printer jams up for no apparent reason (as opposed to needing a ribbon change or a paper alignment), you use a series of commands to accomplish a number of things, for instance to:

- discover what process "owns" the printer.

  stop the process.

- get into the directory where printer files are stored.

- "unlock" the printer .

- ready the printer to accept another printing job.

Begin by logging in as root. Then use a command called **ps -ax** (a variant of the **ps -a** command described in Chapter 6) to list all processes currently running on the system. The **ps -ax** variant shows you not just processes initiated at terminals, as **ps -a** does, but also "invisible" processes initiated by TRS-XENIX itself.

For example, when you type beside the prompt:

```
ps -ax <ENTER>
```

you might see something like this:

```
PID  TTY  TIME CMD
32   co   Ø:Ø1 lpd
```

That means the process currently running is **lpd** (the line printer daemon --see Chapter 6), and that TRS-XENIX has given it a process ID number of 32. (If several processes are running at the time you enter the **ps -ax** command, you have to look carefully to figure out which one is jamming your printer; when you decide which it is, note its process ID number.)

When you know which process you're targeting, type the following series of commands, substituting the appropriate process ID number on the first line. What you'll be doing is killing the process, changing to the printer file directory, and "unlocking" the printer:

```
kill -9 32  <ENTER>
cd /usr/spool/lpd  <ENTER>
rm -f lock  <ENTER>
```

Then choose another file you'd like to have printed-- for instance, memos1 -- and "queue" it up (make it ready for the printer to type), like this:

```
pr memos|lpr &  <ENTER>
```

and the printer should unjam and begin to operate normally.

9.2 Forgotten Password

If a user forgets his or her password, it's not retrievable from the computer. As System Manager, you have to create a new password for the user.

It's a simple process. After agreeing with the user on what the new password will be, log in as root, and type beside the prompt the command **passwd** and the user's login name, like this:

```
passwd johnd   <ENTER>
```

Then TRS-XENIX prompts you twice to type in the new password:

```
New password:
Reenter password:
```

You won't be able to see the password as you type it in (because TRS-XENIX never displays a password), so type with care. And remember to press <ENTER> at the end of each line. When you press <ENTER> the second time, John Doe's new password is automatically substituted for the one he forgot.

9.3 System Out of Space

Use TRS-XENIX's system maintenance tools (described in Chapter 6) to find and delete expendable files. If you're chronically short of space, you may want to consider a regular user file clean-up program, and/or expanding your system's capability (see Chapter 10).

9.4 Damaged Files

If files are damaged through a mishap (anything from a power glitch to an accidental write-over by a user), go to your backup files and use either **sysadmin** or **tar** as appropriate (see Chapter 7) to substitute the backups for the damaged files.

If system files are damaged, you must be in "system maintenance" mode to restore them. Enter system maintenance by typing in the root password instead of <CTRL><D> at boot up. Then follow the normal restore procedures following this example: if the /etc/rc file is lost or damaged, use this command to recover it:

```
restore xf /dev/rfd0 /etc/rc
```

The /etc/rc file is restored from the disk in floppy disk drive 0.

9.5 Problems with Terminal(s)

Occasionally, after you use <BREAK> to terminate a TRS-XENIX program which has screen displays, you might find that you can't call anything else up onto your screen. In that case, the screen is said to be in a "no echo", or "raw", mode, and it's necessary to tell TRS-XENIX that you want your screen displays back. To do so, you first type:

<CTRL><J>

being sure to hold down the <CTRL> key while typing <J>. If TRS-XENIX responds with an error message, just try the <CTRL><J> combination again. You don't have to press <ENTER> and you won't see the command as you type it.

Then, on the next line, type:

stty echo -raw <CTRL><J>

Again, as before, you don't have to press <ENTER>, you hold the <CTRL> key down as you type <J>, and you won't see the command as you type it -- but as soon as you finish, your screen should begin to display whatever material you call up.

9.6 Forgotten Root Password

There is nothing to be done. The only remedy is prevention: make **sure you don't forget the root password**, because it's critical to the operation and maintenance of the whole system. The only recovery from a forgotten root password is to reinstall the TRS-XENIX file system from the distribution diskettes.

9.7 Special Characters in File Names

When you're naming a file or a directory, you're wise to use letter-number combinations and to stay away from special characters, which often have particular meanings for TRS-XENIX. Characters to avoid include these:

- < > . / ? { } ' " ; | ` ( ) * & ^ $

Those characters in file names often prevent TRS-XENIX from responding to commands concerning the file. If you run into that situation, try renaming the file with the **mv** command. For instance, if you have a file named "-x" and want to rename it "junk", position yourself in the file's directory and type beside the prompt:

```
mv -x junk <ENTER>
```

Then you may be able to work normally with the file. If renaming doesn't work, you must remove the file altogether, using the usual **rm** command. If that doesn't work -- and special characters can make it difficult to remove a file -- there's still a last resort available to you. Move everything but the file in question to another directory. Then type ".." to position yourself above the directory which now contains only the problem file, and use a command called **rm -rf** to remove the directory, and of course the problem file it contains. (The **rm -rf** command is a forceful variant of the **rmdir** command.)

For example, if the directory you want to remove is called memos2, type beside the prompt:

```
rm -rf memos2 <ENTER>
```

Then, if you want, you can reposition the files you relocated earlier.

Note: Don't try to delete oddly named files with wild card characters unless you've carefully thought through the situation. Wild card characters are handy -- but they can also lead to unwanted deletions.

9.8 Runaway Process

Once in a while, either human or mechanical error causes a "runaway process" -- either a process you can't stop, or a continuous unwanted stream of output to your terminal. If you're faced with a runaway process, try the following (in the order in which they're suggested):

1. Wait until the process finishes. This is by far the safest course of action, unless the process is causing harm.

2. Try pressing <BREAK>.

3. If the process prevents you from using your terminal, go to another terminal -- if you have one -- and use the **ps -a** command (see Chapter 6) to determine the ID number of the unwanted process. Then type beside the prompt:

   su root <ENTER>

   the root password, and <ENTER> Next, type in a series of **kill** commands -- **kill -2**, **kill -3**, and **kill -9** -- with the process ID number beside each. Wait a few seconds after each **kill** command to see if the process stops. For instance, if the ID number of your unwanted process were Ø71, your list of commands might look like this:

   ```
   kill -2 Ø71 <ENTER>
   kill -3 Ø71 <ENTER>
   kill -9 Ø71 <ENTER>
   ```

   The third command is sure to work, but it's so powerful that it occasionally has an impact on TRS-XENIX's file systems. If you suspect that **kill -9** has affected your files, you should run **fsck** (see Chapter 6) to make certain the files are in good order.

4. If a runaway process keeps you from using the terminal, and you don't have another terminal from which you can issue the **kill** commands, you may have to reset the machine.

   In that case, be sure to wait several minutes before pressing the reset switch that way, you can be certain that TRS-XENIX has made your disk as current as possible before you reset. Always reset only as a last resort -- since it's not the correct way to shut down the system, it can cause file inconsistencies, which you may have to correct with **fsck**.

CHAPTER 1Ø

ADDITIONAL FILE SYSTEMS

A basic TRS-XENIX system uses one hard disk. On the disk are files, and also the information TRS-XENIX needs to locate them; together, the files and the locator information are called a file system. A file system can use virtually all the space on the disk, except for an area reserved for "swapping" -- the process of switching data back and forth between the disk and the computer's main memory.

As you work with your system, you may find that you need more file space than your hard disk can provide. In that case, you can easily create additional file systems on either secondary hard disks or floppy disks. (You have to decide which is more appropriate for you, your Radio Shack dealer can provide you with useful information for making this decision.)

Whether you decide to use hard disks or floppies for your additional file systems, you have to follow the four basic steps outlined in this chapter: formatting each disk, creating a file system structure on it, mounting it onto the original TRS-XENIX file system, and unmounting it.

1Ø.1 Formatting the Disk

Since no computer system can write anything onto a disk that hasn't been properly prepared -- formatted -- your first step is to use TRS-XENIX's **diskutil** program to format your hard or floppy disk.

To format a floppy, follow the directions in Chapter 7, Section 1 (the part called Formatting With **diskutil**). Remember that the floppy should have a write-enable tab.

To format a secondary hard disk, you need to have the disk's media error map handy. Then shut down TRS-XENIX and press the reset switch. When you see on your screen display:

```
XENIX Boot
:
```

type beside the colon:

```
diskutil <ENTER>
```

The following prompts are displayed. The answers you should give are indicated here:

```
Copy or format (c or f)?  <f> <ENTER>

Hard or floppy disk (h or f)?  <h> <ENTER>

Hard drive number (Ø..3)?
     Enter the appropriate drive number (Ø-3), then press
     <ENTER>.

TRS-XENIX format, IBM single-density
   or IBM double-density format (x or i)?  <x> <ENTER>
```

Diskutil tells you to ready the disk, and to press <ENTER> to proceed (or <BREAK> to abort). Press <ENTER> and the formatting begins. The cylinder and side numbers are displayed while the formatting is in progress. If the disk is defective, a descriptive error message is displayed.

Next you're prompted to enter the information from the TRACK and HEAD columns of your media error map. **Diskutil** repeatedly prompts you with:

```
enter numbers or "done":
```

If, for instance, your list looks like this:

TRACK	HEAD	BYTE COUNT	LENGTH
133	ØØ	Ø1333	Ø2
174	Ø1	Ø9826	Ø5

then your completed entry would look like this:

```
enter numbers or "done": 133,Ø  <ENTER>
enter numbers or "done": 174,1  <ENTER>
enter numbers or "done": done <ENTER>
```

with you having typed in the numbers and pressed <ENTER>. If your list is blank, simply type:

done <ENTER>

You're then told that the hard disk is about to be formatted. You're also told the approximate length of time the formatting takes -- usually at least 15 to 30 minutes. The message looks like this:

About to format hard disk drive x
This will take about xx minutes.
Type <ENTER> to proceed or <BREAK> to abort.

Press <ENTER> if you want to go ahead; then, when the formatting is complete, this message is displayed:

Hard disk successfully formatted.

When your disk -- hard or floppy -- is formatted, you're ready for the next step: creating a file system on it. You should now reset your computer system.

10.2 Creating a File System on the Disk

You use a command called **mkfs** to create a file system on the disk -- to ready the disk to accept TRS-XENIX's hierarchical directory-and-file structure. Before you enter the command, you have to know:

- the number of the disk drive you're using for the disk on which you're creating the file system. For the purposes of this command, floppy drive 0 is called rfd0, hard disk drive 1 is rhd1, and so forth.

- The total number of blocks on the disk, the number of sides and the number of sectors per track. Information on the number of blocks on the disk is located in the Appendices. (Floppy disks have 16 sectors/track and hard disks have 17 sectors/track.)

To use the command, type beside the prompt:

/etc/mkfs /dev/device blocks number sectors

The value number is a "1" for hard disks and a "2" for floppy disks.

For instance, if you're using a single-sided floppy in drive Ø, your command would look like this:

/etc/mkfs /dev/rfdØ 1224 2 16 <ENTER>

For an 8-meg hard disk in hard disk drive 1, your command would look like this:

/etc/mkfs /dev/rhd1 16966 1 17 <ENTER>

Your new file system is now created, and you're ready for the next step: mounting the system.

1Ø.3 Mounting Your New File System

In order to be able to access and work on your new file system, you have to mount it -- "mount" means attach it electronically to your original TRS-XENIX file system. To do that, you use a command called **mount**.

As part of the command, type in the appropriate disk drive number (this time without the "r" prefix -- for instance, just fdØ, hd1, and so forth). You also type in the name of the TRS-XENIX directory (in the root file system) where you want the new file system to be mounted. TRS-XENIX provides a directory specifically for that purpose; it's called /mnt. (You should always leave /mnt empty, and never use it for any purpose other than mounting.)

