EUNICE BSD

ADMINISTRATOR'S GUIDE

August 1988

THE WOLLONGONG GROUP, INC.
1129 San Antonio Road
Palo Alto, California 94303
(415) 962-7100
TWX 910-373-2085

Restricted Rights

Use, duplication or disclosure by the Government is subject to restrictions as set forth in subparagraph (c)(1)(ii) of the Rights in Technical Data and Computer Software clause at 52.227-7013. The Wollongong Group, Inc., 1129 San Antonio Road, Palo Alto, California 94303, U.S.A.

Copyrights

Trademarks

EUNICE, REX, and WIN/TCP are trademarks of The Wollongong Group, Inc.

DEC, DECnet, VAX, VMS, and MicroVAX are trademarks of Digital Equipment Corporation.

UNIX is a registered trademark of AT&T.

EUNICE BSD

ADMINSTRATOR'S GUIDE

CONTENTS

LIST OF FIGURES AND TABLES

Document Conventions

The following conventions are used throughout this guide:

$	VMS system prompt.
%	UNIX system prompt.
[ ]	Enclose VMS device name. CTRL key
Bold	Literal commands to be entered exactly as shown. (Observe case sensitivity rules.)
Italics	1) Variable parameters, 2) UNIX BSD commands, or 3) networking commands.
Regular	System response to a user's command.
<< >>	Enclose descriptions of site-specific information.

The VMS DCL (DIGITAL Command Language) converts all file name characters into upper-case letters before processing. UNIX, however, is case-sensitive, and the majority of the commands are in lower case. Examples and file names are given in VMS DCL environment format, except in those cases where the UNIX context is obvious to the reader.

Where VMS device names are to be entered, we have assumed a typical case: "DUA0:" for the disk, and "MUA0:" for the tape drive. Please change these command lines as necessary for your system configuration.

Related Publications

Other Wollongong documents in this product set are:

	EUNICE BSD User Guide
Vol I:	User's Reference Manual
Vol Ia:	SCCS Reference Manual
Vol II:	Programmer's Reference Manual
Vol III:	User's Supplementary Documents
Vol IV:	Programmer's Supplementary Documents
Vol V:	System Administrator's Guide

To order additional copies of the Wollongong publications listed here, write to:

The Wollongong Group, Inc.
Attn: Sales Dept.
1129 San Antonio Road
Palo Alto, CA 94303

or call:

(415) 962-7200

EUNICE™ BSD

ADMINISTRATOR'S GUIDE

1. INTRODUCTION

This guide is designed to enable the user to install EUNICE BSD by typing the lines presented in bold font. Text describing the installation activity accompanies each emboldened section.

If a previous version of EUNICE BSD is currently installed, please note that it will have to be removed before the new EUNICE BSD may be read in from the tape. The new EUNICE BSD constitutes a significantly different system as compared to the previous versions. For this reason, we urge the system administrator to individually save all important data and program source files which might be scattered throughout the existing EUNICE BSD hierarchy (e.g. */usr/spool/mail/**, */usr/spool/uucp/**, */usr/local/**, the */tmp* or */usr/tmp* files, and other site-specific software locations) for subsequent addition to the new release.

After doing this, either rename the [EUNICE...] hierarchy, or perform a complete backup of [EUNICE...]*.*; and then delete the whole directory tree. **Under no circumstances should you mix old EUNICE code with the new.** For complete details on updating a system which has previously been running EUNICE, please refer to Section 2.1 before starting the installation.

> **NOTE: Executables that were created with the older version will function only if the old logical names (ETC, USR, ...) are preserved, and the old shareable library is preserved. We recommend that the programs be recompiled. ALL source code will have to be recompiled using the new shareable libraries.**

The following resources are necessary for EUNICE BSD, Version 4.3.2:

> VMS™ 4.2 or later releases
> 1600 bpi tape drive (or, on a MicroVAX™, a TK-50 drive)
> Approximately 35 MB of disk space (excluding sources)
> UNIX® sources (provided for UNIX source code licensees) require an additional 30 MB of disk space

It may be necessary to reboot the system after installing EUNICE BSD, especially if the non-dynamic SYSGEN parameters are changed. (See Section 2.4, "Checking VMS SYSGEN Parameters".)

1.1 THE PRODUCT PACKAGE

The product package consists of:

1. The *EUNICE BSD Product Release Notice*
2. A 2400-foot tape with EUNICE/UNIX binaries (or, in the case of MicroVAX systems, a TK-50 tape cartridge)

3. 6-Volume set of EUNICE BSD documentation which is based on UNIX 4.3 BSD, and includes EUNICE BSD specific modifications. The set consists of the following manuals:

Vol I:	User's Reference Manual
Vol Ia:	SCCS Reference Manual
Vol II:	Programmer's Reference Manual
Vol III:	User's Supplementary Documents
Vol IV:	Programmer's Supplementary Documents
Vol V:	System Administrator's Guide

Sites with System V licenses also receive:

An extra save-set on the release tape which contains SCCS binaries.

Sites with UNIX source code licenses will also receive:

A 2400-foot tape (or TK-50 cartridge tape) containing the UNIX source code.

Shipments vary according to site. Please check your packing slip.

1.2 THE EUNICE BSD REFERENCE MANUAL

The *EUNICE BSD Reference Manual* is a companion to this Guide. It contains important information for both EUNICE BSD administrators and users.

Attention is especially drawn to:

Section 2: EUNICE BSD Command Availability

Lists the 4.3 BSD UNIX commands supported, or not supported, by EUNICE BSD, as well as EUNICE-specific commands.

NOTE: The introduction (intro(n)) pages of some of the BSD manual sections have been altered to indicate commands and features that have *not* been included in EUNICE BSD.

Section 3: EUNICE BSD Functional Description

This is of particular interest to programmers, as it explains how EUNICE BSD is implemented on VMS. It includes information on UNIX system calls (much of which has been incorporated as EUNICE NOTES to the *UNIX Programmer's Reference Manual [PRM]*) and internal EUNICE BSD calls.

Section 4: VMS/UNIX Interface source, object, and executables

Includes hints for using the C and *Fortran*77 compilers.

Section 5: Guide to Problem Solving

A series of EUNICE BSD run-time problems and their typical solutions.

1.3 UNIX ACCESS ACCOUNTING

All sites are provided with the *eunlogin(1)* and *eunlogout(1)* programs which restrict the number of simultaneous UNIX users. This restriction is based on the EUNICE BSD/UNIX site license.

1.4 VMS 4.x AND EUNICE BSD

Appendix A describes new features of VMS that affect EUNICE BSD.

1.5 NEW FILENAME HASHING

A faster filename encoding mechanism has been incorporated in the EUNICE BSD 4.3.1 release. The new algorithm uses a single file (compared to the old method of two HSH files) with the encoding occurring solely in the new filename. This fact, combined with VMS 4.x's (and later) ability to use long filenames, means that almost no UNIX filenames need to be hashed, allowing a faster overall response time.

All files with names hashed under EUNICE BSD Release 3.2 and earlier will need to be upgraded with the VMS 4.x file naming convention. Sites which are updating from EUNICE BSD, Release 4.2 will have no such problems. The new EUNICE BSD will **not** hash filenames that are up to 39 lowercase characters followed by an extension of up to 39 lowercase characters. Filenames which are longer than "39.39" lowercase characters **will** be hashed according to a scheme similar to the one on EUNICE BSD Release 3.X - two files will be created - one bearing the filename, and the other the actual contents of the file.

1.6 OVERVIEW- INSTALLATION PROBLEM SOLVING

Appendix B contains possible solutions to installation problems, and should be consulted prior to contacting The Wollongong Group (Wollongong).

1.7 TUNING SYSTEM PARAMETERS

Appendix C gives suggestions on the fine-tuning of several system parameters. There are other tuning suggestions in */usr/doc/eunice/sysgen.ms*.

1.8 PRODUCT SUPPORT

When encountering problems, please consult Appendices B and C first. Before contacting Wollongong, we request that you also read Appendix D and E on the subject of Product Support, and how to use it. Appendix E also has further details on EUNICE BSD Support.

2. PREPARING THE VMS ENVIRONMENT FOR EUNICE BSD

EUNICE BSD is distributed on a 9-track, 1600 bpi, 2400-foot magnetic tape which is in VMS BACKUP format. Purchasers of EUNICE BSD 4.3.1 for the MicroVAX will receive the distribution on a TK-50 cartridge tape. The distribution tape comprises ten save sets, nine of which are read in by means of a command file. Sites that do not have a UNIX System V license will not receive the last save-set on the tape (SCCS.BCK). The save sets and their directories are described in Table 2-1.

Table 2-1. Tape Files

SAVE SET	DIRECTORY	DESCRIPTION
INSTALL1.BCK	SYS$SCRATCH:	EUNICE BSD installation tools
INSTALL2.BCK	SYS$SCRATCH:	EUNICE BSD installation tools
LIBCSHARE.BCK	SYS$SHARE:	Shareable libraries
ROOT.BCK	[EUNICE.BIN...]	*/bin* directory (basic UNIX utilities)
	[EUNICE.ETC...]	*/etc* directory (system maintenance files and programs)
	[EUNICE.LIB...]	*/lib* directory (basic UNIX library files and programs)
	[EUNICE.VOS...]	*/vos* directory (Virtual Operating System (VOS) interface)
TMP.BCK	[EUNICE.TMP]	*/tmp* directory (temporary working area for UNIX utilities)
USR.BCK	[EUNICE.USR...]	*/usr* directory and subdirectories (UNIX files and programs)
MAN.BCK	[EUNICE.USR...]	*/usr/man/man** directories (on-line manual pages)
DOC.BCK	[EUNICE.USR...]	*/usr/doc* documents and other on-line documentation
LEARN.BCK	[EUNICE.USR...]	*/usr/lib/learn* the learn(1) utility and its attendant files

Table 2-1. Tape Files (Continued)

SAVE SET	DIRECTORY	DESCRIPTION
SCCS.BCK	[EUNICE.USR...]	*/usr* also houses the binary code for the Source Code Control System. Customers with a System V license will receive this extra save set.
EUN_SRC.BCK	[USR.SRC...]	*/usr/src* and subdirectories (UNIX 4.3 BSD sources) available only to those customers who have a UNIX source code license. This file is distributed on a separate tape.
SCCS_SRC.BCK	[USR.SRC...]	*/usr/src* are subdirectories (UNIX System V SCCS sources) available only to those customers who have a System V source code license.

2.1 DELETING THE PREVIOUS EUNICE BSD RELEASE (IF PRESENT)

If you do NOT already have a version of EUNICE BSD on the system, proceed to Section 2.2.