If, for instance, you want to mount the floppy in drive Ø onto TRS-XENIX's /mnt directory, you type beside the prompt:

```
/etc/mount /dev/fdØ /mnt <ENTER>
```

You can also mount a secondary hard disk file system onto /mnt. But many people find it more convenient to save /mnt for floppy-based file systems, and to create in TRS-XENIX a separate new directory onto which they mount their hard disk systems.

Creating such a directory is simple: just enter the **mkdir** command and the name of the secondary hard disk. If the drive name were hd2, for instance, your command would look like this:

```
mkdir /hd2 <ENTER>
```

The new directory, which has the same name as the disk drive, is automatically created (in this case the directory would be called hd2).

Then, to mount your secondary hard disk onto your new directory, you use the regular **mount** command, specifying the drive number of your secondary disk and the name of the directory onto which you're mounting the disk's file system. Your command might look like this:

```
/etc/mount /dev/hd2 /hd2 <ENTER>
```

When you've mounted your new file system -- whether it's on a floppy or a hard disk -- it becomes part of the TRS-XENIX hierarchy; in effect, everything you put into it becomes part of the single TRS-XENIX pyramid. You work on your new file system just as you do on the original one.

1Ø.4 Unmounting

When you've finished your work with the additional file system, you should unmount it -- detach it from the original TRS-XENIX system. To do so, make sure you're not positioned in the file system to be unmounted, then use the command **umount**.

As part of the command, specify the name of the new file system's disk drive. For example, if you want to unmount the floppy in drive Ø, type beside the prompt:

```
/etc/umount /dev/fdØ <ENTER>
```

and the system on drive Ø is unmounted.

Three notes: first, if you know you're going to be using your new file system often, and want to avoid having to repeat the mount procedure for each use, you can place the **mount** command in the /etc/rc file, which is read by the system when it's first booted. If you do this, you must have the disk to be mounted ready before the boot up begins. (For editing instructions, see Appendix X, and remember that you must be logged in as root to edit /etc/rc.)

However, you should note that if you use mounted file systems mounted by /etc/rc, you must be logged in as root to do any system maintenance. The system maintenance mode (single-user) does not read in the /etc/rc file, therefore the mounted file systems are not recognized.

Second, when you're using floppies with your system, make sure that they're not in their drives when you're powering up or down; the files on them could be damaged. The correct sequence before you use a floppy is: POWER UP, INSERT FLOPPY, BOOT THE SYSTEM. After you've finished with the floppy, the sequence is: SHUT DOWN THE SYSTEM, REMOVE FLOPPY, POWER DOWN.

Third, using the **shutdown** command automatically unmounts any additional file systems currently mounted onto the original system. Therefore, if you've finished your work with any additional systems and know you'll soon be shutting down TRS-XENIX for the day, you needn't unmount the additional systems one by one.

CHAPTER 11

THE MULTI-USER ENVIRONMENT

If your system has more than one terminal, you'll find that TRS-XENIX handles simultaneous users smoothly and well. But, as System Manager, you should be aware of some special cautions and considerations -- described in this Chapter -- which apply to a TRS-XENIX system with more than one terminal.

11.1 Bringing Down the System

Be extremely cautious about shutting down your system when other users are working on it, because a sudden shutdown could damage -- or even destroy -- their data and programs. The safest way to bring down a multi-user system is to use **shutdown**. Otherwise you would need to do each of the following before turning off the system (**shutdown** does each of these automatically):

Use the command called **who** to tell you if other users are currently working with TRS-XENIX. Just type the command, and TRS-XENIX shows you the login names of any other users at work. It also lists for you the terminals at which they're working.

Then use the **wall** command (see Chapter 4) to alert them to the coming shutdown; that way, they can wrap up their work and log out with no risk to their materials. Of course, the normal TRS-XENIX shutdown procedure also automatically gives users a shutdown warning.

Even if no other users are logged in, processes may still be running on TRS-XENIX -- and a shutdown, of course, terminates them, and possibly leaves files in an inconsistent state. Before you shut the system down, use the **ps -a** command (see Chapter 6) to see what, if any, processes are currently running on the system; you may need to wait until they're done before you shut down.

11.2 Connecting a DT-1 Data Terminal to Your Computer

You can find detailed information about connecting the DT-1 to your computer in the Data Terminal Owner's Manual, but the basic steps are listed here. Before you begin, you should have two items on hand: a DB-25 cable (cat. no. 26-44Ø3) and a null modem adapter (cat. no. 26-1496). Then follow these steps:

1. Connect one end of the DB-25 cable to the DT-1's RS-232 jack.

2. Connect the other end of the cable to the female plug of the null modem adapter.

3. Connect the adapter's male plug to a Serial Channel A or B of your computer.

4. If your computer requires it (check your owner's manual), insert a terminator plug into any unused serial channel.

11.3 Setting Input/Output Parameters for the DT-1

Follow these instructions to set the I/O parameters for the DT-1 terminal:

1. Connect the DT-1 as described in Section 11.2. You must use the null modem (26-1496) as described.

2. Power up your computer system and the DT-1.

3. Set the Input/Output Parameters. Type:

   <CTRL><SHIFT><ENTER>

   to display the I/O Parameter Menu.

4. Set the parameters to the following:

```
Ø Ø 1 Ø  Ø Ø 1 Ø
Ø Ø Ø Ø  1 Ø Ø 1
       Ø 1 Ø Ø 1
```

11.4 Setting Up Multiple Terminals

Your TRS-XENIX system is programmed to accommodate a wide range of terminal types -- but the system has to know what kind of terminals you're connecting to it, so that it can calibrate itself to the terminals' particular characteristics and capabilities.

Terminal Types

Your console is automatically set to accommodate two DT-1 terminals. TRS-XENIX calls them the "adds25" terminal type, and expects the terminals you connect to be set to the DT-1's Adds25 emulation mode. That information is stored in a TRS-XENIX file called /etc/ttytype. If you're connecting a DT-1 to your console, just follow connection instructions, and proceed directly to enabling your terminal (see below). If, however, you're connecting a terminal other than a DT-1, you have to let TRS-XENIX know.

A file called /etc/termcap stores the characteristics of all commonly used terminals. The following list gives the names and codes for the terminals pre-configured in /etc/termcap:

NAME	CODE
VT 1ØØ	vt1ØØ
VT 52	vt52
ADM 3a	adm3a
ADM 5	adm5
Televideo 91Ø	tvi91Ø
ADDS 25	adds25

To let TRS-XENIX know what kind of terminal you'll be using, edit the /etc/ttytype file to include one of the above names. At that point, TRS-XENIX can work with your terminal -- just as soon as you've enabled it.

Enabling A Terminal

Before you can use a terminal, you have to enable it. First, check which serial channel you're using for the terminal. Then, to enable a terminal at Serial Channel A, type:

```
enable tty01 <ENTER>
```

To enable a terminal at Serial Channel B, type:

```
enable tty02 <ENTER>
```

Then you're ready to use your terminal with TRS-XENIX.

Disabling A Terminal

If you have to disconnect a terminal for any reason, be sure you first disable it. Again, check the serial channel the terminal is using, and type either:

```
disable tty01 <ENTER>
```

or

```
disable tty02 <ENTER>
```

and proceed with the disconnection.

11.5 Setting Your System For Remote Use

If you plan to use a terminal from a remote location (i.e. via a telephone modem), there are some settings which must be changed.

1. Edit the etc/ttys file (using ed described in Appendix X) to look like this:

```
1hconsole
Ø3ttyØ1
Ø3ttyØ1
```

   The '3' in the second column indicates that the terminals are remote.

2. Enable the necessary tty ports as described in Section 11.4.

3a. For 3ØØ baud, set the I/O parameters to the following:

```
Ø Ø 1 Ø  Ø Ø 1 Ø
Ø Ø Ø Ø  Ø 1 Ø Ø
       Ø 1 Ø Ø 1
```

3b. For 12ØØ baud, set the I/O parameters to the following:

```
Ø Ø 1 Ø  Ø Ø 1 Ø
Ø Ø Ø Ø  Ø 1 1 Ø
       Ø 1 Ø Ø 1
```

4. Now, when you log into TRS-XENIX, the system will automatically switch between 12ØØ and 3ØØ baud until it receives something it understands. You may see some garbage displayed. If this happens, press <BREAK> until the word "login:" appears and then proceed with your work.

11.6 Setting User ID for Multiple Systems

If you're working in an environment with several TRS-XENIX systems, and users want to be able to work on more than one computer -- moving their files back and forth on floppy disks --they have to have valid logins and home directories on each system.

In addition, each user has to have the same user ID number on every system; otherwise, he or she won't be able to switch file systems between computers. Remember that the **mkuser** program, through which you create user accounts on the system, sequentially assigns a different user ID number for each account. Therefore, you have to edit the /etc/passwd file to make a user's ID numbers identical on each of your TRS-XENIX systems. Check Appendix X for information about how to edit.

APPENDIX A

SOME TRS-XENIX FILES AND DIRECTORIES

If you're considering changing, moving or deleting any of the basic TRS-XENIX system files or directories, it's absolutely essential that you read this Appendix very carefully, noting particularly the names of files you shouldn't touch under any circumstances.

When you do edit system files, make sure you have a recent full backup of your system -- just in case you run into a problem; in that event, you're able to restore lost or damaged files to your system.

Understand, too, before you touch system files, that you run the risk of damaging the system so severely that you may not even be able to reboot, and/or to use your system productively. If that happens, you have to recreate the TRS-XENIX system from scratch (following the installation process described in Chapter 3), and boot the system again; then you have to restore all your files from the most recent full backup.

A.1 Files You Should Never Touch

You may have some, or all, of these files in your system. Don't modify or delete them; if you damage them, you risk not being able to boot your system again.

/fdboot	/hdboot	/diskutil
/xenix	/z80ctl	

A.2 /bin

The /bin directory contains many TRS-XENIX commands you'll very probably use. The ones listed here should never be removed from the directory.

basename	echo	passwd	su
cp	expr	rm	sync
date	fsck	sh	tar
dump	login	sleep	restor
dumpdir	mv	stty	

Note: Don't remove a file called "[" from the /bin directory; it's needed for the operation of system shell (command-level) scripts.

A.3 /dev

The /dev directory contains files which control access to peripheral devices -- disks, printers, and so forth. You shouldn't change or delete any of the ones listed here, since they're used by essential TRS-XENIX commands.

/dev/console	system console
/dev/fdX	floppy drive (X can be Ø, 1, 2, or 3)
/dev/hdX	hard disk (X can be Ø, 1, 2, or 3)
/dev/lp	line printer
/dev/mem	physical memory
/dev/null	null device (used to redirect unwanted output)
/dev/rXX	unbuffered interface to corresponding device name
/dev/root	root file structure

/dev/swap	swap area
/dev/ttyXX	terminals
/dev/tty	the terminal you're using (the system supplies a number)

You shouldn't rename any of these files, since the system relies on the names as they are. However, if a name change would make a file name easier for you to remember, you can use a command called ln (link) to link a name variant of your choice to the name TRS-XENIX knows. For instance, if you or your users would rather call "fdØ" "floppyØ", link the two names like this:

```
ln /dev/fdØ /dev/floppyØ <ENTER>
```

Then, when you can give commands containing the phrase "floppyØ", the system knows that you mean "fdØ".

Note: If you do accidentally destroy a special device file, you have to restore it from a backup done with the **dump** command, since tar can't handle special device files.

A.4 /etc

The /etc directory contains miscellaneous system data files, and also administrative and other system programs. Some of the files in /etc are listed here: unless you're absolutely certain about what you're doing, you shouldn't change any of them. The one exception is /etc/motd, the message-of-the-day file, which is meant to be edited regularly.