As was suggested in the previous chapter, if your system is currently running a previous version of EUNICE BSD, it will have to be removed before the new software can be installed. The following steps are an example of how the current EUNICE BSD system can be disabled such that it will not interfere with the installation of the new version.

1. Log in on a CRT-type terminal and edit the SYS$MANAGER:SYSTARTUP.COM file. Comment out the lines that start up the EUNICE BSD system. If your system is running WIN/TCP™, it is recommended that you also comment out the call to STARTINET.COM - since the logical names used by WIN/TCP will conflict with those of EUNICE BSD.

   $ SET DEFAULT SYS$MANAGER:
   $ EDIT SYSTARTUP.COM

2. Reboot the system so that all the EUNICE BSD files and global sections are de-installed and unlocked, thus allowing them to be easily deleted.

   $ SET DEFAULT SYS$SYSTEM:
   $ @SHUTDOWN
   <<Multiple system prompts and responses>>

3. Having logged back in again, either rename (or backup and delete) the existing [EUNICE...] hierarchy so that the new EUNICE BSD can be read in without conflicts.

```
$ SET DEFAULT DUA0:[000000]
$ RENAME EUNICE.DIR OLDEUNICE.DIR
```

4. Only if you are upgrading directly from EUNICE 4.2, go to the SYS$SYSTEM directory and delete all versions of the following files: DEV.;*, ROOT.;*, EUNUSERS.GBL;*, LAVDRIVER.EXE;* and VFORKCOMM.GBL;*.

```
$ SET DEFAULT SYS$SYSTEM:
$ DELETE DEV.;*, ROOT.;*,—
EUNUSERS.GBL;*,—
LAVDRIVER.EXE;*,—
VFORKCOMM.GBL;*
```

Once these files have been removed, only the EUNICE BSD administration directory (located in the SYS$MANAGER directory) should remain. When the command procedure INSTALL.COM reads in the release tape, it will place several setup command procedures in a sub-directory called [.EUNINSTALL] in SYS$MANAGER. This directory may still be present on your system. We suggest that you delete it, along with its contents, before beginning to install the new release of EUNICE BSD. This will insure that the system is properly prepared for installation of the new EUNICE BSD.

2.2 SETTING UP THE INSTALLATION ENVIRONMENT

Before beginning the EUNICE BSD installation, you will need to verify (and if necessary modify) several areas within the VMS environment so that EUNICE BSD will operate properly. To begin, log on to the system on the console, OPA0: - using the SYSTEM account and, for safety, remove the BYPASS privilege. (By using the console device, you will have a hardcopy of the activities you perform. This output will be very useful later on.)

```
USERNAME: SYSTEM
PASSWORD:  <<enter system password>>
$ SET PROC/PRIV=(ALL,NOBYPASS)
```

Now, obtain a listing of all the devices available on your system:

```
$ SHOW DEVICES
      << multiple system responses >>
```

This listing will make the setup of the EUNICE BSD device "directory" much easier, since when you edit the command files which set up the logicals, you will have a hardcopy to use as a checklist for the different devices. This lessens the chance that you might forget one of them.

2.3 VERIFICATION/MODIFICATION OF SYSTEM AND USERS' ACCOUNTS

The VMS operating system provides many more adjustable parameters which affect the system, the user accounts, and processes, than does native UNIX. As a result, there are a number of abilities and permissions which are taken for granted in UNIX that must be specifically set under VMS. For EUNICE BSD to run well, the users' accounts must be set up with specific minimum privileges and quotas.

The AUTHORIZE program is used to check and modify login accounts. Relevant quotas for important areas are shown in Table 2-2. Notice that a range is given for the majority of the quotas.

Two privileges are required to use EUNICE BSD: TMPMBX and NETMBX. (See Table 2-3.) The TMPMBX privilege is particularly important, since it allows the UNIX piping capability within EUNICE BSD to function properly. These are the minimum privileges required for both SYSTEM and unprivileged users.

The SYSTEM user's quotas should be set to the maximum value for each item listed in Table 2-2. (This will facilitate successful administration of the EUNICE BSD software.)

Every unprivileged EUNICE BSD user's account should be checked to make sure that it has at least the minimum quotas required for EUNICE BSD. If the quotas shown for your users are **greater** than those indicated here, then **do not lower them.** These suggested numbers are **minimum** values.

Table 2-2. EUNICE BSD User Account Quotas

QUOTA	SYMBOL	RANGE
Subprocess creation limit	PRCLM	10-24*
Base priority	PRIO	4
File and logical link limit	FILLM	100-200**
Asynchronous system traps limit	ASTLM	40-50
Timer queue entry limit	TQELM	20-30
Buffered I/O byte count limit	BYTLM	50000-64000***

* A PRCLM of 6 is an absolute minimum. Because UNIX utilities create many sub-processes, 16 is a more realistic value. Programmers who use Makefiles, or users who do a lot of piping may require as many as 24.

** The FILLM parameter's minimum is set at 100 to reflect approximately 5 times the value of PRCLM.

*** Users who redirect large amounts of data, use many pipes, or who use major memory-consuming programs like *troff(1)*, will need to have their BYTLM quota increased to between 50000 and 64000.

Table 2-3. EUNICE BSD User Account Privileges

All users should have:	TMPMBX, NETMBX

To compare a user's quotas or privileges to the limits specified above, run the AUTHORIZE program and execute the SHOW command.

Once the system quotas and privileges have been appropriately set for each EUNICE BSD user, exit from the AUTHORIZE program. In the example below, the account for the SYSTEM user is being examined.

```
$ SET DEF SYS$SYSTEM
$ RUN AUTHORIZE
UAF> SHOW SYSTEM
UAF> << modify as necessary using data in Tables 2-2 and 2-3 >>
For example: MODIFY SYSTEM/FILLM = 100
user record(s) updated
UAF> SHOW SYSTEM
UAF> EXIT
```

2.4 CHECKING VMS SYSGEN PARAMETERS

Next, verify and if necessary, modify the value of the maximum process parameter (MAXPROCESSCNT). Begin by setting the default directory to the proper location. Then run the SYSGEN program on the CURRENT system parameters, and check the VMS SYSGEN parameters.

For further tips on VMS tuning, please see Appendix C of this document.

```
$ SET DEF SYS$SYSTEM:
$ RUN SYSGEN
SYSGEN> USE CURRENT
SYSGEN> SHOW MAXPRO
MAXPROCESSCNT nn nn nn nnnn Processes
        << If the value of the first field is less than 64, then enter the following >>
          SYSGEN> SET MAXPRO 64
          SYSGEN> WRITE CURRENT
```

Since EUNICE BSD executables are installed from seven directories, the KFILSTCNT (the known file list head count) will increase by seven. Depending on current usage, this value may need to be increased by running SYSGEN. Later in the installation procedure, the command files will install various programs with privilege. If the KFILSTCNT is not high enough, an error will be returned. Run SYSGEN and do a SHOW KFILSTCNT to check this parameter (the first number printed is the applicable one), and then adjust it as necessary. Wollongong has found that values between 16 and 20 work well in almost all cases.

```
SYSGEN> SHOW  KFIL
KFILSTCNT nn nn nn nnnn Slots
                << If the first number is less than 16, raise it;  otherwise raise it
                   as necessary for your system.  See your VMS manual for
                   full details. >>
                For example:  SYSGEN> SET KFIL 20
                              SYSGEN> WRITE CURRENT
SYSGEN> EXIT
```

A total of 661 global pages will be used by EUNICE BSD. This value is also increased by means of the SYSGEN utility. VMS will return an error message later in the installation if this value is not high enough. Run INSTALL and do a /GLOBAL to check the unused global pages, and ensure that there are about 700 unused pages. If there are not enough, raise the GBLPAGES SYSGEN parameter by 700 using the syntax shown in the examples above. The last line printed after the output from the /GLOBAL command will contain the pertinent data.

```
$ RUN  INSTALL
INSTALL> /GLOBAL
      <<multiple system responses>>
      nn Global Sections Used,  nnnn/nnnn Global Pages Used/Unused
INSTALL> /EXIT
```

After changing the SYSGEN parameters interactively, it is recommended that the system administrator update the SYS$SYSTEM:MODPARAMS.DAT file, adjusting it to reflect the required values of these parameters. Wollongong recommends that you include comments in the file explaining (for future reference) that these parameter values are necessary for EUNICE BSD to work. If the old values remain in this file, an AUTOGEN reboot could reset these to lower values and EUNICE BSD would not function properly. Follow the instructions in the VMS manuals regarding the use of this file.

```
$ EDIT  SYS$SYSTEM:MODPARAMS.DAT
```

2.5 REGISTERING NEW SYSGEN PARAMETERS

NOTE: After modifying any non-dynamic SYSGEN parameter, including MAXPRO, KFILSTCNT, or GBLPAGES, the system must be re-booted for these new values to take effect. Setting these values should be the only reason why a re-boot would be necessary during the installation.

If you wish to minimize the number of system reboots, please take the time to read Appendix C, which discusses the tuning of SYSGEN parameters. Sites which are planning to bring up *uucp(1)* should also read Section 10.1, since there is one further system parameter there which needs to be set correctly.

Once all these areas have been checked, the reboot may be performed (if necessary). Therefore, if any of the non-dynamic parameters have been changed, re-boot the VAX by means of the "SHUTDOWN" command file:

```
$ @SYS$SYSTEM:SHUTDOWN.COM
```

2.6 SELECTING DISK STORAGE SPACE FOR EUNICE BSD FILES

EUNICE BSD requires that one shareable library file resides in the SYS$SHARE: directory. Five directory trees contain the remainder of the EUNICE BSD executables. Table 2-4 shows the various filenames and the location where each is to reside.

Table 2-4. Files and File Locations

EUNICE Run-Time Environment Files	File Location	Size in VMS 512 Byte Blocks
TWG_LIBC_43.EXE	SYS$SHARE:	203
[.BIN...]	[EUNICE]	
[.ETC...]	[EUNICE]	Total to
[.LIB...]	[EUNICE]	4941
[.VOS...]	[EUNICE]	
[.USR...]	[EUNICE]	40841
[.USR.MAN...]	[EUNICE]	4373
[.USR.DOC...]	[EUNICE]	8949
[.USR.LIB.LEARN...]	[EUNICE]	1458
[.USR.LIB.SCCS...]	[EUNICE]	2183

The largest directory hierarchy is /usr. If disk space is a problem, the following directories may be considered for deletion:

/usr/doc	8900* blocks	documentation
/usr/games	1870* blocks	games and subdirectories
/usr/lib/learn	1400* blocks	learn facility
/usr/man	4300* blocks	on line manual pages

* These sizes are approximate.

The balance of the software requires that the top-level directory be placed at the root of a disk or disks having a total of **approximately** 35 MB of free space (excluding UNIX sources), which translates to approximately 60,000 blocks of disk space. The most effective way to install EUNICE BSD is to provide a top level directory called [EUNICE] and then to place the contents of the distribution tape in that directory.