/etc/mtab	mounted device table
/etc/passwd	password file
/etc/mount	for mounting a new file system

/etc/mkfs	for creating a file system
/etc/init	first process after boot
/etc/rc	bootup shell script
/etc/motd	message of the day

A.5 /lib

Most of the files in this directory are libraries for the C compiler, a computer language encoder. If you're not using the C compiler, you won't need the files, although you may want to save them for possible future use.

A.6 /mnt

This is an empty directory onto which you can mount additional file systems you create. Leave the directory empty, and don't remove it.

A.7 /tmp

The /tmp directory contains temporary files, most of which are created by TRS-XENIX itself, and some of which are created by users. Since they're temporary, they're logical candidates for deletion when you're short of disk space, although you should check them individually to make sure they're expendable.

You can, if you want, place a command in the /etc/rc file to delete all temporary files automatically at each boot. To do so, see Appendix X for editing instructions, and follow the directions to add these lines to the /etc/rc file:

```
rm -fr /tmp
mkdir /tmp
chmod 777 /tmp
```

(Of course you should warn your users that their temporary files are automatically deleted at every boot.)

A.8 /usr

In addition to all the users' home directories, the /usr directory contains the following:

/usr/bin	more commands, generally those less frequently used or non-essential to TRS-XENIX system operation.
/usr/include	header files for compiling C programs. These can be deleted if you're not using the C compiler.
/usr/lib	more libraries and data files used by various commands; don't delete this.
/usr/spool	various "spoolers" which store files in directories (for instance, /usr/spool/lpd).
/usr/tmp	more temporary files, which are good candidates for deletion when you're short of disk space.

/usr/adm/messages — a record of console error messages. Typically, these are reports of both disk and user errors (like "out of disk space" messages).

Looking at the file can be helpful in determining whether you have a hardware problem; you can scan a week's worth of messages and see, for example, if one particular drive is generating an unusual number of errors. As you read the file, you may see error messages which didn't appear or your console screen, usually because they were minor and transient.

HINT: This file is apt to grow quickly and use a great deal of system space, so periodically check it, print it out, and delete the file. Save the printout as a record of errors the system has encountered.

APPENDIX B

INSTALLING TRS-XENIX APPLICATIONS AND USING save TO MAKE BACKUPS

Installing TRS-XENIX Applications

Applications users will naturally have to install their applications packages before they can use them. Installation is a straightforward process, and its steps are described below. Please read through the instructions before beginning, and, during installation, be sure to watch the screen and answer all the prompts you're given.

To begin installing the package:

1. Turn on all peripherals and then turn on the computer.

2. Log in as root (to install TRS-XENIX Applications, you must be multi-user logged in as root).

3. At the root prompt (#), type:

   install <ENTER>

   The screen will display a TRS8Ø Model-16 XENIX System, Application Installation Menu. It will also display two prompts: 1) to install or q) to quit.

4. Enter <1> to install your application.

5. At the prompt which says:

   Insert diskette in Drive Ø and press <ENTER>

   insert your TRS-XENIX Application diskette in Drive Ø and press <ENTER>.

The next screen shows the name of the application being installed, the version number, and the catalog number. At the bottom of the screen a welcoming message appears.

When TRS-XENIX is finished, the message:

> Installation complete - Remove the diskette, then
> press <ENTER>

appears. Pressing <ENTER> returns the menu to the screen. You're then prompted to press:

> 1) to install
>
> q) to quit.

Your installation is now complete, and you may now install another TRS-XENIX application or quit. (Pressing <q> returns you to TRS-XENIX.)

Using save to Make Backups

TRS-XENIX offers applications users a special program, called **save** to make applications backups. (see the introduction to Chapter 7 for information about the importance of backups.) Of course you can use TRS-XENIX's regular **sysadmin** program to backup your system -- but **sysadmin** copies everything on your disk, including built-in TRS-XENIX files and directories; **save** is more specific, copying only your applications material.

The following instructions tell you how to use **save**; use these steps instead of the ones in your applications manual. Note that first you must format some floppy disks; formatting instructions are in Chapter 7. After you've formatted, begin with **save** step 1.

1. Boot TRS-XENIX and log in as root, while you are at the console in multi-user mode. At the root prompt, type:

   save <ENTER>

2. The following menu is displayed:

   ```
   Save/Restore User File Systems
   ------------------------------

   1  to do full daily data save
   2  to do daily data save by system
   3  to restore full daily data save
   4  to restore daily data save by system
   5  to save programs
   6  to restore programs
   q  to quit
   ```

3. You're now prompted to enter a selection. A description of each selection follows:

   full daily data save (1) -- saves the data files for all TRS-XENIX® Applications on the hard disk.

   daily data save by system (2) -- saves the data files for one TRS-XENIX Application. You're prompted to enter the two-letter initial of the application you want backed up (for instance, Ar for Accounts Receivable, Gl for General Ledger, and so on.

   restore full daily data save (3) -- restores files saved by selection (1).

   restore daily data save by system (4) -- restores files saved by selection (2).

   save programs (5) -- Saves all TRS-XENIX Application programs.

   restore programs (6) -- restores all TRS-XENIX Application programs, stored with selection (5).

4. After you enter your selection, you're prompted to enter a <1> if you are using single-sided diskettes or a <2> if you are using double-sided diskettes.

5. You're then prompted:

   . Enter Drive Number:

   Enter a <Ø> or a <1> for Floppy Drive Ø or Floppy Drive 1, respectively.

6. Some selections prompt you for more information. You're then prompted to insert the first floppy disk into the drive. When the first floppy disk is full, you're prompted to enter the next disk, and so on.

7. You should label each floppy disk with a description, date and disk number in the backup series, like this:

   General Ledger Data Files ** BACKUP **
   January 23, 1983 Disk # 1 of 1Ø

8. When the process is complete, you are returned to the SAVE menu. If you're through, press <q> to exit to the root prompt.

HINT: You're wise to make applications data backups, using either option 1 or option 2, every day. Plan to make one set of applications program backups (option 5) immediately; then make additional program backups as needed -- for instance, if one of your programs is modified or updated by Radio Shack.

APPENDIX C

USING tsh -- THE TRSSHELL PROGRAM

The **trsshell** is a second shell -- command level -- for TRS-XENIX. **Trsshell** commands use command names familiar to TRSDOS users, and are designed to make the transition to using the TRS-XENIX system easier for them. (TRS-XENIX users who aren't familiar with TRSDOS needn't be concerned with **trsshell**, although of course the can use it if they wish.

After you've booted TRS-XENIX -- or in fact at any prompt -- you can invoke the **trsshell** by typing:

tsh <ENTER>

At this point, you can use any of the commands listed in this Appendix; of course, you can also keep using regular TRS-XENIX commands.

Before you log out of TRS-XENIX, you must first exit **trsshell**; do so by typing either:

exit <ENTER>

or

<CTRL><D>

and you're back in the regular TRS-XENIX shell. (Note that, if you want to log out of TRS-XENIX, you type <CTRL><D> again.

Note that if /etc/passwd specifies that your home shell is **trsshell** then you needn't exit **trsshell** before logging out. Simply type <CTRL><D> to log off the system.

NOTE: In any **tsh** command that allows multiple file names to be specified, you may use wild cards.

auto [command line]

The **auto** function creates or removes an automatic command that is executed when tsh is invoked. When an **auto** command line is entered, the command line is stored in the .tshrc file. This file is executed when trsshell starts up. If the file doesn't already exist, it is created.

To cancel the auto function, type **auto** <ENTER> and the automatic command line is removed from the .tshrc file.

backup

To backup a diskette, run the **diskutil** program at the XENIX Boot prompt (not from within XENIX). To run **diskutil**, type:

XENIX Boot
:diskutil <ENTER>

Further information on **diskutil** can be found in Chapter 8 of this guide.

chdir [directory]

Changes the **current working directory** to the specified directory. If no directory is specified, chdir finds your home directory and makes it the current working directory.

cls

Clears the screen and homes the cursor.

copy file1 file2
copy file1 [file2...] **directory**

Copies file1 to file2. The permissions and owner of file2 are kept if the file already exists; otherwise, the permissions of the source file are used.

Multiple files can be copied into a directory by using the second form of copy. The filenames are not changed.

Note: copy refuses to copy a file onto itself.

dir [-abcdgrstuR] [name...]

Lists in long format all files specified. If a directory is specified, the name is printed and its contents are listed.

The listing is sorted alphabetically by default. If no directory name is given, the current directory is listed. By specifying multiple directory names, you can see the contents of several directories. Filenames are listed first and followed by the directories when mixed names are listed.

The available options for **dir** are:

-a Lists files beginning with "." as well as all other files. (Super-user mode automatically uses -a.)

-b Prints invisible characters as /nnn (octal)

-c Uses file creation time for sorting or printing.

-d If a directory name is specified, lists only its name, not the contents of the directory.

-q Gives group ID instead of owner ID in listing.

-r Reverses the order of the sort to get reverse alphabetic or oldest date first, depending on the options specified.

-s Gives size in blocks, including indirect blocks, for each entry.

-t Sorts by time modified unless used with the -c or -u options.

-u Uses last access time instead of last modification for sorting or printing.

-R Lists the contents of all the subdirectories for the specified directory.

dismount [:] [drive-number]

Informs TRS-XENIX that a previously mounted file system on the specified drive is to be removed.

Execute the **dismount** command before removing the mounted disk.

For a full explanation of the mount process, see "mount".

display [-srw] [-n] [+linenumber] [name ...]

Displays a screenful of text for examination.

It normally pauses after each screenful and prompts "--More--". To see one more line of text, press <ENTER>. To see the next screen of text, press <SPACEBAR>.

The command line options are:

-n an integer number specifying the number of lines for the display window. If this option is omitted, the entire screen is used.

-s "squeezes" multiple blank lines from the output by printing only one blank line. This increases the amount of useful information displayed on the screen.

-r displays control characters as ^c where c represents the character. Otherwise, control characters that cannot be interpreted are not displayed.

-w when end of input is reached, "wait" for any key to be pressed before exiting. Otherwise, **display** exits immediately.

+linenumber display file, starting at linenumber.

Display looks in the file /etc/termcap to determine terminal characteristics, and to determine the default window size. On a terminal capable of displaying 24 lines, the default window size is 22 lines.

If **display** is reading from a file, rather than a pipe, then a percentage is displayed along with the "--More--" prompt. This gives the fraction of the file (in characters, not lines) that has been read so far.

Other sequences which may be typed when **display** pauses, and their effects are as follows

Note: n is an optional. If omitted, one (1) is used.

n<SPACEBAR> displays n more lines. If n is omitted, the next screenful is displayed.

<CTRL><D> displays 11 more lines ("scroll"). If n is given, then the scroll size is set to n.

d same as ^D (control D)

nz same as typing a space except that n, if specified, becomes the new window size.

ns skips n lines, then prints the next screenful of text.

nf skips n screenfuls, then prints the next screenful of lines

q exits **display** (same as Q)

= displays the current line number.

v starts up the editor vi at the current line.

h help command; gives a description of all the **display** commands (same as ?).

!command invokes a shell with command. The characters '%' and '!' in "command" are replaced with current file name and the previous shell command respectively. If there is no current file name, '%' is not expanded. The sequences "\!" are replaced by "%" and "!" respectively.

n:n skips to the n-th next file given in the command line (skips to last file if n doesn't make sense).

n:p skips to the n-th previous file given in the command line. If this command is given in the middle of printing out a file, then **display** goes back to the beginning of the file. If n doesn't make sense, **display** skips back to the first file. If **display** is not reading from a file, nothing happens.

:f displays the current file name and line number.