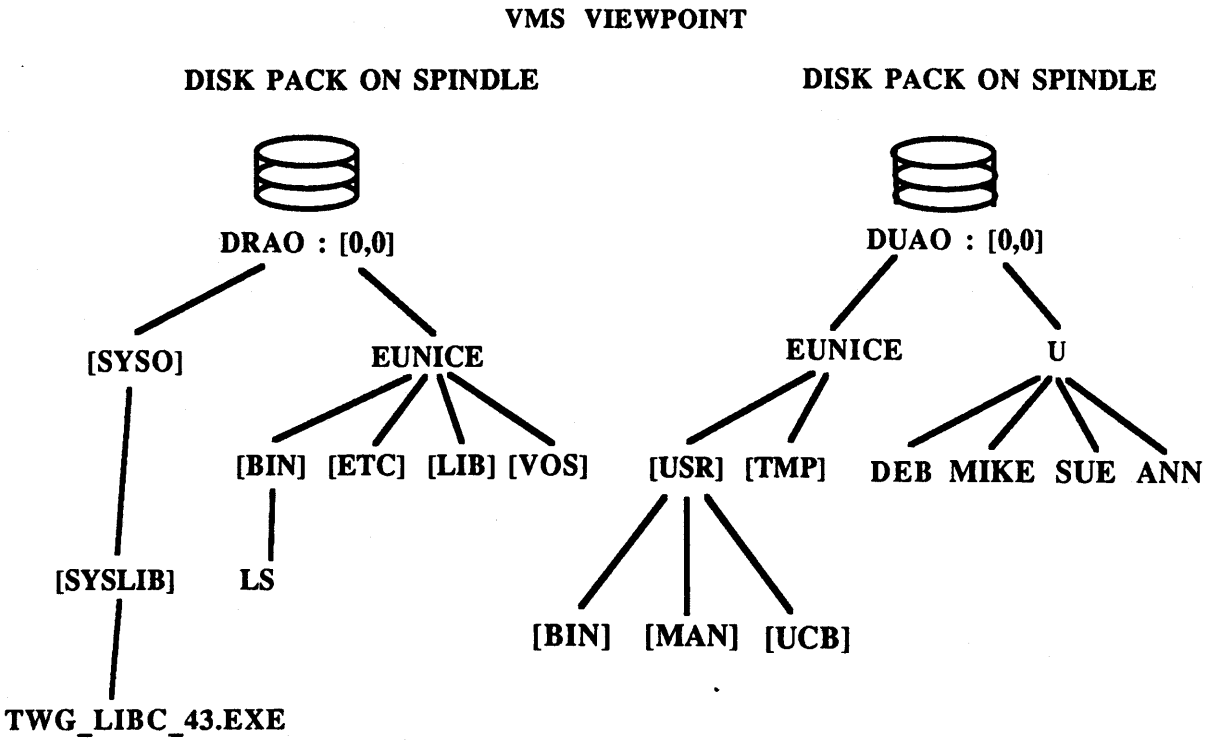
In the following sections, the discussion and examples are based on the assumption that all the files reside on the VMS device DUA0:. All references to DUA0: should be replaced with the VMS device names of the disk or disks selected for storage of EUNICE BSD files.

If your system has less than 35 MB available on a single disk, it is a good idea to place the directories BIN.DIR, ETC.DIR, LIB.DIR, and USR.DIIR on two or more separate disk drives. It is recommended, however, that each of these directories be set up as subdirectories of a single top-level directory called EUNICE.DIR. This strategy avoids directory-name conflicts and simplifies site configuration management.

Since the UNIX hierarchy is emulated by using VMS logical names for the directory specifications, it is not crucial that the entire EUNICE BSD system be resident upon one disk. For example, if your system has insufficient disk space on the system disk, system performance will be improved by having the */bin*, */etc*, */lib*, and */tmp* hierarchies on the system disk, and the */usr* hierarchy on another disk. With clusters, it is essential to put */tmp* on a non-system disk. Moving the */tmp* directory to yet another disk is also helpful. This practice may not be optimal for all sites since it is more difficult to keep track of software modules which are kept in a number of subdirectories on several disks. If disk quotas are enabled the directories where users write temporary files (*/usr* and */tmp*) should be on a disk with available quotas.

The INSTALL.COM file which reads the product tape will ask you for the names of the disks upon which the different directories should be placed, so it is a good idea to determine the file locations before beginning the install procedure. Use the estimated hierarchy sizes shown in Table 2-4 as a basis for your calculations.

Figure 2-1 shows how EUNICE BSD software might be placed onto two disks and also shows what the UNIX appearance of the VMS hierarchical directory structure would be from the viewpoint of the EUNICE BSD run-time environment. Section 4 describes, in detail, how to create a UNIX directory hierarchy in the VMS environment. (See also Section 10 on changing existing EUNICE BSD layouts.)



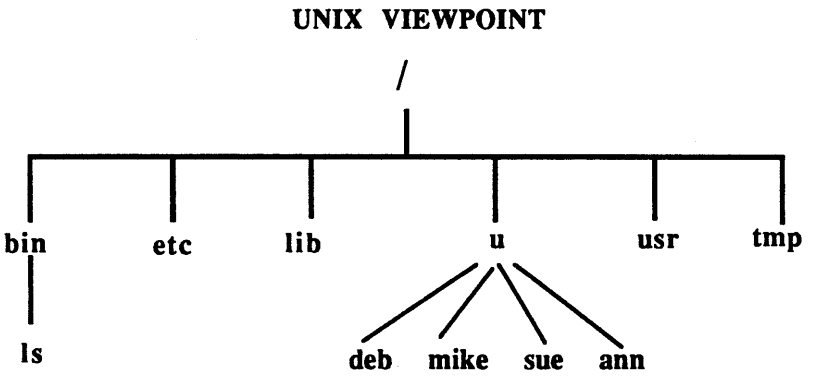


Figure 2-1. EUNICE BSD Run-Time Environment

3. MOUNTING, LOADING AND READING THE DISTRIBUTION TAPE

The first step is to use VMS BACKUP to read in the installation command file. The first save-set on the tape (INSTALL1.BCK) contains the file INSTALL.COM, which will automatically read in the rest of the tape. The instructions below put this command file into SYS$SCRATCH (which, when the installation procedure is begun, should be pointing to SYS$MANAGER).

In the examples below, it is presumed that your tape drive is called MUA0:; change the command lines as appropriate to your site. If your system is *not* a MicroVAX, the drive density **must** be set to 1600 bpi.

Depending upon the type of VAX you are using, use the appropriate "MOUNT" command, as shown.

```
USERNAME: SYSTEM
PASSWORD:  <<enter system password>>

$ SET DEFAULT SYS$SCRATCH
```

For MicroVAX systems type:

```
$ MOUNT/FOREIGN MUA0:
```

For all other VAX systems type:

```
$ MOUNT/FOREIGN/DENSITY=1600 MUA0:

$ BACKUP/LOG MUA0:INSTALL1.BCK SYS$SCRATCH:
```

3.1 RUNNING THE INSTALLATION COMMAND FILE

When INSTALL.COM runs, it performs a series of installation and system-checking steps. Please read the following summary of its functions now so that it will be easier for you to follow its activities as they occur.

```
$ SET NOVERIFY
$ @INSTALL.COM
```

After printing a brief summary of the instructions, INSTALL.COM will ask you for the name of your tape drive. When entering this name **make sure that you include the colon!** (i.e. Enter "MUA0:" rather than "MUA0".) If you forget this character, the command file will generate bad device specifications and the procedure will fail.

Next, INSTALL.COM will ask if you would like to have a listing of the tape-reading session. (This adds or leaves off the /LOG option on the BACKUP command.) It is always possible to use the 'Control-O' character to stop and restart the output log even if you have selected the /LOG option.

Next, INSTALL.COM asks questions about disks upon which the different directories are to be placed.

LOOK AT TABLE 2-1 CAREFULLY. THESE ARE THE FILES BEING EXTRACTED FROM THE TAPE. VERIFY THE LOCATION ON THE DISK WHERE YOU WANT EACH OF THESE TO RESIDE.

If the EUNICE BSD directories are to be spread over more than one disk, note that all disks which have EUNICE BSD user directories or EUNICE/UNIX software should have W:E protection at the top of the disk hierarchy.

The EUNICE BSD hierarchy can be automatically divided across a maximum of three disks. Any further partitioning will have to be done later by hand. The most common configuration is to place all three save sets on the same disk - usually the system disk. In the event that it is decided to split the hierarchy up, the ROOT.BCK save set is typically placed upon the system disk, with TMP.BCK and USR.BCK placed on a separate disk.

In the case of a clustered system, the executables may be shared, but each licensed CPU will have to boot its own EUNICE BSD *kernel.*

Once you have answered these questions, the installation procedure will first go to the SYS$MANAGER directory and create (or check for) the [.EUNINSTALL] sub-directory, where it will copy the setup procedures described in the following paragraphs.

NOTE: If you exit the INSTALL.COM procedure during its normal run, you will find that the SYS$SCRATCH logical has been temporarily reassigned to point to the [.EUNINSTALL] directory. When you subsequently restart INSTALL.COM, do not run the one in the [.EUNINSTALL] sub-directory, as you will get errors when it attempts to overwrite the operational file. Instead, use the one that you have created in SYS$MANAGER.

NOTE: Once the INSTALL.COM command file has been run successfully, there will be a [.EUNINSTALL] directory under SYS$SCRATCH, which contains a duplicate copy of the INSTALL.COM file; the copy of INSTALL.COM in SYS$SCRATCH may be deleted once it has been run. If INSTALLCOM is needed again, use the copy in [.EUNINSTALL], which also contains other setup files. Do, however, move the INSTALL.COM file out of [.EUNINSTALL] if you wish to run it; otherwise it will fail when it attempts to read the INSTALL2.BCK save-set.

3.1.1 INSTALLATION ENVIRONMENT SETUP

The ILOGIN.COM file is used to set up the environment each time the installer logs in to do a portion of the installation. It should ALWAYS be run by the system administrator prior to "touching" files contained within the [EUNICE...] hierarchy. This avoids the common problems associated with protections being incorrectly set on EUNICE BSD files. It also sets up the EUNICE BSD installation directory as SYS$SCRATCH. (If you do not want to use this feature after the installation has been completed, then the logical assignment line may be commented out.) The ILOGIN.COM file executes the following three commands:

```
$ SET  PROCESS/PRIVILEGE = (ALL,NOBYPASS)
$ SET  PROTECTION = (S:RWED,O:RWED,G:RE,W:RE) /DEFAULT
$ ASSIGN/JOB  SYS$SYSROOT:[SYSMGR.EUNINSTALL]  SYS$SCRATCH
```

INSTALL.COM will execute these commands for you before it reads in the distribution tape.

3.1.2 CHECKING FOR FILE NAME COLLISION (EUNFN.COM)

The next step in the installation process is to confirm that the name of the shareable library used by EUNICE BSD run-time environment files does not collide with any existing files, and that there are no directories named EUNICE.DIR on any of the disks that you have specified as targets for the EUNICE BSD hierarchy.

Table 3-1 lists those files which must be verified. A message indicating "No files found" should appear as the INSTALL.COM procedure checks each of the four file names referenced in EUNFN.COM. If a shareable library file is found, delete or re-name it. If there are old EUNICE BSD trees, rename them.

Table 3-1. EUNICE BSD Run-Time Environment Files
(Stored Outside of EUNICE BSD Directories)

FILE ENTRY	DESCRIPTION
SYS$SHARE:TWG_LIBC_43.EXE	The C language run-time library.
<ROOT.BCK destination device>:EUNICE.DIR	/bin, /etc, /lib, /vos
<TMP.BCK destination device>:EUNICE.DIR	/tmp
<USR.BCK destination device>:EUNICE.DIR	/usr

3.1.3 CHECKING FOR LOGICAL NAME COLLISION (EUNLN.COM)

The next step in the verification process is to confirm that VMS logical names used by EUNICE BSD do not collide with existing VMS logical names. There should be a message indicating that no logical name has been found as INSTALL.COM checks each of the logical names referenced in EUNLN.COM. Logical names are checked on system, group, job and process levels from the system account.

Table 3-2 lists the logical names which are checked and a description of each logical name. The message "No translation for logical name" should occur for each logical name checked. If a logical name is displayed on the terminal, a logical name collision problem exists. Re-name the existing VMS logical name or, if this is a problem for your site, telephone Wollongong.

Sites which are currently running Wollongong's WIN/TCP communications software will find that certain logical name collisions occur, mainly in areas surrounding the "REX™" kernel (the EUNICE BSD foundation upon which the WIN/TCP product is based). Therefore, once the collision-checking code in INSTALL.COM inquires as to whether you have observed any conflicts, answer affirmatively. When INSTALL.COM terminates, edit SYS$MANAGER:SYSTARTUP.COM to remove the call to STARTINET.COM. Next, reboot the system, and restart the EUNICE BSD INSTALL.COM procedure. This time there should be no conflict presented by the WIN/TCP logicals.

Table 3-2. Logical Names

LOGICAL NAME	DESCRIPTION
CONSOLE	*/dev*-name for system console.
CUA0	*/dev/cua0* - The *uucp(1)* A.C.U. device.
DST_FLAG	Daylight Saving Time Flag.
EUN_ROOT	VMS rooted directory specification pointing at location of */bin, /etc, /lib, /vos.*
EUNICE_DUMMY_IMAGE	Location of EUNICE run-time support files.
EUNICE_DUMMY_IMAGE1	Location of EUNICE run-time support files.
EUNICE_VFORKCLI	EUNICE run-time support file.
EUNICE_VFORK_IMAGE	EUNICE run-time support file.
EUNICE_1VERSION	Advice to EUNICE run-time regarding the creation of multiple versions of files.
GMT_DSP	Coercions for UNIX time.
MTn	*/dev*-name for tape drives (n = number of device).
TTYPn	*/dev*-name for WIN/TCP virtual login devices (n = number of device). (A separate product from Wollongong.)
NULL	*/dev*-name for the null device.

Table 3-2. Logical Names (Continued)

LOGICAL NAME	DESCRIPTION
PRINTER	*/dev*-name for line printer.
RMTn	*/dev* alternate (raw) name for tape drive. (n = number of device).
RTYn	*/dev*-name for DECnet™ remote terminal (n = number of device).
TERM	Default *termcap(5)* terminal type for your system.
TTY	*/dev*-name for user's login terminal.
TTYn	*/dev*-name for each asynchronous communication device (n = number of device).
TWG$ADMIN	Logical which takes you to the */etc/eunice* directory.
TWG$ATQUEUE	Logical which indicates queue to use for *at(1)* jobs.
TWG$BIN	Location of UNIX */bin* directory.
TWG$ETC	Location of UNIX */etc* directory.
TWG$EUNICE	EUNICE version number - also used by other Wollongong product command files.
TWG$LIB	Location of UNIX */lib* directory.
TWG$SYS	EUNICE directory for system disk.
TWG$TMP	Location of UNIX */tmp* directory.
TWG$U	Usual location of user (*/u*) directory.
TWG$USR	Location of UNIX */usr* directory.
UNIX_ASSEMBLER_CASE_CONVERT	UNIX Assembler case flag.
UUCP_HOST_NAME	UNIX *uucp* system name.

Having checked these items, the installation procedure gives you the option of stopping its execution if any conflicts occured. If there were none, the procedure goes on to read in the tape.

3.1.4 READING THE BASE SAVE SETS

The distribution save-sets INSTALL2.BCK (5 files), LIBCSHARE.BCK (1 file), ROOT.BCK (124 files), and TMP.BCK (1 file) transfer to disk fairly quickly.

The USR.BCK save-set (the 4.3 BSD */usr* file system) may take anywhere from 20 to 45 minutes (or more), depending on the type of tape drive used and the level of activity on the system.

> **NOTE: Without the terminal I/O caused by using the /LOG flag (with BACKUP), the files will be read in faster. Alternatively,** *CTRL-O* can be used to turn off/on the LOG information without stopping the BACKUP.

3.1.5 DOCUMENTS, LEARN, AND THE SOURCE CODE CONTROL SYSTEM

With this release, installation of the on-line documentation is optional. Since the manuals take up so much space, many users have asked that their inclusion be left up to the system administrator. The installation command file will ask whether the individual save sets which contain the manual pages or the supplementary documentation should be read in or not. Please see Table 2-4 for the sizes of these save-sets to determine whether or not you have sufficient room on your system to install the documentation.

Should you, at some later time, desire to read one of these save-sets off the tape, simply use the same command line which is used in the INSTALL.COM command file. Examine INSTALL.COM, and find the command line that relates to the save-set that you need. Then, execute that command directly - USING THE SAME SYNTAX - and the files will automatically be added to the correct directories in the hierarchy. If you are in any doubt about this, call Wollongong Support for assistance.

As with the documentation, installation of the *learn(1)* command (and all its attendant library files) has also been made optional. INSTALL.COM will ask for instructions on whether or not to read in this save set.

Finally, if your site has a System V.2 or greater UNIX license, you will have one last save set on the tape - SCCS.BCK. This contains the latest AT&T System V UNIX version of the Source Code Control System software.

3.2 EUNICE BSD SOFTWARE DISTRIBUTION SOURCES

When you order EUNICE BSD source, you receive an additional tape that contains source code for 4.3 BSD UNIX. Some of the BSD code has been modified to work with EUNICE BSD. Source code for the modules that contain proprietary Wollongong technology is not included on tape. Therefore, the entire EUNICE BSD product cannot be created from this source tape.

DISK SPACE REQUIREMENTS

You need about 60K blocks of disk space for EUNICE BSD source, and about 1.1K blocks of disk space for SCCS source.

STARTING DIRECTORY

The VAX operating system is limited to eight levels of directory hierarchy. The *device* is usually designated as the [000000] level. This leaves only seven levels of usable hierarchy. We recommend that you start your source tape at *<drive>*:[000000.USR.SRC].

REDEFINE SYS$OUTPUT

In order to speed the source backup procedure, and to save error messages, we recommend that you redefine your SYS$OUTPUT. Backup messages are sent both to SYS$OUTPUT and SYS$ERROR. Therefore, messages will be sent to your terminal, even though SYS$OUTPUT is redefined.

PUTTING THE SOURCE ON DISK

Follow these instructions to place the sources on disk.

Customers with MicroVAX and VAX systems may use the following commands:

```
$ MOUNT/FOREIGN <tape device>
$ DEFINE SYS$OUTPUT LOGFILE.LIS
$ BACKUP  <tape device>:EUN_SRC.BCK/SAVE_SET  <drive>:[USR.SRC...] -
   /LOG/REW
```

If you have ordered SCCS, use the following commands:

```
$ BACKUP  <tape device>:SCCS_SRC.BCK/SAVE_SET  <drive>:[USR.SRC...] -
   /LOG/REW

$DEASSIGN SYS$OUTPUT
$DISMOUNT <tape device>
```

4. ADAPTING THE EUNICE BSD ENVIRONMENT

Once the EUNICE BSD files are present on your VAX, EUNICE BSD can be configured to reflect site-specific requirements. Because each computer may have different devices and device names, EUNICE BSD is shipped with a "template" version of the various files which must be adjusted for each specific hardware installation.

4.1 CREATING ASSIGNMENTS FOR UNIX DIRECTORIES

Before configuring the EUNICE BSD environment, use ILOGIN.COM (from INSTALL2.BCK) to set up the interactive environment. If you have not logged out since you read in the distribution tape, the SYS$SCRATCH logical should still be pointing to the location of this command file (SYS$SYSROOT:[SYSMGR.EUNINSTALL]).

Next, create the VMS logical name EUN_ROOT (this is a system-wide logical name used by EUNICE BSD to keep track of the "kernel" files) and the "TWG$ADMIN" logical name (which points to the */etc/eunice* EUNICE BSD setup directory), as indicated below.

```
$ SET DEF DUA0:[EUNICE.ETC]
$ @SYS$SCRATCH:ILOGIN.COM
$ ASSIGN/SYSTEM/EXEC/TRANS=CONC DUA0:[EUNICE.] EUN_ROOT
$ ASSIGN/SYSTEM DUA0:[EUNICE.ETC.EUNICE] TWG$ADMIN
```

The TWG$ADMIN directory is the repository for all EUNICE BSD setup-related files. In past releases, these files have been placed directly in the */etc* directory. Placing them in this sub-directory leaves */etc* looking more UNIX-like, and keeps these command files out of the way of users.

Having set up the interactive installation environment, the next step is to modify the EUNICE BSD initialization files.

4.2 EDITING THE EUNICE BSD STARTUP PROCEDURE (STARTEUNICE.COM)

The same assignments you made above must now be added to the TWG$ADMIN:STARTEUNICE.COM file so that they will be set up whenever EUNICE BSD restarts automatically after a system re-boot.

The STARTEUNICE.COM command file calls all of the other sub-procedures which are run at system startup to provide the necessary EUNICE BSD coercions described in the introduction to this manual. Set your default directory to TWG$ADMIN, and edit the STARTEUNICE.COM file to reflect the two logical names that you defined in Section 4.1 above. To edit the STARTEUNICE.COM file, do the following:

```
$ SET DEF TWG$ADMIN
$ EDIT STARTEUNICE.COM
```

NOTE: It is assumed that the installer is familiar with the editors available under VMS.