:q or :Q exits from **display** (same as q or Q).

. (dot) repeats the previous command.

The commands take effect immediately; i.e., it's not necessary to type a carriage return. Up to the time when the command character itself is given, you can press the line kill character (<CTRL><U>) to cancel the numerical argument being formed. In addition, you may press the erase character (<BACKSPACE>) to redisplay the "--More--(xx%)" message.

At any time when output is being sent to the terminal, you may press the quit key (normally <CTRL><7>). **display** will stop sending output, and will display the usual "--More--" prompt.

You may then enter one of the above commands in the normal manner. Unfortunately, some output is lost when this is done, due to the fact that any characters waiting in the terminal's output queue are flushed when the quit signal occurs.

The terminal is set to **noecho** mode by this program so that the output can be continuous. What you type is not displayed on your terminal, except for the slash (/) and exclamation (!) commands.

do *filename*

The list of commands stored in *filename* is executed as keyboard entries. Nesting of *do* commands is allowed as deep as is necessary or as the system has memory.

files [-abcdqrstux1CR] [*name*...]

Lists in columns all files in the specified directory. The listing is sorted alphabetically by default. If no name is given, the current directory is listed. By specifying multiple directory names, you can see the contents of several directories. File names are listed first, and then the directories when mixed names are specified.

Also, all directories are marked with a trailing "/" and executable files are marked with a trailing "*".

The available options for **files** are:

-a Lists files beginning with "." as well as all other files. (Super-user mode defaults to -a.)

-b Prints invisible characters as /nnn (octal)

-c Uses file creation time for sorting and printing.

-d If a directory name is specified, lists only its name, not the contents of the directory.

-q Gives group ID instead of owner ID in listing.

-r Reverses the order of the sort to get reverse alphabetic or oldest date first, depending on the options specified.

-s Give size in blocks, including indirect blocks, for each entry.

-t Sorts by time modified unless used with the -c or -u options.

-u Uses last access time instead of last modification for sorting and printing.

-R Lists the contents of all the subdirectories for the specified directory.

-x Forces columnar printing. (Sorted material is printed across rather than down the page.)

-1 Forces a one entry per line output format, e.g. to a teletype.

-C Forces multi-column output; this is used with output redirection.

free [filesystem...]

Prints out the number of free blocks available on the specified filesystem. If no file system is specified, the free space on all of the mounted file systems are printed.

help [subject]

Displays the help text for the specified subject. A few special help commands are:

help *	displays a list of subjects for which help is available.
help	displays an explanation of the help text and the notations used within.

If the help text is longer than one screen, the message --More--(nn%) is displayed. Press <SPACEBAR> to see the next screenful, or press <ENTER> to expose one more line. To abort the display of help text, press <q>.

i

To inform the system you have placed a diskette in a drive or wish to remove one, you need to use the commands mount and dismount. If you have placed a XENIX disk in drive Ø, for example, you may type "mount Ø" to tell the system to make its contents available from the directory "/mntØ". Similarly, when you are finished and wish to remove it, you type "dismount Ø" and then the system unmounts your floppy.

kill [-r] file...

-r Deletes the entire contents of the specified directory, and the directory itself.

Removes the entries for one or more files from a directory, thus destroying the file. A directory must have write permission if files are to be deleted from it. The file however, does not need to have read or write permission to be deleted.

You are prompted for each file before it is removed. Answer <y> <ENTER> to delete the file.

If a designated file is a directory, an error comment is printed unless the optional argument -r has been used.

You cannot remove the file ".." from a directory.

lib

Displays a list of the trsshell commands.

mount drive-number

Announces to the system that a file system is now present on Drive drive-number. The file system's contents are made available in the directory "/mntn", where n is the specified drive. For example, if "mount Ø" is typed, then the contents of the floppy on drive Ø are made available in the directory "/mntØ".

Dismount performs the inverse operation, announcing to the system that the file system previously mounted on Drive drive-number is to be removed.

Mount refuses to mount a file system which is not marked clean: this can happen if a system crash prevented the use of dismount or /etc/shutdown. In such a case, use **fsck** to clean the file system, then try **mount** again. (See Chapter 7 of this manual for more information.)

These commands keep track of where each device is mounted. If no drive-number is given, mount prints a table showing the currently mounted filesystems (if any).

Note: You cannot mount a disk with an alien file system on it, for example, one with TRSDOS on it. If you attempt to do so, you will crash the system. Likewise, you cannot mount a floppy or hard disk which has been newly formatted. You must use the **mkfs** command on it first.

move [-i] file1 file2
move [-i] file1 [file2...] **directory**

-i prompts if the destination file already exists. You then choose to keep or overwrite the existing file.

Copies file1 to file2. The permissions and owner of file2 are kept if the file already exists; otherwise, the permissions of the source file are used.

Multiple files can be copied into a directory by using the second form of move. The filenames are not changed.

Note: move refuses to copy a file onto itself.

print [-r|-c] [-m|-n] [file...]

Causes the file(s) to be queued for printing on the line printer. If no files are named, the keyboard is read until you type a ^D at the beginning of a line.

The following options are available:

-r Removes the file when it has been queued.

-c Copies the file to avoid any changes being made before it's printed.

-m Reports by mail when printing is complete.

-n Does not report by mail. This is the default option.

rename old.file new.filename

Changes old.file's name to new.filename.

restore [:] drive-number [-d] | [name...]

-d displays a list of files on the floppy diskette (directory).

Reads in files that have been written to a floppy with save and restores them to where they were saved from. You may use multiple filenames or directory names. If a directory name is specified, it restores the directory and its files and subdirectories (and their contents).

save [:] drive-number {-ss|-ds} [name...]

-ss specifies that the destination floppy is a single-sided disk.
-ds specifies that the destination floppy is a double-sided disk.

One of the above (-ss or -ds) must be specified.

Saves files or entire directories and their contents to the floppy in the specified drive. You may use multiple file names or directory names. If a directory name is specified, it saves the directory and its files and subdirectories (and their contents).

You must tell save whether you are using single-sided diskettes or double-sided by using the -ss flag if they are single sided, and by using the -ds flag if they are double-sided. The floppy disks must be formatted before use.

set [option]

Allows you to change tsh's internal options. To display the options settings, type **set** <ENTER>.

Options that are currently available are:

verbose [on|off] -- if verbose is off, trsshell does not inform you of a command's success or failure.

retry [on|off] -- enables/disables the retry feature. When retry is on, the command line is converted to lower-case if trsshell was unable to execute a command. Trsshell then tries to execute the lower-case command line.

prompt string -- sets the trsshell prompt to string. For example: set prompt cue> <ENTER>, sets the string "cue>" as the new trsshell prompt.

showdir

Prints the pathname of the current working directory.

time

Displays the current date and time.

version

Displays the name of the shell and its current version.

Appendix D

DEFAULTS OF THE TRS-XENIX SYSTEM

This appendix summarizes TRS-XENIX's defaults -- automatic settings by, or entries in, the basic system. It's ordinarily appropriate to leave them as they are; occasionally, however, you may want to change a default setting, and, where it's applicable, information on how to make those changes is included here.

D.1 The Root Password -- The initial setting of the root password.

Default: <ENTER>
Files Affected: /etc/passwd
To Change: use **passwd** command (see Chapter 4)

D.2 Adding Users with the mkuser command -- default settings used by mkuser.

D.2.1 Group and User Numbers

Default: user# = sequentially assigned numbers above 200
group# = 50
Files Affected: /etc/passwd
To Change: use **chown** or **chgrp** to alter owner number or group number

D.2.2 Standard .profile File

Default: creates standard .profile for each user added with **mkuser**.
Files Affected: /usr/lib/mkuser.prof
To Change: edit the /usr/lib/mkuser.prof file

D.3 Drive Defaults

D.3.1 Number of Blocks and Kbytes Per Disk

Type of Disk	Blocks	Kbytes
single-sided floppy disk	1216	608 Kbytes
double-sided floppy disk	2448	1224 Kbytes
Eight Meg Hard Disk		
Primary	14,909	7454.5 Kbytes
Swap Area	2057	1028.5 Kbytes
Secondary	16,966	8483 Kbytes
Twelve Meg Hard Disk		
Primary	20,961	10480.5 Kbytes
Swap Area	2057	1028.5 Kbytes
Secondary	23,018	11,509 Kbytes

D.3.2 **Mounted Drives** -- tells TRS-XENIX which drive devices to mount to the root file system.

Default: Only the primary hard disk is mounted (/dev/hd0)
Files Affected: /etc/mtab
To Change: Edit /etc/rc file to include a mount statement.

D.3.3 **Write Verifies** -- tells TRS-XENIX whether or not to verify all writes.

Default: Verify is turned off.
Files Affected: none
To Change: Type **verify y** to turn on verify.

D.3.4 **drive Command** -- tells TRS-XENIX what type of floppy disk drives are on the system.

Default: Drive Ø: 1Øms,detect,wait
Drive 1-3: 15ms,nodetect,wait
Files Affected: /xenix
To Change: issue a drive command using the following syntax:

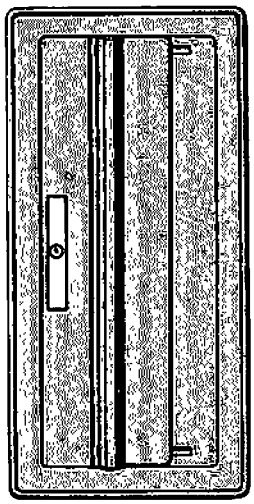
drive [#] [options]

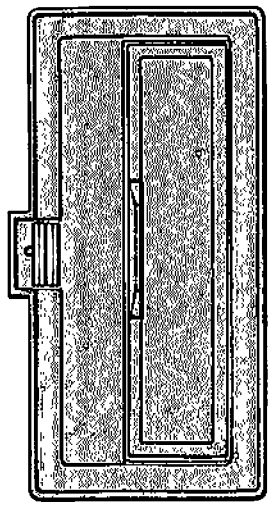
options available are:

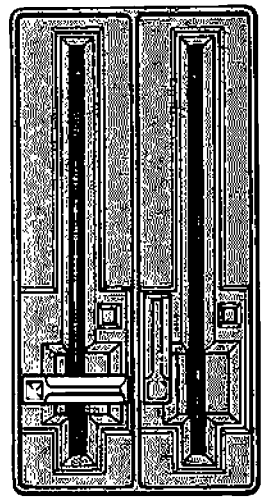
rate=n - sets the seek rate.
n can be Ø,1,2 or 3 for 3ms,
6ms, 1Øms or 15ms respectively.
detect|nodetect - sets the
door open detection on or off.
wait|nowait - sets "wait for
drive motor to reach proper speed"
on or off.

Suggested settings:
Thinline = Ø,detect,wait*
Latch = 3*,nodetect*,nowait
Push-Button = 2,detect,nowait
* indicates a required setting

This procedure changes the settings for the next time you boot up.


Push-Button


Latch


Thinline

D.4 Terminals

D.4.1 **Terminal Settings** -- tells TRS-XENIX what type of terminal is expected.

Default: Console = "trs16" terminal mode
Both terminals (tty01,tty02) are set to Adds25 emulation mode of a DT-1 terminal.
Files Affected: /etc/ttytype
/etc/termcap
To Change: Edit the /etc/ttytype file.
Terminal descriptor must be listed in the /etc/termcap file.

D.4.2 **Terminal Status** -- tells TRS-XENIX at which terminals user can log in.