The only other changes that should be made to this file concern the calls to UUCP.COM and LPR.COM. By default, these are commented out, since few sites require their use. If you are planning to bring up *uucp*, remove the exclamation mark in front of the call to the command file "UUCP.COM"

The printer queue, as set up under VMS, should work perfectly well with EUNICE BSD, since EUNICE writes to the VMS printer. In the (extremely) unlikely event that your *lpr* command does not spool correctly to the VMS printer, the LPR.COM file may be edited and enabled. Wollongong recommends, however, that you ensure that your VMS printer queues and devices are all correctly configured before resorting to this measure, since failure usually indicates problems at the VMS level.

For the first installation of EUNICE BSD, you will be running the individual startup procedures by hand. However, once all these files have been tailored to your site's requirements, running STARTEUNICE.COM will bring the whole system up at once by invoking all of the command files in succession. Therefore, after tailoring STARTEUNICE.COM for your site, do NOT run it until the system has been rebooted.

4.3 SETTING UP THE USER-LIMIT GLOBAL SECTION (USERS.COM)

The USERS.COM command file is the first file to be called by STARTEUNICE.COM. (Previous users of EUNICE BSD should note that it is no longer necessary to edit this file manually.) USERS.COM calls a program in */etc/eunice* which sets up a file which keeps track of the number of users currently accessing the EUNICE BSD system. If, for example, you have a 16-user UNIX license, it will limit the number of users to 16. The executable that sets up the global section which is correct for your license level is automatically included and will announce, at run-time, the number of users that are permitted to access EUNICE BSD.

Since you are bringing up the system by hand, run this file manually:

```
$ @TWG$ADMIN:USERS.COM
  <<Created g-sect(EUNUSERS) size nnbytes nn users>>
```

4.4 BUILDING THE ROOT DIRECTORY (ROOT.COM)

There are two methods of translating UNIX path names into VMS directory specifications. If the directory specification consists of a device name, then the top directory of the device ([000000]) is used. If the specification consists of both a device and a directory name, then the specified directory (with its subdirectories) is used. Examples are given in Table 4-1.

Table 4-1. Example Directory Specification

Logical Name	Translation	Unix.File	Equivalent VMS File
TWG$ETC	DUA0:[EUNICE.ETC.]	/etc	DUA0:[EUNICE]ETC.DIR
		/etc/a	DUA0:[EUNICE.ETC]A.DIR
		/etc/a/b.c	DUA0:[EUNICE.ETC.A]B.C
TWG$U	DUA0:	/u	DUA0:[0,0]
		/u/account	DUA0:[ACCOUNT]

The commands in the ROOT.COM file set up these logical names. The lines in the current version of ROOT.COM show how to set up the file systems for a single disk drive configuration which contains both the EUNICE BSD files and the user directories. If the various directories have been placed in other than the specified locations, those lines must be modified as necessary. The lines referring to */bin, /etc, /lib, /usr* and */tmp* are necessary in order to emulate a UNIX environment on EUNICE BSD. It is **strongly recommended** that the logical name */u* be assigned to the user disk. If there is only one disk, it is best to assign the logical TWG$U to the directory containing the users' home directories.

A major change in the new EUNICE BSD release is the manner in which the "/" (or UNIX "root") directory is emulated. What, in past, was done with two extra files is now accomplished by means of a VMS Search List named "/". This logical names all the directories that reside in the root directory. Additionally, there has to be an equivalent logical commencing with the prefix "TWG$" for each of these directories.

In other words, for every directory under VMS that you wish to access directly from "/" in EUNICE BSD:

1. Create a logical name which starts with "TWG$" in ROOT.COM.
2. Add the name to the search list at the end of ROOT.COM.

Suppose you had a directory under VMS called DUA1:[DATABASES.PROJECT2]. Since this directory is often accessed by people in your EUNICE BSD group, you decide that you want it at the top of the UNIX hierarchy. You decide to call the directory "project2". Perform the two steps indicated in the following example:

```
$ EDIT  ROOT.COM
```

1. Add this line to the ROOT.COM file:

```
$ ASS/SYS/EXEC/TRANS=CONC DUA1:[DATABASES.PROJECT2] TWG$PROJECT2
```

2. To the existing root search list add the name of the directory as you wish to access it:

```
$ define/exec/sys  "/"  "bin","dev","etc","lib","tmp","usr",-
  "sys","u","project2"
```

Now, when you wish to access this directory from EUNICE BSD, simply type *cd /project2* - and you will be in the DUA1:[DATABASES.PROJECT2] directory.

> **NOTE: Although there is a search list entry for *dev* in this file, no logical name is required.**

After editing each of the lines as applicable to your site, execute ROOT.COM. It is usually a good idea to set VERIFY in DCL so that you can see the progress of the procedure.

```
$ SET VERIFY
$ @TWG$ADMIN:ROOT.COM
$ SET NOVERIFY
```

> **NOTE: It is very important to check that the logical names all have matching search list entries (and *vice versa*). If there are incompatibilities between the two lists (except */dev*), run-time messages like *Image Activation Error* will be generated.**

4.5 BUILDING THE DEVICE DIRECTORY (DEV.COM)

The next set of logical name assignments are used to emulate the remaining (non-disk) devices; they are the contents of the */dev* directory. At every reference from EUNICE BSD to */dev/<device>*, EUNICE BSD will look for the logical name <device> and use its translation or the VMS device name.

Notice how logical names are used to assign UNIX *tty* names to VMS terminal ports in Table 4-2.

Table 4-2. Explanation of */dev* Logical Name Translations

VMS Device Name (translation)	VMS Logical Name (for EUNICE)	UNIX Path name	Description
NLA0:	NULL	*/dev/null*	Null device
OPA0:	CONSOLE	*/dev/console*	Console terminal
MTA0:	MT0	*/dev/mt0*	Tape drive
MTA0:	RMT0	*/dev/rmt0*	Raw mode tape drive
LPA0:	PRINTER	*/dev/printer*	Printer
RTA1:	RTY1	*/dev/rty1*	DECnet remote terminal
RTA2:	RTY2	*/dev/rty2*	DECnet remote terminal
TT:	TTY	*/dev/tty*	Process control terminal
TXA0:	TTY0	*/dev/tty0*	EUNICE tty-name translation
TXA3:	TTY3	*/dev/tty3*	EUNICE tty-name translation
TXB2:	TTY10	*/dev/tty10*	EUNICE tty-name translation
TXC2:	TTY18	*/dev/tty18*	EUNICE tty-name translation
NTY0:	TTYP0	*/dev/ttyp0*	TCP/IP pseudo-terminal

You will need to edit DEV.COM to identify those devices which are to be accessible from the UNIX environment. Back at the start of the installation preparation, you made a hard-copy listing of the devices on your VAX. Take this out and check off the ones that will have to be accessed from EUNICE BSD. Beside each of the VMS devices write in their UNIX names, so that you will be sure to have them all identified. Once you have identified all the devices which will be accessed from the UNIX environment, proceed to edit the DEV.COM file.

The DEV.COM file works on exactly the same principle as ROOT.COM. Every device will have a logical name assigned to it which will be the name by which it is known to EUNICE BSD. The "*/dev*" directory is (as is the "/" directory) emulated by means of a search list - this one being called "*/dev*". After you have assigned all the logical names in the first part of DEV.COM, make sure that the search list in the second part reflects EVERY device for which you have created a logical name. Devices which should be included in the DEV.COM file include all the terminal lines, tape drives, the preferred line printer, and graphic display units.

While the TXnn:-type devices specified in Table 4.2 (on the previous page) require actual logical names to be associated with each physical device present, there are some devices which EUNICE BSD handles internally. The main reason for this, apart from the obvious convenience, is that the system's logical name tables will become far less crowded this way. The devices so handled are VTAn:- and LTAn:-type virtual terminals, and regular TTAn:-type hard-wired terminals.

Most systems have at least some TTAn: or TXAn:-type hard-wired terminal devices. The following three scenarios show different configurations that may come up, and the ways in which the logical assignments are affected.

Scenario #1:

If your system supports only TTAn:-style terminals, then there is no need for you to set up any device logical name translations. Simply count the total number of terminals that you have and assign that number of *tty*-devices inside the "*/dev*" logical.

For example, a system having a range of TT-devices from TTA0: to TTD7: has 32 different terminals on which people might log in. This would require that the "*/dev*" search list have definitions for devices *tty0* to *tty31*.

Scenario #2:

If you have a system that supports only TXAn:-style terminals, then you will have to define a logical for every TX-terminal that exists. There are example lines in the DEV.COM file that may be used if this is your situation.

For example, a system having a range of TX-devices from TXA0: to TXB7: will need 15 logical assignment lines. In addition, the "*/dev*" search list will have to be set up to expect the UNIX devices *tty0* to *tty15*.

Scenario #3:

If your system has a **combination** of TX- and TT-devices, then you will have to be more careful with the definitions. The TT-terminals take precedence (being internally defined), and will be automatically translated to appropriate *tty*-names. In order to handle the TX-terminals correctly, it is necessary to first figure out the total number of TT-devices that you have. Having determined

this, start defining individual logicals for each of the TX-terminals - starting at the NEXT available *tty*-number.

For example, if the system had devices TTA0: to TTC7: (24 terminals), and TXA0: to TXD7: (32 terminals), then the logical assignment lines in DEV.COM would read:

```
$ ASSIGN/SYSTEM/EXEC TXA0:     TTY24          !/dev/tty24
                    .
                    .
                    .
$ ASSIGN/SYSTEM/EXEC TXD7:     TTY55          !/dev/tty55
```

In this case, the "*/dev*" logical will have definitions for *tty0* to *tty55*. Anyone who logs in on a TT-type terminal will automatically have the appropriate *tty*-number assigned to his login, based on the TT-device number. The TX-device names will be resolved with the aid of the logicals which have been set up in DEV.COM.

If you have enabled virtual terminals on your system using VTA-type devices, then EUNICE BSD will check what the REAL device name is and use that. So, if one logs in on VTA115:, but the real port is TTA6:, then EUNICE BSD will assume that the login was on */dev/tty6*; this translation is made internally. If, however, your real device for VTA115: actually translates to TXA4:, then, unless you have defined a set of logicals for each of the TX-ports, you will have an undefined terminal. (i.e.: Issuing the *tty(1)* command from EUNICE BSD will return a login device-name of "tty??".) Therefore, even when using VTA-type terminals, be sure to follow the setup instructions detailed above in whichever of the three scenarios provided above is most applicable to your site. Remember to also perform a suitable adjustment of the "*/dev*" logical.