Default: console = enabled
tty01 = disabled (Serial Channel A)
tty02 = disabled (Serial Channel B)
Files Affected: /etc/ttys
To Change: Use the **enable** or **disable** command. (See Chapter 11)

D.4.3 **Key Values** -- sets the values for the erase (destructive backspace) key and the line kill key

Default: erase = ^H (<BACKSPACE>)
kill=^U (<CTRL><U>)
Files Affected: .profile
To Change: Use the **stty** command:
stty erase ^X to set erase
stty kill ^X to set kill
stty <ENTER> to see values
Or edit .profile or
/etc/lib/mkuser.prof file.

D.5 Miscellaneous Defaults

Defaults: dump and dumpdir are used by the **sysadmin** program
lpd specifies how many banner pages when a file is printed by **lpr**
mkuser sets the default for home directory and shell that are used by the **mkuser** command
Files Affected: /etc/default/dump
/etc/default/dumpdir
/etc/default/lpd
/etc/default/mkuser
To Change: edit the appropriate file.

Appendix E

TRANSFERRING TRSDOS-II FILES TO TRS-XENIX

The **tx** command is used to transfer files from TRSDOS-II formatted diskettes to TRS-XENIX. It also transfers files from TRSDOS-II SAVE formatted diskettes (-r option).

When the files are transferred to TRS-XENIX, their filenames are changed to meet XENIX file specifications. That is, all upper-case letters are changed to lower-case, and the slash (/) preceding the extension is changed to a period (.).

The **tx** command also allows you to display directory information for TRSDOS-II diskettes.

Notes:

1. Password protected files are transferred to TRS-XENIX. Ownership is given to the user only.

2. The /dev files for the floppy drives must have the "r" permission set (/dev/rfd*n* and /dev/fdbt*n* -- *n* is Ø,1,2 or 3 depending on the drive number).

Important: TRS-XENIX can only read TRSDOS-II 4.x media. If you wish to transfer TRSDOS files to TRS-XENIX, use the FCOPY command to copy the files to TRSDOS-II and then use the **tx** command to transfer the files to TRS-XENIX.

The syntax for tx is:

a) tx :d options Tfspec|Twc Xdir

b) tx :d options Tfspec Xfspec

c) tx :d -f [Tfspec|Twc]

options can be any of the following:

-a absolute; copy files even if file already exists
-c do not convert upper-case letters in the filename to lower-case
-f display the directory for the specified drive
-l do not add a line feed to TRSDOS-II VLR files
-p prompt operator before each file is moved
-r specifies that the source is TRSDOS-II SAVE media
-s do not strip length byte from TRSDOS-II VLR files
-v verbose; display informative messages during copy.
-x converts carriage returns to linefeeds (ØAH to ØDH)

Where:
Tfspec = TRSDOS-II file specification
Twc = TRSDOS-II wildcard specification
Xfspec = TRS-XENIX file specification
Xdir = TRS-XENIX directory
:d = specifies the drive that contains the TRSDOS-II diskette. :d is required

If Xdir is omitted, the default is the current working directory (form a of syntax).

The options and drive number (:d) are position-independent. They may be located in any order on the command line (except within filespecs).

If a wild card mask is used, it must be enclosed in quotes.

Examples

1. tx :Ø

   Copies all files from the TRSDOS-II diskette in Drive Ø to the current working directory of TRS-XENIX (default).

2. tx :Ø ARMAIN/DAT .
   tx ARMAIN/DAT . :Ø

   Copy file ARMAIN/DAT from the TRSDOS-II diskette in Drive Ø to the current working directory (.). The TRS-XENIX filespec is armain.dat.

3. tx :1 LEAOWN/IDX LEAOWN/DAT OWNER/DAT /usr/lease/owners -x
   tx -x LEAOWN/IDX LEAOWN/DAT OWNER/DAT /usr/lease/owners :1
   tx LEAOWN/IDX :1 -x LEAOWN/DAT OWNER/DAT /usr/lease/owners

   Copies files: LEAOWN/IDX, LEAOWN/DAT and OWNER/DAT from the TRSDOS-II diskette in Drive 1 to the TRS-XENIX directory /usr/lease/owners. The TRS-XENIX filenames are: leaown.idx, leaown.dat, and owner.dat.

   Any carriage returns in the files are converted to line feed. (This is useful when transferring BASIC source to TRS-XENIX.)

4. tx :1 -c LEAOWN/IDX LEAOWN/DAT OWNER/DAT /usr/lease/owners
   tx LEAOWN/IDX LEAOWN/DAT OWNER/DAT -c /usr/lease/owners :1
   tx LEAOWN/IDX -c :1 LEAOWN/DAT OWNER/DAT /usr/lease/owners

   Copies files: LEAOWN/IDX, LEAOWN/DAT and OWNER/DAT from the TRSDOS-II diskette in Drive 1 to the TRS-XENIX directory /usr/lease/owners. Since the -c switch was specified, the filenames are not converted to lowercase:

TRSDOS-II	TRS-XENIX
LEAOWN/IDX	LEAOWN.IDX
LEAOWN/DAT	LEAOWN.DAT
OWNER/DAT	OWNER.DAT

5. tx :3 "AR*" /usr/AR -pav
   tx -p -a -v :3 "AR*" /usr/AR
   tx "AR*" /usr/AR :3 -apv

   Displays all files found on the TRSDOS-II diskette in Drive 3 which match the wild card mask **AR***. The -p option causes the message **copy ?** to be displayed. Answer <y>es or <n>o. If <y> is entered, the TRSDOS-II file is copied into the TRS-XENIX directory /usr/AR.

   Because the absolute switch was specified (-a), you're not prompted if the destination file already exists. However, informative messages are displayed because the -v switch was specified.

6. tx :Ø -f

   Displays directory information for every file on the TRSDOS-II diskette in Drive Ø.

7. tx :1 -f -r

   Displays directory information for every file on the TRSDOS-II SAVE diskette in Drive Ø.

8. tx :3 INIT/* -f

   Displays directory information for all files matching the wild card mask INIT/* found on the TRSDOS-II diskette in Drive 3.

9. Wild card Example:

The wild card character "*" matches zero or more occurrences of any legal filespec character including the slash.

The wild card character "?" matches only one occurrence of any legal filespec character including slash.

Suppose you have a diskette with the following files in the directory:

TEST	TEST1	TEST/DAT
TESTA/DAT	TEST1/DAT	TEST/C

The wild card mask:

"TEST*"

matches all of the above files. It's important to remember that TRS-XENIX does not follow TRSDOS-II conventions for wildcarding. TRS-XENIX does not recognize extensions; therefore, a TRSDOS-II extension is considered to be part of the filename. In this example "TEST*" matches any filename that begins with the letters "TEST",

However, the wild card mask:

"TEST?"

matches only the file TEST1 because the "?" character means to match only one letter. Therefore filespecs with extensions are too long to match this wildcard.

The wild card mask:

"TEST?*"

matches the following files:

TEST1	TEST/DAT	TESTA/DAT
TEST1/DAT	TEST/C	

The file TEST did not match because the "?" character must match at least one character.

Note: Any filespec containing the wildcard mask characters (* or ?) must be enclosed in quotes.

ERRORS

"Unknown option found" An illegal option was used.

"Multiple drive specification" More than one device type was specified.

"Illegal drive unit number" An illegal drive number was given. Drive numbers may be in the range [Ø,7].

"Drive number not found on the command line" A source drive number is required in the command line.

"Directory Xdir does not exist" The directory name specified was not found on the TRS-XENIX file system.

"Filestat failed on Tfspec" Internal Error. The specified file was opened successfully, but the file status could not be read.

"Filename Xdir is not a directory" The TRS-XENIX directory specified is not a directory, but is a file.

Appendix W: Basic Concepts

This appendix gives you an understanding of the basic concepts that you need to function in the XENIX environment. It discusses the XENIX file system, naming conventions, commands, and input and output. After reading this appendix you will have an understanding of how the system's files, directories, and devices are organized and named, how commands are entered, and how a command's input and output can be manipulated.

W.1 File System

A file system is a set of files organized in a logical fashion. In XENIX, this set of files consists of all available system resources including data files, programs, line printers, and disks. Thus, the XENIX file system is more generally a system for accessing all of the resources of the system.

To logically structure the files and resources of the system, the XENIX file system is organized hierarchically into a "tree-structure." See Figure W-1 for an illustration of a typical tree-structured file system. In this tree of files, the root of the tree is at the top and branches of the tree grow downward. Directories correspond to nodes in the tree; ordinary files correspond to "leaves." If a directory contains a downward branch to other files or directories, then those files and directories are "contained" in the given directory. It is possible to specify any file in the system by starting at the root (where the root is at the top) and traveling down any of the branches to the desired file. Similarly, you can specify any file in the system, relative to any other. Specification of these files depends on a knowledge of XENIX naming conventions, discussed in Section W.3.

Privacy and security for files and directories can be arranged. Each file and directory has read-write-execute permissions that can be set to control access by the owner, by a group of users, and by everyone else.

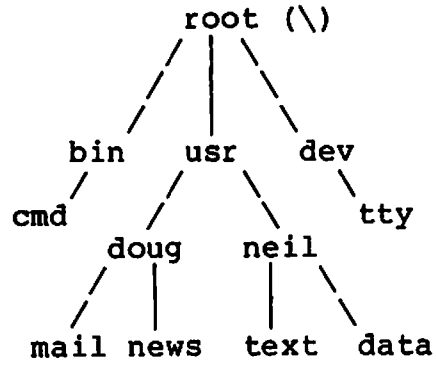


Figure W-1. A Typical File System

In the typical tree-structured file system of Figure W-1, the "tree" grows downward. The names root, bin, usr, dev, doug, and neil all represent directories, and are all nodes in the tree. By convention, in XENIX the name of the root directory is given the one character name, "/". The names mail, news, text, and data all represent normal data files, and are all "leaves" of the tree. Note that cmd, which here is an executable command, is also a leaf and therefore a file. The name tty represents a terminal and is also represented in the tree.

W.1.1 Files

The file is the fundamental unit of the file system. Conceptually, everything is treated as a file. However, there are really three different type of files: ordinary files (what we usually mean when we say "file"), special files, and directories. Each of these file types is discussed below:

Ordinary Files

Ordinary files typically contain textual information such as documents, data, or program sources. Executable binary files are also of this type. Any file that is not a special file or a directory is an ordinary file.

Special Files

A special file is one that corresponds to a physical device of some sort, such as a disk, a line printer, a terminal, or system memory. To the XENIX user, special files can be treated like ordinary files. However, the internal handling of a printer or terminal is much different from that of an ordinary disk file, and the operations that can be performed on devices vary from those that

can be performed on ordinary files.

Directories

Directories are read-only files containing information about the files or directories that are conceptually (but not physically) contained within them. The nesting of directories in other directories is the way in which XENIX implements its characteristic tree-structured directory system. Directories are discussed further in the next section.

W.1.2 Directory Structure

With multiple users and multiple projects, the number of files in a file system can proliferate rapidly. Fortunately, as explained earlier, XENIX organizes all files into a tree-structured directory hierarchy. Each user of the system has his own personal directory. Within that directory, the user may have directories or other subdirectories owned and controlled only by the user.

When you log in to XENIX, you are "in" your directory. Unless you take special action when you create a file, the new file is created in your working directory. This file is unrelated to any other file of the same name in someone else's directory.

A diagram of part of a typical user directory is shown in Figure W-2.

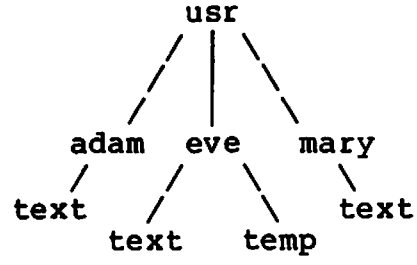


Figure W-2. A Typical User Directory

In Figure W-2, the usr directory contains each user's own personal directory. Notice that Mary's file named text is unrelated to Eve's. This is not important if all the files of interest are in Eve's directory, but if Eve and Mary work together, or if they work on separate but related projects, this division of files becomes handy indeed. For example, Mary could print Eve's text by typing:

```
pr /usr/eve/text
```

Similarly, Eve could find out what files Mary has by typing:

```
ls /usr/mary/*
```

W.2 Naming Conventions

Now that we have discussed what the XENIX file system is, and what it consists of, we need to use filenames in a more precise way. The first thing to remember is that every single file, directory, and device in XENIX has both a filename and an absolute pathname. The absolute pathname is unique to all names in the system; filenames are unique only within directories and need not be unique system-wide. This is similar to someone who's "absolute" name is John Robert Smith, but whom everyone calls John. The name John need not be unique, although it will greatly simplify life if John Robert Smith is a unique name.

W.2.1 Filenames

A simple filename is a sequence of 1-14 characters other than a slash (/). Every single file, directory, and device in the system has a filename. Filenames are used to uniquely identify directory contents. Thus, no two names in a directory may be the same. However, filenames in different directories may be identical.

W.2.2 Pathnames

A pathname is a sequence of directory names followed by a simple filename, each separated from the previous one by a slash. If a pathname begins with a slash, the search for the file begins at the root of the entire tree. Otherwise, it begins at the user's current directory (also known as the working directory). A pathname beginning with a slash is called a full (or absolute) pathname because it does not vary with regard to the user's current directory. A pathname not beginning with a slash is often called a relative pathname, because it specifies a path relative to the current directory. The user may change the current directory at any time by using the cd command.

In most cases, a filename and its corresponding pathname may be used interchangeably.

W.2.3 Sample Names

Some sample names follow:

/	The absolute pathname of the root directory of the entire file system.
/bin	The directory containing most of the frequently used XENIX commands.
/usr	The directory containing each user's personal directory. The subdirectory, /usr/bin contains frequently used XENIX commands not in /bin.
/dev	The directory containing files corresponding to each available physical device (e.g., terminals, line printers, and disks).
/lib	The directory containing special data files used by some standard commands.
/tmp	This directory contains temporary scratch files.
/usr/joe/project/A	This is a typical full pathname. This one happens to be a file named A in the directory named project belonging to the user named joe.
bin/x	A relative pathname; it names the file x in subdirectory bin of the current working directory. If the current directory is /, it names /bin/x. If the current directory is /usr/joe, it names /usr/joe/bin/x.
file1	Name of an ordinary file in the current directory.

Each user resides "in" a directory called the current directory. All files and directories have a "parent" directory. This directory is the one immediately above and "containing" the given file or directory. The XENIX file system provides special shorthand notations for this directory and for the current directory:

- The shorthand name of the current directory. Thus ./filexxx names the same file as filexxx, if such a file exists in the current directory.

.. The shorthand name of the current directory's parent directory. If you type

```
cd ..
```

then the parent directory of your current working directory becomes your new current directory.

Although you can use almost any character in a filename, common sense says you should stick to ones that are visible, and that you should probably avoid characters that might be used with other meanings. For instance, the dash (-) is used in specifying command options, and should be avoided when naming files. To avoid pitfalls, you will do well to use only letters, numbers, and the period.

W.2.4 Special Characters

XENIX provides a pattern-matching facility for specifying sets of filenames that match particular patterns. For example, examine the problem that occurs when naming the parts of a large document, such as a book. Logically, it can be divided into many small pieces: chapters or perhaps sections. Physically, it must be divided too, since the XENIX editor, ed, cannot handle really big files. Thus, you should construct a document as a number of files. For example, you might have a separate file for each chapter:

```
chap1
chap2
...
```

Or, if each chapter were broken into several files, you might have:

```
chap1.1
chap1.2
chap1.3
...
chap2.1
chap2.2
...
```

You could then tell at a glance where a particular file fits into the whole.

There are other advantages to a systematic naming convention that are not so obvious. What if you wanted to print the whole book? You could type

```
pr chap1.1 chap1.2 chap1.3 ...
```

but you would get tired pretty fast, and would probably even make mistakes. Fortunately, there is a shortcut: a sequence of similar names can be specified with the use of two special "wild card" characters.

For example, you can type:

```
pr chap*
```

The asterisk (*), called "star" in XENIX, means "zero or more characters of any type," so this translates into "print all files whose names begin with the word "chap", listed in alphabetical order."

This shorthand notation is not a unique property of the pr command; it is a system-wide service of the shell program that interprets commands, sh. Using this fact, you list the names of the files in the book by typing:

```
ls chap*
```

This produces

```
chap1.1
chap1.2
chap1.3
...
```

The star is not limited to the last position in a filename; it can be used anywhere and can occur several times. As a special case, a star by itself matches every filename, so

```
rm *
```

removes all your files.

The star is not the only pattern-matching feature available. Suppose you want to print only chapters 1 through 4, and 9. Then you can say

```
pr chap[12349]*
```

The brackets ([and]) mean "match any of the characters inside the brackets." A range of consecutive letters or digits can be abbreviated, so you can also do this with

```
pr chap[1-49]*
```

(Note that this does not match forty-nine filenames, but

only five.) Letters can also be used within brackets: "[a-z]" matches any character in the range "a" through "z".

The question mark (?) matches any single character, so

```
ls ?
```

lists all files that have single-character names, and

```
ls -l chap?.1
```

lists information about the first file of each chapter (i.e., chap1.1, chap2.1, ...).

Of these pattern matching conventions, the star (*) is certainly the most useful, and you should get used to it at once.

If you should ever need to turn off the special meaning of any of the special characters (*, ?, and [...]) enclose the entire argument in single quotes. For example, the following command will print out only files named "?" rather than all one character filenames:

```
ls '?'
```

W.3 Commands

Commands are used to invoke executable programs. When you type the name of a command, the XENIX shell looks for a program with the given name to execute. If the shell finds an executable program, it will execute it. Commands always contain the name of the executable program as the first word of the command. Commands may also contain switches that specify options or other arguments as needed by the program. Commands are entered on a command line that is read by the shell. Command lines are discussed in the following subsection.

W.3.1 Command Line

Whether typing at the terminal, or executing commands from a file, XENIX always reads commands from command lines. The command line is a line of characters that is scanned and read by the shell command interpreter to determine what to do next. When you are typing at a terminal, you are editing a line of text called the command-line buffer that becomes a command line only when you type <RETURN>. This command-line buffer can be edited with the <BKSP> and <CONTROL-U> keys. Typing <RETURN> causes the command-line buffer to be submitted to the shell as a command line. The shell reads

the command line and executes the appropriate command. If you type <INTERRUPT> before you type <RETURN>, then the command-line buffer is aborted. Multiple commands can be entered on a single command line so long as they are separated by a semicolon (;). For example, the following prints out the current date and the name of the current working directory:

```
date ; pwd
```

Commands can be submitted for processing in the background by appending an ampersand to the command. Thus

```
mv file1 file2 file3 file4 dir1&
```

will move the files file1, file2, file3, and file4 to the directory dir1 without tying up your terminal. You can execute other commands from your terminal in the foreground while the mv command executes in the background.

W.3.2 Syntax

The general syntax for commands is as follows:

```
cmd  [ switches ][ arguments ] [ filenames ]
```

In practically all cases, command names are all lowercase. Switches are flags that select various options available when executing the command. Switches are optional and always precede other arguments and filenames. Switches consist of a dash prefix (-) and an identifying alphanumeric character. Some switches are also prefixed by a plus sign (+). Switches can often be grouped as a single switch as in:

```
ls -arl
```

Here the -a switch (pronounced "minus a") selects the option which lists all files in the directory. The -r switch selects the option which causes the names in the directory to be sorted in reverse alphabetical order. And the -l switch selects the option which causes listing of a long format for each directory entry.

Sometimes switches must be given separately, as in:

```
copy -v -a source dest
```

Here the -v switch specifies a verbose option. The -a switch tells the copy command to ask the user before copying the two given directories.

Arguments of various types can also be given, such as search strings, as in:

```
grep 'string of text' outfile
```

In the above example, "string of text" is an argument and is the search string that the grep command searches for in the file outfile. Outfile, itself, is a filename argument that specifies the name of a file required by the command.

In most cases, commands are executable object files compiled from C programs. In some cases, commands are executable command files called "shell procedures."

W.4 Input and Output

XENIX handles input and output from commands in a unique way: it assumes that input and output, by default, are associated with the terminal from which the command originates. That is, input comes from the keyboard and output goes to the terminal screen. To illustrate typical command input and output, type:

```
cat
```

This command now expects input from your keyboard. It will accept as many lines of text as you can type as input, until you type a <CONTROL-D> as an end-of-file indicator. For example, type:

```
this is two lines
of input
<CONTROL-D>
```

When you type the <CONTROL-D>, input ends and output begins. The cat command then immediately outputs the two lines that you typed -- since output is sent to the terminal screen by default, that is where the two lines are sent. Thus, the complete session will look like this on your terminal screen:

```
$ cat
this is two lines
of input
this is two lines
of input
$
```

The flow of command input and output can be "redirected" so that input comes from a file instead of from the terminal keyboard, and so that output goes to a file or to a line

printer, instead of to the terminal screen. In addition, "pipes" can be created that allow the output from one command to become the input to another. Redirection and pipes are the subjects of the next two subsections.

W.4.1 Redirection

It is universal in XENIX systems that a file can replace the terminal for either input or output. For example

```
ls
```

displays a list of files on your terminal screen. But if you say

```
ls >filelist
```

a list of your files is placed in the file filelist (which is created if it does not exist). The output redirection symbol (>) means "put the output from the command into the following file, rather than display it on the terminal screen." As another example, you could combine several files into one by capturing the output of cat in a file:

```
cat f1 f2 f3 >temp
```

The output append symbol (>>) operates very much like the output redirection symbol (>) does, except that it means "add to the end of." That is

```
cat file1 file2 file3 >>temp
```

means to concatenate file1, file2, and file3 to the end of whatever is already in temp, instead of overwriting and destroying the existing contents. As with normal output redirection, if temp doesn't exist, it is created for you.

In a similar way, the input redirection symbol (<) means to take the input for a program from the following file, instead of from the terminal. Thus, you could make up a script of editing commands and put them into a file called script. Then you could execute the commands in the script on a file using the XENIX editor by typing:

```
ed file <script
```

As another example, you could use ed to prepare a letter in file letter.txt, then send it to several people with

```
mail adam eve mary joe <letter.txt
```

W.4.2 Pipes

One of the major innovations of the XENIX system is the concept of a pipe. A pipe is simply a way to connect the output of one command to the input of another command, so that the two run as a sequence of commands called a pipeline.

For example

```
pr frank.id george.id hank.id
```

prints the files named frank.id, george.id, and hank.id, beginning each on a new page. Suppose you want them run together instead. You could type:

```
cat frank.id george.id hank.id >temp
pr <temp
rm temp
```

But this is more work than is necessary. What we want is to take the output of cat and connect it to the input of pr. So we use a pipe:

```
cat frank.id george.id hank.id | pr
```

The vertical bar (|) means to take the output from cat, which would normally have gone to the terminal, and put it into pr to be neatly formatted.

There are many other examples of pipes. For example,

```
ls | pr -3
```

prints a list of your files in three columns. The program wc counts the number of lines, words, and characters in its input, and who prints a list of currently logged on people, one per line. Thus,

```
who | wc
```

tells how many people are logged on. And of course

```
ls | wc
```

counts your files.

Any program that reads from the terminal keyboard can read from a pipe instead. Any program that displays output to the terminal screen can send input to a pipe. You can have as many elements in a pipeline as you wish.