Finally, your system may be running a LAT terminal server. If this is the case, you have nothing to add to the setup. If your system uses only LTA-type logins, then you may leave the terminal section of DEV.COM alone. If your site combines hard-wired terminals with LTAs, then set up the hard-wired terminals as detailed above; allow EUNICE BSD to take care of the LTA-devices by itself.

Be sure to notice how many of the device logicals in the search lists are set by using ranges of numbers, e.g. "rty[0-9]". This short-hand notation can be helpful to you when you have large numbers of devices.

Taking into account the scenarios described above, edit DEV.COM by typing:

```
$ EDIT  DEV.COM
```

After editing it, execute DEV.COM.

```
$ SET  VERIFY
$ @TWG$ADMIN:DEV.COM
    <<multiple system responses>>
$ SET  NOVERIFY
```

4.5.1 Building the terminal ID files

Having set up the device directory, it is now necessary to tailor the terminal identification and location files which are located in the parent directory - TWG$ETC:[000000].

Some EUNICE BSD utilities are screen-oriented and require identification of the terminals on which the utilities are being run. Once the files which contain the terminal definitions and locations have been established, EUNICE BSD can access the information automatically. Edit */etc/ttys*, and */etc/locations* as below, changing directories using the VMS [-] convention to get there. (Templates have been provided for both of these files.)

To edit */etc/ttys*, type the following:

```
$ SET DEFAULT [-]
$ EDIT TTYS.
```

The following is an example of the */etc/ttys* file:

```
console none su on
tty0 none vt100 on
tty1 none vt131 on
tty2 none vt100 on
tty3 none vt102 on
tty4 none vt100 on
.
.
.
tty31 none vt131 on
```

NOTE: Be sure not to leave any blank lines, tabs, or blanks at the end of the line in this file.

The */etc/ttys* file must have an entry for each login line on the system, including *nty's* or *ttyp's* for Wollongong WIN/TCP logins and *rty's* for DECnet virtual logins. When the login program is run at login time, it reads the */etc/ttys* and sets the TERM environment variable accordingly.

For a complete description of the *ttys* file, please see *ttys(5)*, the *UNIX Programmer's Reference Manual [PRM]*, Section 5.

To edit */etc/locations*, type the following:

$ EDIT LOCATIONS. The following is an example of the locations file:

```
opa0:,VAX Console.
tta1:,Vt100 terminal in Computer Room.
tta2:,Room 286.
tta3:,President's office
.
.
```

txd7:,Classroom 5, Terminal # 20.

The */etc/locations* file contains terminal/location information for programs that need to know the location of a particular terminal. Edit the sample file to suit your particular configuration. Note that this particular file uses VMS device names rather than the equivalent UNIX names. Also note that this file is not critical to the success of your installation. If the information about terminal locations is not readily available, simply edit the file to reflect the VMS names of your devices, and add the location information as it becomes known.

4.6 SET UP THE EUNICE BSD RUN-TIME ENVIRONMENT (EUNICE.COM)

The next file to edit is the TWG$ETC:[EUNICE]EUNICE.COM, file which sets up the actual run-time environment that makes EUNICE BSD behave like UNIX. Only a few lines need to be changed in this file, which have all been flagged with a comment field that states "Site dependent". These site dependent lines are the only ones that will be addressed in this manual. Users who are curious about the remainder of the EUNICE.COM file's functions may read the comments contained therein for explanations.

```
$ SET DEFAULT TWG$ADMIN
$ EDIT EUNICE.COM
```

The EUNICE.COM file sets up the system-wide logical name, GMT_DSP, required to convert local time to Greenwich Mean Time (GMT), which is the time base used by UNIX. The GMT_DSP file shipped with EUNICE BSD is set for California, which is 8 hours west of GMT. (The East Coast of the U.S.A. is 5 hours west of GMT.) Modify this line (if necessary) to reflect the displacement west of GMT for your site. If the displacement is east of GMT, then use the negative of that displacement.

```
$ DEFINE/SYSTEM GMT_DSP "0 08:00:00.0" ! Eight hours west of GMT
        or:
$ DEFINE/SYSTEM GMT_DSP "0-01:00:00.0" ! One hour east of GMT
```

The next logical to be set in the EUNICE.COM file determines whether or not Daylight Saving Time (DST) is in effect. (Remember that DST is changed TWICE a year.)

If DST is in effect, remove the exclamation mark from in front of the line which assigns the "DST_FLAG" logical. (Change "! DEFINE" to read "DEFINE", as shown below.)

```
$ DEFINE/SYSTEM  DST_FLAG  ON  ! DST in effect
```

The next user-dependent logical name which is set in the EUNICE.COM file is the environment variable "TERM". "TERM" is used to specify the type of terminal being used, such as "vt100". The "TERM" logical name is optional; it provides a default in case the user does not define the "TERM" environment variable during login initialization. Section 6, "Building a Sample User Account", describes how each user can select his or her own value for TERM. The character sequences required to perform a given operation on a terminal type specified by the "TERM" variable are contained in the file */etc/termcap*. The following shows how the default value for "TERM" is set in EUNICE.COM.

```
$ DEFINE/SYSTEM  TERM  "vt100"  ! vt100 by default
```

The next logical, UUCP_HOST_NAME, is required by a subsequent procedure - *mbox.csh* - to set up users' mailboxes. It is recommended that the system administrator decide upon a hostname for the system, so that if *uucp* is later used, the logical is already incorporated.

Simply edit the line in EUNICE.COM to replace the word "myhost" with the appropriate LOWERCASE name. (Remember that UNIX is case-sensitive.) For example:

```
$ DEFINE/SYSTEM  UUCP_HOST_NAME  "vax1"
```

NOTE: Never comment this line out! This logical must always be defined, even if the name is just left as *myhost*.

The logical name EUNICE_1VERSION is used to make file creation behave the same way as it would in the UNIX environment. In other words, *creat(2)* and *open(2)* do not create extra copies of files, as would happen in VMS (with new versions of files being created every time *create(2)* and *open(2)* are called). For a correct emulation of all UNIX utilities, EUNICE_1VERSION should be set to ON, as follows:

```
$ DEFINE/SYSTEM  EUNICE_1VERSION  "ON"
```

The EUNICE BSD run-time environment can now be initialized. To do this, turn on the VMS DCL verify mode and, if you have logged out between steps, re-set the default protection; then execute EUNICE.COM. Be sure to watch the system messages for errors and note the messages logged by INITGBL describing the various parameters which have been set up for the process database. If further modification is necessary, then edit EUNICE.COM and re-execute it. (When EUNICE.COM is being re-executed, there will be messages about logical name assignments being replaced.) When all modifications have been made correctly, turn off the VERIFY mode.

```
$ SET  VERIFY
$ @SYS$SCRATCH:ILOGIN.COM
$ @TWG$ADMIN:EUNICE.COM
      <<multiple system messages>>
$ SET  NOVERIFY
```

4.7 VERIFYING THE ADAPTATION OF THE EUNICE BSD ENVIRONMENT

To verify the adaptation of EUNICE BSD, perform a quick check on the environment which was built by going into the UNIX *csh(1)* (c-shell) and seeing whether the welcome message and the "%" prompt are printed. Do not try using any UNIX commands at this time -- this test is only to verify the basic structure of EUNICE BSD.

```
$ @TWG$ADMIN:CSHELL
    <<EUNICE message of the day>>
% logout
```

If EUNICE BSD has been initialized properly, you should have seen the "%" sign prompt on the screen when the shell was invoked.

5. INITIALIZING THE UNIX ENVIRONMENT

This section describes the emulation of the *setuid*, and saved text of *chmod(2)* UNIX facilities, the initialization of UNIX detached monitors (daemons), and the installation of optional UNIX software products. The DCL scripts described in this section have been set up to be used unmodified by a "typical" time-shared VAX computer.

5.1 INITIALIZING PROGRAMS WITH SPECIAL PRIVILEGE (SUCHMOD.COM)

The procedure TWG$ETC:[EUNICE]SUCHMOD.COM uses the VMS INSTALL utility to emulate the *setuid* and *sticky* protection mode bits of some UNIX programs. Normally, in a native UNIX environment, the system manager would use *chmod*(1) to turn on the *setuid* and *sticky* mode bits for the files containing the affected programs. *Setuid* bit is used to provide the ability to perform privileged operations while *sticky* is used to increase performance by keeping code in memory.

Before initializing the UNIX environment, you can review the list of UNIX programs in SUCHMOD.COM, and see the list of privileges and resources which are requested for each program. Wollongong has pre-configured these files to keep the most-used programs close to memory in order to increase EUNICE BSD performance. Certain files have been installed with privilege to make it possible for them to perform tasks that would otherwise require the USER to have privilege. Images are installed in VMS on a last-in/first-out basis, so the most frequently used commands should be installed last. An example of such an installed file is *csh(1)*, which can be installed into VMS memory to allow faster user response time.

If your site is licensed for SCCS, the command file will automatically install the *delta(1)* command with the privilege needed to overwrite a read-only file.

> **NOTE: An error message generated during the execution of the SUCHMOD.COM file usually indicates that the VMS SYSGEN parameters need to be adjusted to accommodate resource requests being made. For example, the KFILSTCNT parameter (refer to Section 2.4, "Checking VMS Sysgen Parameters") might be too low, and thus disallow the installation of all the programs mentioned in SUCHMOD.COM.**

If you have no other files to add to those indicated in the existing list, proceed to run the SUCHMOD.COM file, as follows:

```
$ SET VERIFY
$ @TWG$ADMIN:SUCHMOD.COM
$ SET NOVERIFY
```

5.2 SETTING UP THE UNIX USER DATABASE FILES (ADDUSER.COM)

Before building the UNIX administrative databases, set the default directory to TWG$ADMIN. Then, if you have logged out between any of these steps, use ILOGIN.COM to set up the default environment so that new files will have the protections necessary for proper access and execution. If you logged out, the SYS$SCRATCH logical name will have to be reset to point to SYS$MANAGER,

as indicated in the following example.

```
$ SET DEFAULT TWG$ADMIN
$ ASSIGN/JOB SYS$SYSROOT:[SYSMGR.EUNINSTALL] SYS$SCRATCH
$ @SYS$SCRATCH:ILOGIN.COM
```

You can now run the VMS procedure TWG$ADMIN:ADDUSER.COM which calls the shell script *adduser.csh* to first create */etc/passwd*, and then run */etc/eunice/mketcgrp* to create */etc/group*. Next, it sets up certain mail directories with */etc/eunice/mbox.csh*, and finally updates the */usr/lib/aliases.dir*.

If you have a cluster configuration, this procedure will query you for the VMS authorization file name if it cannot find SYS$SYSTEM:SYSUAF.DAT. You must then enter the VMS name for this file. An instance in which a different authorization file name might be supplied occurs when a site needs only a small group of users set up for EUNICE BSD. Temporarily moving or renaming the SYS$SYSTEM: SYSUAF.DAT file will permit the use of a separate authorization file.

An example of the output of *adduser.csh* follows.

```
$ @TWG$ADMIN:ADDUSER.COM

Adduser.csh is now running ...

Creating a new /etc/passwd file ...
      << Possible multiple system responses >>
Editing /etc/passwd ...
Setting ownership and protections on /etc/passwd ...
-rwx-r--r-- 1 system  3202 Jan 6 15:01 /etc/passwd

Creating a new /etc/group file ...
Editing /etc/group for VMS group names ...
Done editing; set ownerships and protections ...
-rw-r--r-- 1 system  1084 Jan 6 15:02 /etc/group

Generating new mail aliases ...
11 aliases: 459 bytes, longest 84 bytes, 0 errors

Adduser.csh has ended successfully!
```

5.3 CHECKING THE /etc/passwd FILE

Of the two files covered in this section, the more critical is the */etc/passwd* file. This file contains information about all the users on the system. The utility *cvtuaf(8)* uses the data in the VMS authorization file to generate an equivalent *passwd* file.

If *adduser.csh* generated any error messages during the execution of *cvtuaf* (in the "Possible multiple system responses" message in the previous example) the errors are usually the result of an inability to match a user's home directory specification in the SYS$SYSTEM:SYSUAF.DAT file with a logical in TWG$ADMIN:ROOT.COM. Verify that the entries in the */etc/passwd* file are correct. The initial working directories should all be valid UNIX path names. An example of a valid UNIX *passwd* entry is:

guest::65293:255:Guest Account:/u/guest:/bin/csh

where the colon-separated entries are:

A. The login name

B. Null (UNIX password location - not used in EUNICE BSD)

C. UserID equivalent (produced by the *cvtuaf* program)

D. GroupID equivalent (produced by the *cvtuaf* program)[1]

E. User's full name

F. Home directory path[2]

G. Default shell (always the cshell)

More information on */etc/passwd* is available in *passwd(5)* in the *UNIX Programmer's Reference Manual [PRM]*, Section 5.

If there are any lines in the */etc/passwd* database that contain a "//" in the sixth field (the user's home directory path) check */etc/root.com* and SYS$SYSTEM:SYSUAF.DAT to make sure that the logical (and the actual directory it defines) really exist. Also, make sure that the directory in question is represented in the "/" search list.

If the */etc/password* file is correct, but *cvtuaf* still refuses to resolve the directory specification, edit TWG$ADMIN:ADDUSER.CSH directly. This file has a section which uses *ex(1)* to edit the *passwd* file. It includes examples on how to modify *passwd* to provide appropriate home directories. There should be no need to change SYS$SYSTEM:SYSUAF.DAT. Now, re-run TWG$ADMIN:ADDUSER.COM as previously shown. If this utility is not run when a new user is added to the system, the new user will not be recognized by a number of UNIX commands.

5.4 INITIALIZING UNIX DAEMONS AND PUBLIC FILES (RC.COM)

The design of RC.COM is based upon */etc/rc*, which is used in native UNIX environments to start up daemons and remove accumulated temporary information in the public files and directories. The following paragraphs explain the various functions of RC.COM.

Some of the daemons have separate startup command files, such as LPR.COM to use the printer from EUNICE BSD, and UUCP.COM which starts the *uucp(1)* code. Review the RC.COM file to check its suitability to your situation, and edit RC.COM if necessary by typing the following:

$ EDIT RC.COM

1. The Group ID is the same as the Group portion of the VMS UIC. The User ID (which must be unique across the entire system) is generated by shifting the Group portion of the VMS UIC left by 8 bits and then adding the Member portion of the VMS UIC. (This same algorithm is used within the EUNICE BSD run-time system to generate UIDs.)

2. The VMS Default directory is converted to a UNIX directory specification which requires the "/" directory to be correctly set up.

5.4.1 Public Files And Directories

The first portion of RC.COM adjusts the permissions of special system databases and temporary directories. Section 5.6 gives explanations of these databases.

Next, RC.COM cleans up the temporary scratch directories used by the UNIX utilities.

5.4.2 The Batch Job Daemon

In the (extremely unlikely) event that your system does not have a batch queue started up at system boot time, lines are provided in RC.COM to start the batch queue for you. We recommend, however, that you start any VMS queue from the appropriate place - the system startup file, using approved DEC procedures. The assumption made is that the queue IS running by default. If in doubt, use the following command to check if there is such a queue:

```
$ SHOW QUEUE/BATCH/ALL
```

The batch queue is used by the UNIX *at(1)* command.) If you wish to start the queue from EUNICE BSD, simply activate the lines by removing the comment ("!") characters at the beginning of each of the two lines.

The UNIX utility *at(1)* utilizes a new environmental variable called "TWG$ATQUEUE" which makes it possible to use a logical assignment to specify the queue to which the *at(1)* will submit jobs. It is possible to use SYS$BATCH, another existing queue, or a new queue that might be set up specifically for *at(1)* utility. The VMS privilege "OPER" is required when setting up the queue.

To submit *at(1)* jobs to the default queue, the following specification is used:

```
$ ASSIGN/SYSTEM SYS$BATCH TWG$ATQUEUE
```

For example, to submit *at(1)* jobs to a queue called *slow*, the following specification is used:

```
$ ASSIGN/SYSTEM SLOW TWG$ATQUEUE
```

To set up a new queue to which *at(1)* jobs may be submitted, do the following:

```
$ START/QUEUE ATQUEUE
$ IF $STATUS THEN GO TO BATCH_DONE
$ INIT/QUEUE/BATCH/JOB_LIMIT=4 ATQUEUE
$ START/QUEUE ATQUEUE
$ BATCH_DONE
$ ASSIGN/SYSTEM ATQUEUE TWG$ATQUEUE
```

All of these options are listed in the EUNICE BSD command file RC.COM which is located in the */etc* directory. The system administrator should uncomment the proper command or sequence of commands.

5.4.3 The cron(8) Daemon

With this release of EUNICE BSD, the UNIX *cron(8)* daemon has been implemented. The system administrator who wants to implement this tool has to activate the *cron* startup lines at the bottom of the RC.COM file which calls the procedure CRON.COM which, in turn, contains the process startup information.

5.4.4 Execute RC.COM

When there are no more changes to be made to RC.COM, run the procedure as follows:

```
$ SET  VERIFY
$ @TWG$ADMIN:RC.COM
$ SET  NOVERIFY
```

5.5 MODIFYING THE /etc/motd FILE

The */etc/motd* file contains a default "Message of The Day" to EUNICE BSD users, which is printed when the */etc/login* program is invoked on entering the c-shell. The template for */etc/motd* which is provided includes details which show the user how to get copies of the UNIX environment files. This template can be modified later to state something appropriate for your organization. A copy of the */etc/motd* template is also contained in */usr/skel* so it will be available for future use. If a user does not wish to see the *motd* on login, this feature can be turned off by creating an empty *.hushlogin* file in the user's home directory.

5.6 LOGIN ACCOUNTING FILES

5.6.1 The /etc/utmp file

The */etc/utmp* file is used to keep track of who is logged into the system. The UNIX utilities, *who(1)* and *w(1)* reference this file. When a user logs in, an entry is made in this file; the logout procedure removes the entry. If a user logs out by killing his process, the */etc/utmp* file will still have an entry for this person. However, the next time that a system re-boot is performed, */etc/utmp* will be cleaned out by */etc/eunice/rc.com*, and the permissions changed back to READ and WRITE to the WORLD.

5.6.2 The /usr/adm/wtmp file

A record of all logins and logouts is kept in the */usr/adm/wtmp* file. */usr/adm/wtmp* will be restarted at system re-boot by */etc/eunice/rc.com* and the permissions changed to READ and WRITE to the WORLD.

5.6.3 The /usr/adm/lastlog file

The */usr/adm/lastlog* file keeps track of the terminal number and the date and time of each user's login session. This file needs to be cleaned out periodically and the permissions set to READ and WRITE to WORLD. RC.COM does **NOT** clean out this file.

Housekeeping on the */etc/utmp* and */usr/adm/wtmp* files is done by TWG$ADMIN:RC.COM when the system is rebooted.

6. BUILDING A SAMPLE USER ACCOUNT

In Section 2.3, the quotas and privileges necessary to run EUNICE BSD were discussed. Assuming that the users' accounts have now been created and set up, they must now be provided with the files required to run EUNICE BSD from their own environment. In this section, a sample EUNICE BSD user account will be set up which will serve two purposes:

1. The System Administrator will have a guide to follow when adding new EUNICE BSD users to the system.
2. The installer will have an unprivileged account with which to test the installation.

To build the sample user account, change your current default directory to SYS$SYSTEM: and run the AUTHORIZE utility and begin building the example User Account by entering information indicated in the following example. Note that the user "TWG" is on the top directory of DUA0: with a UIC of [177,200]. The EUNICE BSD logical referring to DUA0: is "TWG$U:".

```
$ SET DEF SYS$SYSTEM
$ RUN AUTHORIZE
UAF> ADD TWG /PASSWORD=TWG/UIC=[177,200]
%UAF-I-ADDMSG,user record successfully added
%UAF-I-RDBADDMSGU,identifier TWG value: [177,200] added to RIGHTSLIST.DAT
UAF> MODIFY TWG/DIR=[TWG] —
/DEV = TWG$U: —
/OWNER = "TWG" —
/PRIV = (NOALL,TMPMBX,NETMBX) —
/DEFPRIV = (NOALL,TMPMBX,NETMBX)
/BYTLM=60000/PRCLM=16/ASTLM=40/FILLM=100/TQELM=25
user record updated
UAF> EXIT
%UAF-I-DONEMSG, system authorization file modified
%UAF-I-RDBDONEMSG, rights database modified
```

Verify that the TWG account has **at least** the minimum value for each quota mentioned in Table 2-2. Then, go to the users' directory area and create a home directory called "TWG" for this user.

```
$ SET DEF TWG$U:[000000]
$ CRE/DIR [TWG]
$ SET FILE/OWN = [177,200] TWG.DIR
```

Remember to run TWG$ADMIN:ADDUSER.COM again. It should be run every time a new user (who wishes to access EUNICE BSD) is added to VMS. (Please see Section 5.2 for information on running ADDUSER.COM.)