Many XENIX programs are written so that they take their input from one or more files, if file arguments are given. If no arguments are given, they read from the terminal keyboard, and thus can be used in pipelines. For example

```
pr -3 albert.txt bernard.txt carl.txt
```

prints, in order, the files albert.txt, bernard.txt, and carl.txt. But in

```
cat albert.txt bernard.txt carl.txt | pr
```

pr prints the concatenation of these files coming down the pipeline. The difference is that here, albert.txt, bernard.txt, and carl.txt are run together and then treated as one file rather than three.

Appendix X: Frequently Used Commands

X.1 The Model-16 Console Keyboard

In the description of the commands in this appendix, note the mapping of Model-16 console keys to important XENIX characters not normally available on the standard Model-16 console keyboard:

XENIX Character	Model-16 Console
<INTERRUPT>	<BREAK>
<INTERRUPT>	<CONTROL-C>
<CONTROL-S>	<HOLD>
\ (backslash)	<CONTROL-/>
` (back quote)	<CONTROL-'>
| (vertical bar)	<CONTROL-!>
	<CONTROL-1>

X.2 Gaining Access to the System

Logging In

To gain access to the system, respond to the "login:" prompt by typing your user name followed by a <RETURN>. Then respond to the password: prompt with your password. For example, a login for the user joe might look like this:

```
login:joe
password:abracadabra
```

Note that the password is not shown on the terminal screen.

Logging Out

The logout procedure is simple -- all you need to do is type:

```
<CONTROL-D>
```

Since within other programs, <CONTROL-D> signifies the end-of-file, at times it may be necessary to type <CONTROL-D> several times before you can log yourself out.

Changing Your Password

To change your password, use the **passwd** command. For the user joe, a session might go like this:

```
Changing password for joe
Old password: palooka
New password: Bazookah
Retype new password: Bazookah
```

To maintain security, none of your responses are shown on the screen. It is best to mix upper- and lowercase letters and to pick a password greater than five characters in length. These measures should be taken to foil automated attempts at guessing your password.

X.3 Terminal Configuration

Setting Terminal Options

There are a number of terminal options that can be set with the command **stty**. When entered without parameters, **stty** displays the current terminal settings. For example, typical output might look like this:

```
speed 9600 baud
erase '^h' ; kill '^u'
even -nl
```

This says that the rate of data transmission to and from the terminal is 9600 baud, that the backspace character (erase) is <CONTROL-H>, that the line kill character is <CONTROL-U> that even parity is set, and that a <RETURN> is acceptable as a new line character. Each of the above characteristics can be set with **stty**.

Changing Terminals

If you have to log in to XENIX on a terminal of a type different than the terminal you normally use, you may need to change the shell TERM variable. This is normally set to the proper default terminal when you login, but if you switch terminals, you'll need to type something like:

```
TERM=termname; export TERM
```

where *termname* is the name of a known terminal. A wide variety of terminals are supported; terminal names are listed in the system file named */etc/termcap*.

X.4 Status Information

Finding Out Who is on the System

The who command lists the names, terminal line numbers, and login times of all users currently logged on the system. For example, type:

```
who
```

This command should produce something like the following output on your terminal screen:

```
arnold   tty02   Apr  7 10:02
daphne   tty21   Apr  7 07:47
elliot   tty23   Apr  7 14:21
ellen    tty25   Apr  7 08:36
gus      tty26   Apr  7 09:55
adrian   tty28   Apr  7 14:21
```

Finding out What Processes are Running

Because processes can be placed in the background for processing, it is not always obvious which processes you are responsible for. The ps command stands for "process status" and lists information about currently running processes associated with your terminal. For instance, the output from a ps command might look like this:

```
  PID TTY TIME CMD
10308 38  1:36 ed chap02.man
   49 38  0:29 -sh
11267 38  0:00 sh -c ps
```

The PID column gives a unique process identification number that can be used to kill a particular process. The TTY column gives the terminal that the process is associated with. The TIME column gives the cumulative execution time for the process.

X.5 Process Control and Command Line Editing

Placing A Process in the Background

Normal commands executed at the keyboard are executed in strict sequence: one must finish executing before the next can begin. Executing commands of this type are called foreground processes. A background process, in contrast, need not finish executing before you execute your next command. Background commands are

especially useful for commands that may take several minutes or even hours to complete, because they can be placed in the background while you continue executing other commands at your terminal.

To place a process in the background, type an ampersand (&) at the end of the command. For example, to move files from the current directory to another directory, while simultaneously continuing with whatever else you have to do, type:

```
mv file1 file2 file3 otherdir&
```

Note that when processes are placed in the background, you lose control of them as they execute. For instance, typing <INTERRUPT> does not abort a background process. You must use the kill command instead, described below.

Killing a Process

To abort execution of a foreground process press your terminal's <INTERRUPT> key. This kills whatever foreground command you have running. To kill all of your processes that are executing in the background, type:

```
kill 0
```

To kill only a specified process executing in the background, first type:

```
ps
```

Ps displays the Process Identification Numbers (PIDs) of your existing processes:

```
  PID TTY TIME CMD
 3459 03  0:15 -sh
 4831 03  1:52 ed chap01.s
 5185 03  0:00 sh -c ps
```

Next, you might type

```
kill 4831
```

where 4831 is the PID of the process that you want killed.

Erasing A Command Line

When entering commands, typing errors will occur. To erase the current command line so that you can start retyping a new one, enter a <CONTROL-U>, as shown below:

```
kat file2<CONTROL-U>
cat file1
```

In the above command line, the first line is aborted and automatically a newline is generated so that typing may resume. You then can enter the correct command line.

Halting Screen Output

In many cases, you will be examining the contents of a file on the terminal screen. For longer files, the contents will often scroll off the screen faster than you can examine them. To temporarily halt a program's output to the terminal screen, type <CONTROL-S>. To resume output, type any key except <INTERRUPT>.

X.6 File Manipulation

Creating Files

To create an empty file, simply type:

```
>filename
```

Here, filename is the name of the newly created file. The greater-than sign (>) is used to redirect output from the terminal to a file. In this special case no information is sent. In general, new files are created by commands as needed.

Displaying File Contents

To display the contents of a file, use the cat command. Cat displays the contents of a file on the default standard output file which is the terminal screen. For example, the following command displays the contents of file1 on the screen:

```
cat file1
```

Cat can also display the contents of more than one file as in

```
cat file1 file2
```

Combining Files

Cat, as mentioned above, is normally used to send the contents of files to the terminal screen. However, the name cat comes from the word concatenate, and cat is also frequently used to combine files into some other new file. Thus, to combine the two files named file1 and file2, and to create a new file named bigfile, type:

```
cat file1 file2 >bigfile
```

Note here that we are putting the contents of the two files into a new file with the name bigfile. The greater than sign (>) is used to redirect normal output of the cat command from the terminal screen to the new file.

Moving a File

With the mv command, two ways of moving a file are supported:

1. Moving a file so that it now has a new name. For instance, to move a file named text to a new file named book, type:

   ```
   mv text book
   ```

   After this move completes, no file named text will exist in the working directory.

2. Moving a file into a specified directory. In this case, you give the name of the destination directory as the final name in the move command. For instance, to move file1 and file2 into the directory named /tmp, type:

   ```
   mv file1 file2 /tmp
   ```

   The two files you have moved no longer exist in your working directory, but files with the same names now exist in the directory /tmp. The above command has exactly the same effect as typing the following two commands:

   ```
   mv file1 /tmp/file1
   mv file2 /tmp/file2
   ```

Remember that the mv command always checks to see if the last argument is the name of a directory, and, if so, all files designated by filename arguments are moved into that directory.

Renaming a File

To rename a file, you simply "move" the file to a file with the new name: the old name of the file is removed. Thus, to rename the file anon to johndoe, type:

```
mv  anon johndoe
```

Note that moving and renaming a file are essentially identical operations.

Copying a File

There are two forms of the cp command: one in which a file is copied to another file and another in which files are copied into a directory. Thus, to create two copies of a file in your own working directory, you must rename the new copy. To do this, the copy command can be invoked as follows:

```
cp file clone-of-file
```

After the above command has executed, two files with identical contents reside in the working directory.

To copy three files into a directory named filedir, type:

```
cp file1 file2 file3 filedir
```

In the above command, three files are copied into the directory filedir; the original versions still reside in the working directory. There is a one-to-one correspondence between the names in the two directories.

Deleting A File

To delete or remove files, simply type:

```
rm file1 file2
```

In the above command, the files file1 and file2 are removed from your working directory.

X.7 Editing

The following commands all deal with use of the XENIX line editor, ed.

Invoking the Editor

To invoke ed, type:

ed filename

where filename is the name of the file you want to edit. If no name is given, a question mark (?) is printed. This is not an error -- (you are simply creating and editing a new file. The text in the new file being worked on is kept in a special buffer file. Think of the buffer as a work space that you are going to be editing. You tell ed what to do to your text by typing instructions called "commands." Most commands consist of a single letter, which must be typed in lowercase. Each command is typed on a separate line and terminated with a <RETURN>. Ed makes no response to most commands -- there is no prompting or typing of messages. If at any time you make an error in the commands you type to ed, it will tell you by typing:

?

Adding Text to a File

To enter lines of text into the buffer, just type an "a" to append lines to the file, followed by a <RETURN>. Next, enter the lines of text you want, like this:

```
a
Now is the time
for all good men
to come to the aid of their party.
.
```

To stop appending, type a line that contains only a period on a line by itself. A period (.) is used to tell ed that you have finished appending. (You can also use <CONTROL-D>.)

Writing Out The Editing Buffer

It's likely that you will want to save your text for later use. To write out the contents of the ed editing buffer into a file, use the "write command, w, followed by the name of the file that you want to write to. This copies the buffer's contents to the specified file, destroying any

previous information in the file. For example, to save the text in a file named text, type:

```
w text
```

Leave a space between w and the filename. Writing a file just makes a copy of the text -- the buffer's contents are not disturbed, so you can go on adding lines to it. By default, ed writes out to the file that you named on the ed invocation line. Ed at all times works on a copy of a file, not the file itself. No change in the contents of a file takes place until you give a w command.

Exiting The Editor

To terminate a session with ed, save the text you're working on by writing it to a file using the w command, and then type:

```
q
```

The system responds with the XENIX prompt character. At this point your buffer vanishes, with all its text, which is why you want to write it out before quitting. Actually, ed will print "?", if you try to quit without writing. At that point, write out the text if you want; if not, type another q

A good way to operate is this:

```
ed file
[editing session]
w
q
```

This way, you can type w from time to time and be secure in the knowledge that if you got the filename right in the beginning, you are writing out to the proper file each time.

Printing Editing Buffer Contents

To print the contents of the buffer (or parts of it) on the terminal screen, use the "print" command, p. To do this, specify the lines where you want printing to begin and where you want it to end, separated by a comma, and followed by the letter "p". Thus, to print the first two lines of the buffer, (that is, lines 1 through 2) type:

```
1,2p
```

Suppose you want to print all the lines in the buffer. You'll then want to use the shorthand symbol for "the line number of the last line in the buffer" -- the dollar sign ($). Use it this way:

```
1,$p
```

This will print all the lines in the buffer (from line 1 to the last line).

To print the last line of the buffer, type:

```
$p
```

You can print any single line by typing the line number, followed by a p. Thus

```
1p
```

prints the first line of the buffer.

In fact, you can print any single line by typing just the line number; there's no need to type the letter p. So if you type

```
$
```

ed prints the last line of the buffer.

An area in which you can save typing effort in specifying lines is to use plus and minus as line numbers by themselves. For example

```
-
```

by itself is a command to move back up one line in the file.

Ed maintains a record of the last line that you did anything to (in this case, line 3, which you just printed) so that it can be used instead of an explicit line number. The line most recently acted on is referred to with a period (.) and is called "dot." Dot is a line number in the same way that dollar ($) is; it means "the current line," or loosely, "the line you most recently did something to." You can find out the value of dot at any time by typing:

```
.=
```

Deleting Text

To delete text from the ed editing buffer, use the "delete" command, **d**. The lines to be deleted are specified for d exactly as they are for p:

starting-line,ending-lined

Thus, the command

```
4,$d
```

deletes lines 4 through the end. There are now three lines left in our example, as you can check by typing:

```
1,$p
```

Notice that $ now is line 3! Dot is set to the next line after the last line deleted, unless the last line deleted is the last line in the buffer. In that case, dot is set to $.

Substituting Text

The "substitute" command, **s**, is used to replace a string of characters existing in the editing buffer with another string that you specify. This is the command that is used to change individual words or letters within a line or group of lines. For example, suppose that, due to a typing error, line 1 says:

```
Now is th time
```

The letter "e" has been left off of the word "the". You can use s to fix this up as follows:

```
1s/th/the/
```

This says to substitute for the characters "th", the characters "the", in line 1. To verify that the substitution has worked, type

```
p
```

to get

```
Now is the time
```

which is what you wanted. Notice that dot must have been set to the line where the substitution took place, since the p command printed that line. Dot is always set this way with the s command.

The syntax for the substitute command follows:

starting-line,ending-lines/pattern/replacement/

Whatever string of characters is between the first pair of slashes is replaced by whatever is between the second pair, in all the lines between starting-line and ending-line. Only the first occurrence on each line is changed, however. The rules for line numbers are the same as those for p, except that dot is set to the last line changed. (But there is a trap for the unwary: if no substitution takes place, dot is not changed. This causes printing of a question mark (?) as a warning.

Thus, you can type

```
1,$s/speling/spelling/
```

and correct the first spelling mistake on each line in the text.

If no line numbers are given, the s command assumes we mean "make the substitution on line dot," so it changes things only on the current line. This leads to the very common sequence

```
s/something/something else/p
```

which makes some correction on the current line, and then prints it, to make sure it worked out right. (Notice that there is a p on the same line as the s command. With few exceptions, p can follow any command; no other multi-command lines are legal.) To change all occurrences on the current line, you should type:

```
s/something/something else/g
```

where g stands for a global substitution of all occurrences on the line.

It's also legal to type

```
s/string//
```

which means "change the first string of characters to nothing," or, in other words, remove them.

Searching

Suppose you have the following three lines of text in the editing buffer:

```
Now is the time
for all good men
to come to the aid of their party.
```

Now, suppose you want to find the line that contains the word "their", so that you can change it to the word "the.". With only three lines in the buffer, it's pretty easy to keep track of which line the word "their" is on. But if the buffer contained several hundred lines, and you'd been making changes, deleting and rearranging lines, and so on, you would no longer really know what this line number would be. Context searching is simply a method of specifying the desired line, regardless of what its number is, by specifying some textual pattern contained on the line.

The way to say "search for a line that contains this particular string of characters" is to type:

```
/string of characters we want to find/
```

For example, the ed command

```
/their/
```

is a context search which is sufficient to find the desired line -- it will locate the next occurrence of the characters between slashes ("their"). It also sets dot to that line and prints the line for verification:

```
to come to the aid of their party.
```

"Next occurrence" means that ed starts looking for the string at line .+1, searches to the end of the buffer, then continues at line 1 and searches to line dot. (That is, the search "wraps around" from $ to 1.) It scans all the lines in the buffer until it either finds the desired line or gets back to dot again. If the given string of characters can't be found in any line, ed prints an error message:

```
?
```

Otherwise, ed prints the line it found.

X.8 Directory Manipulation and Travel

Printing Your Working Directory

All commands are executed relative to a "working" directory. The name of this directory is given by the **pwd** command, which stands for "print working directory." For instance, if your current working directory is /usr/joe, then when you type

```
pwd
```

you will get the output:

```
/usr/joe
```

You should always think of yourself as residing "in" your working directory.

Listing Directory Contents

The most basic directory command is **ls**. The **ls** command sorts and lists the names of the files and directories that reside in a given directory. By default, the contents of your working directory are listed. If arguments are given, then for each directory argument **ls** lists the contents of the given directory; for each file argument, **ls** repeats its name. For instance, if you type

```
ls
```

the output from the command might typically look like this:

```
dir1
dir2
dir3
file1
file2
file3
```

Using the same directory, the command

```
ls d*
```

would list the files within each of the directories dir1, dir2, and dir3.

When working at a terminal, it is sometimes distracting to see the list of names from the **ls** command scroll off

the screen. This problem can be avoided by using the lc command, which stands for "list in columns." Because names are printed in columns, more information can fit on the screen than with ls. A sample listing follows:

```
atfile      help        oem         size        v0
bin         lib         papers      src         v1
calendar    maketape    po          termcap     v2
cmds        memos       port        termnames   v5
convert     mgr         probs       test.s
doem        mkfs        rand        testdir
errs        msg         rand.c      ttc
errs.sh     nroff       sco         typeset
```

Note that when lc sends output through a pipe, no columns are sent. This is so that you can use lc like ls whenever you want. If you really want the output to a pipe to be columnar, you can force it with the -C switch. Thus, to send a columnar listing to the line printer, you would type:

```
lc -C | lpr
```

Note that lc also lets you recursively list a directory and all of its subdirectories by typing

```
lc -R
```

where the -R stands for recursive.

A command very similar to ls and lc is the l command. L gives an expanded "long" listing of a directory, producing an output that might look something like this:

```
total 501
drwxr-x--- 2 boris     272 Apr  5 14:33 dir1
drwxr-x--- 2 enid      272 Apr  5 14:33 dir2
drwxr-x--- 2 iris      592 Apr  6 11:12 dir3
-rw-r----- 1 olaf      282 Apr  7 15:11 file1
-rw-r----- 1 olaf       72 Apr  7 13:50 file2
-rw-r----- 1 olaf     1403 Apr  1 13:22 file3
```

Reading from left to right, the information given for each file or directory includes:

1. Permissions

2. Number of links

3. Owner

4. Size in bytes

5. Time of last modification

6. Filename

The information in this listing and how to change permissions are discussed below in Section X.9, "File and Directory Permissions."

Changing Your Working Directory

Your working directory represents your location in the file system: it is "where you are" in XENIX. To alter your location in the XENIX file system, you need only type:

```
cd
```

This changes your working directory to that of your home directory. To move to any given directory, simply specify that directory as an argument to cd. For instance

```
cd /usr
```

moves you to the /usr directory. Because you are always "in" your working directory, changing working directories is much like "traveling" from directory to directory.

Creating a Directory

To create a subdirectory in your working directory, use the mkdir command. For instance, to create a new directory named phonenumbers, simply type:

```
mkdir phonenumbers
```

After this command has been executed, a new but empty directory will exist in the your home directory.

Removing a Directory

To remove a directory located in your working directory, use the rmdir command. For instance, to remove the directory named phonenumbers from the current directory, simply type:

```
rmdir phonenumbers
```

Note that the directory phonenumbers must be empty before it can be removed; this prevents catastrophic deletions of files and directories.

Renaming a Directory

To rename a directory, use the **mv** command. Note that directories and their contents cannot be moved with the mv command; they can only be renamed. For instance, to rename the directory little.dir to big.dir, type:

```
mv little.dir big.dir
```

This is a simple renaming operation; no files are moved.

X.9 File and Directory Permissions

To determine the permissions associated with a given file or directory, use the **l** command. Permissions are given by the first ten characters of the output from this command. A sample output follows:

```
total 501
drwxr-x--- 2 boris     272 Apr  5 14:33 dir1
drwxr-x--- 2 enid      272 Apr  5 14:33 dir2
drwxr-x--- 2 iris      592 Apr  6 11:12 dir3
-rw-r----- 1 olaf      282 Apr  7 15:11 file1
-rw-r----- 1 olaf       72 Apr  7 13:50 file2
-rw-r----- 1 olaf     1403 Apr  1 13:22 file3
```

The first character is:

d	if the entry is a directory
-	if the entry is a ordinary file

The next nine characters are interpreted as three sets of three permissions each. The first set refers to user (owner) permissions; the next to permissions for others in a group associated with the file or directory; and the last to permissions for all other users. Within each set, the three characters indicate permission to read, to write, and to execute the file as a command, respectively. For a directory, "execute" permission is interpreted to mean permission to search the directory for any included files or directories. In summary, these permissions are indicated as follows:

r if the file is readable

w if the file is writable

x if the file is executable

- if the indicated permission is not granted

Changing Read, Write, and Execute Permissions

To change a file or directory's read, write, and execute permissions, you need only use one command: **chmod**. Several examples are included here to illustrate the syntax for this command. Remember that there are three distinct classes of users: user, group, and other, and that for each class all three permissions are either set or not set.

For example, presume that a file named filel exists with the following permissions:

```
-rw-r-----
```

To give filel read permission for all classes of users, type:

```
chmod a+r filel
```

Here, "a" stands for "all," which is a synonym for "ugo" where "u" stands for user, "g" for group, and "o" for other users. If no user class is specified, then "a" is normally the default, although this default may vary. The resulting permissions are:

```
-rw-r--r--
```

To give write and execute status to members of a group only, type:

```
chmod g+wx filel
```

For a filel with the attributes

```
-rw-------
```

the above command would alter attributes so that they looked like this:

```
-rw--wx---
```

To remove write and execute permission by the user

(owner) and group associated with filel, type:

```
chmod ug-wx filel
```

Note how letters are grouped to set specific user/permission combinations.

Changing Directory Search Permissions

Obviously, directories cannot be executable files, yet they, also, have an execute attribute that can be set. This attribute signifies search permission; if this permission is denied to a particular user, then that user cannot even list the names of the files in the directory.

For example, assume that the directory dirl has the following attributes:

```
drwxr-xr-x
```

To remove search and read permission for other users to examine dirl, type:

```
chmod o-rx dirl
```

The new attributes for dirl are then:

```
drwxr-x---
```

X.10 Information Processing

Sorting A File

One of the most generally useful file processing commands is sort. By default, sort sorts the lines of a file according to the ASCII collating sequence. For example, to sort a file named phonelist, type:

```
sort phonelist
```

In the above case, the sorted contents of the file are displayed on the screen. To create a sorted version of phonelist named phonesort, type:

```
sort phonelist >phonesort
```

Note that sort is useful in sorting the output from other commands. For example, to sort the output from execution of a who command, type:

```
who | sort >whosort
```

A wide variety of other sorting options are available for sort.

Searching for a Pattern in a File

The grep command selects and extracts lines from a file, printing only those lines that match a given pattern. A useful way to think of the name of this command is as a synonym for the word "grab."

For example, to print out all lines in a file containing the word "tty38", type:

```
grep 'tty38' file
```

In general, you should always enclose the pattern you are searching for in single quotes ('), so that special metacharacters are not expanded unexpectedly by the shell command interpreter.

X.11 Line Printers

Sending a File to the Line Printer

One of the most common operations that you will want to perform is printing files on the line printer. The most straightforward method for doing this is to type

```
lpr file1
```

for one file, or

```
lpr file1 file2 file3
```

for multiple files. Other common uses of lpr involve pipes. For instance, to paginate and print a file of raw text, type:

```
pr textfile | lpr
```

The pr and lpr commands are very often used together. As another example, to sort, paginate, and print a file, type:

```
sort datafile | pr | lpr
```

Getting Line Printer Queue Information

More than one file may be waiting to be printed at the line printer: XENIX does not require that the file be printed before the lpr command finishes. Instead, lpr makes sure only that the file is placed in a special directory where it will wait its turn to be printed. Thus, to examine the files in the line printer queue, type:

```
ls -l /usr/spool/lpd
```