6.1 AUTHORIZING THE USER ACCOUNTS GLOBALLY

All user accounts should meet the minimum quota requirements listed in Table 2-2. If appropriate, all accounts can be modified at one time, as indicated below:

```
$ SET  DEF  SYS$SYSTEM
$ RUN  AUTHORIZE
UAF > SHOW *
UAF > << modify as necessary >>
          For example:
          << MODIFY  */FILLM=100 >>
          or:
          << MODIFY  [*,*] /FILLM=100 >>
          user record(s) updated
UAF > SHOW *
UAF > EXIT
```

All users must have the TMPMBX privilege. While in AUTHORIZE, type the following:

```
UAF > SHOW *
UAF > MODIFY */PRIV=TMPMBX/DEFPRIV=TMPMBX
```

Beware, however, that modifying accounts in this way can reduce some users' auotas.

6.2 SETTING UP EUNICE BSD USER ACCOUNTS

6.2.1 User's Home Directory Files

After EUNICE BSD has been installed, certain files should be added to each user's home directory to provide EUNICE BSD with user-specific information. The directory */usr/skel* includes samples of the following files which are typically found in each user's home directory:

LOGIN.COM
.login
.cshrc
.exrc
.mailrc

The template message of the day (*/usr/skel/README*) provides the user with details on how to get copies of these files. If using the Bourne shell, the user will also need the *.profile* file, which is also found in */usr/skel.*

6.2.1.1 VMSLOGIN.COM Template for LOGIN.COM

VMSLOGIN.COM provides sample DCL foreign commands (symbols) for the first of these files, LOGIN.COM, which is executed upon logging onto the system. Existing users will need the following line added to their LOGIN.COM file in order to invoke EUNICE BSD automatically upon logging onto the system. Note, that the */usr/skel/vmslogin.com* file sets this up as a symbol which can be run by hand or from the LOGIN.COM file.

$ @TWG$ADMIN:CSHELL.COM

or for the Bourne shell:

$ @TWG$ADMIN:SHELL.COM

Another possibility is to invoke the TWG$USR:[SKEL]VMSLOGIN.COM file directly from the system-wide LOGIN.COM file. There are also sample symbols for UNIXIN and UNIXOUT set to EUNLOGIN and EUNLOGOUT, respectively, to enable the use of UNIX utilities from VMS.

If you are using a terminal other than the one specified by default in EUNICE.COM (usually "vt100") then you can set the logical up for your own terminal type. In your local LOGIN.COM file, just establish the valid name for the device as listed in the */etc/termcap* file, and then enter it in the LOGIN.COM file as follows:

$ DEFINE/JOB TERM "terminal name"

For example, if you have a Wyse-100 terminal, do a search in the *termcap* file for the string "wyse". It will yield the name "wyse" as the appropriate selection to obtain the correct terminal characteristics. Therefore, this line should be added to your LOGIN.COM file:

$ DEFINE/JOB TERM "wyse"

For further information, please see the manual page on *termcap(5)* in the *UNIX Programmer's Reference Manual [PRM]*, Section 5.

Note that if an installed image, e.g. *pwd(1)* is set up as a foreign command, the same file specification syntax must be used. Look at TWG$ADMIN:SUCHMOD.COM for the syntax used.

6.2.1.2 The .login file

The UNIX equivalent to the user's LOGIN.COM file is the *.login* file. This file is executed by the UNIX *cshell* at login time. Two template files, named *.login* and *login.csh* have been provided for this file in TWG$USR:[SKEL] or */usr/skel*.

The *login.csh* file has comments for almost every entry in the file. The *.login* file is stripped of all comments, and therefore runs far more quickly at login time. It is recommended that the new EUNICE BSD user first take a look at the *login.csh* file to understand its functions, and then copy across the *.login* file for run-time use.

Each user can copy this file as detailed in the template message of the day, */etc/motd* (message of the day). System administrators copying any of the template files should make sure that resultant file ownerships are correct.

Just as there are two versions of the *.login* file, so there are two versions of the *.cshrc*, *.exrc*, *.mailrc*, and *.profile* files. The ones ending in ".csh" (or ".sh") are in all cases the commented ones, while the others are the operational counterparts.

Remember that *.login* and other "." files can only be seen in UNIX listings with the *-a* option, i.e., *ls -al*.

The contents of the *.login* file are explained in Table 6-1.

Table 6-1. Contents of *.login* File

```
# A template .login file for EUNICE.
# SCCS_ID - "@(#)login.csh 1.2  (TWG) 87/07/08 "

# When using redirection (">"), cshell warns if file already exists:
set noclobber

# Prevent logout with ctrl-z "end-of-file" signals:
# set ignoreeof

setenv PATH .:/usr/ucb:/bin:/usr/bin:/usr/eun:/usr/local/bin

# Csh history will remember the last 30 events:
set history=30

# Remember the last 20 events for your next cshell login:
set savehist=20

# Set prompt to event number and csh prompt:
set prompt=' % '

# ALIASES.
#   Note:  which(1) only recognizes aliases set in .cshrc, so you may
#   wish to move some to that file. That can, however, slow performance.

# Compiler switches for cc(1) and f77(1):
# With Eunice you have your choice of assemblers and loaders. You
# may choose UNIX or VMS executable style. The default is UNIX style.
# You may switch to VMS style using the vmsobj alias and switch back to UNIX
# using the unixobj alias.
```

Table 6-1. Contents of *.login* File (Continued)

```
# Have compiler use UNIX assembler and loader:
alias unixobj 'unsetenv AS_IMAGE; unsetenv LD_IMAGE'

# - or have compiler use VMS assembler and loader:
alias vmsobj 'setenv AS_IMAGE /usr/eun/vmsas; setenv LD_IMAGE /usr/eun/vmsld'

# This will make cc(1) use the UNIX assembler and loader now:
# unixobj

# This can make cc(1) use the VMS assembler instead:
# vmsobj

# Allow use of VMS's EDT from the csh:
alias edt  'vms -t edit/edt'

# Allow use of the VMS mailer program from the Cshell:
alias vmail  'vms -t mail'

# Set up some other shorthand calls for frequently used commands:
alias cp cp -i
alias a alias
alias cl clear
alias h history
alias ll ls -l
alias lo logout
alias mv mv -i
alias pu 'vms purge &'
alias rew 'vms set mag/rew mt0:'
# alias rm /bin/rm -i
alias so source
alias sus suspend
```

6.2.1.3 The .cshrc file

The *.cshrc* file is run by the *csh(1)* program each time it is started, including any sub-process *cshells* which may be run. Additional *cshell* functions can be defined in this file.

Table 6-2 shows the contents of a sample *.cshrc* file. Read *csh(1)* in the *UNIX User's Reference Manual [URM]*, Section 1, and the article by William Joy in the *UNIX User's Supplementary Documents (USD:4)* for other ways of writing your *.login* and *.cshrc* files.

Table 6-2. Contents of the *.cshrc* File

```
# A template .cshrc file for EUNICE.
# SCCS_ID - "@(#)cshrc.csh 1.1  (TWG) 87/06/22 "
# your own version of this (and other) login files.

# Remember:  which(1) only recognizes aliases which are
# set in .cshrc; however, as this file gets larger, performance slows down.

# DO NOT REMOVE!  Allows a background task to notify the shell immediately.
set notify

# Set up a path for cd.
set cdpath=(~ /usr/eun /usr/local /usr/man)
```

6.2.1.4 The .exrc file

The *.exrc* file is used by the *vi(1)* and *ex(1)* editors to set up a minimal environment for these related editors.

For further information on the contents of a *.exrc* file, and on *vi(1)* and *ex(1)* in general, read William Joy's "*An Introduction to Display Editing With Vi*" in the *UNIX User's Supplementary Documents (USD:15 and USD:16).*

Exrc.csh (the template) has several options in it. The actual *.exrc* file provided has four commonly used options set up, with three of them toggled off.

6.2.1.5 The .mailrc file

The *.mailrc* file is used when one invokes UNIX mail. Individual users can copy this into their home directory if it is not already there.

Options in this file can be set or unset as desired. Please note that all comments in this file must have a BLANK after the pound-sign, (e.g., # this is a comment) if they are to be successfully disabled. For further information, see the "*Mail Reference Manual*" in the *UNIX User's Supplementary Documents (USD:7).*

See the *mailrc.csh* file for comments and explanations.

6.2.1.6 The .profile file

The *.profile* file is required if the user will be accessing the Bourne shell. It serves the equivalent purpose of the *.login* file in *cshell.*

Please note that the commented version of this particular file is called *profile.sh* - since it is read by the Bourne shell rather than the C-shell.

7. INSTALLING EUNICE BSD ON VAX CLUSTERS

All the files in EUNICE BSD can be shared on VAX clusters. To install EUNICE BSD on a cluster, the following steps need to be taken.

1. Install EUNICE BSD on one VAX in the cluster as though it was a standalone machine (using the standard installation procedure).
2. After installation, modify STARTEUNICE.COM to establish the name of the node upon which EUNICE BSD is being brought up, by adding the following line at the beginning of the file:

   $ NODE = F$GETSYI("NODENAME")

3. If device names vary between the nodes, this must be reflected in separate DEV_*NODE_NAME*.COM files, where *NODE_NAME* is the node-name of each node. For example:

   If you have two nodes named TARDIS and DALEK, you will have to create the two files DEV_TARDIS.COM and DEV_DALEK.COM.

   Each of these files should contain a list of devices appropriate to that node.

4. Include the following line so that the right DEV.COM file is called in STARTEUNICE.COM by replacing the line that says:

   $ @EUN_ROOT:[ETC.EUNICE]DEV.COM

   with

   $ @EUN_ROOT:[ETC.EUNICE]DEV_'NODE'.COM

5. If the two nodes are not equal (with different users and/or devices) then administrative files cannot be shared between nodes. This will mean that administrative files have to retain their position in the EUNICE BSD tree, yet remain different. This is achieved by symbolic links. (For an explanation, see the *UNIX User's Reference Manual [URM], ln(1)*). All the administrative specific files are put under a separate tree for each node.

   In this and the following examples, the two nodes are named TARDIS and DALEK. Please replace these names with the node names used on your cluster. For example, create two directories, as follows:

   ```
   $ CREAT/DIR/PROT=(G:RE,W:RE) EUN_ROOT:[LOCAL_TARDIS]
   $ CREAT/DIR/PROT=(G:RE,W:RE) EUN_ROOT:[LOCAL_DALEK]
   ```

6. Add to the ROOT.COM file the following line to include the path */eunlocal* to the root path.

   ```
   $ ASS/SYS/EXEC/TRANS=CONC -
     EUN_ROOT:[LOCAL_'F$GETSYI("NODENAME")'.] TWG$EUNLOCAL  ! /eunlocal
   ```

Remember that this is a rooted logical name just like the other logicals in ROOT.COM.

7. Add this to the line where the logical "/" is set up

```
$ define/exec/sys "/" "bin","dev",....."u2","eunlocal"
```

8. Run the ROOT.COM file to make these logicals effective:

```
$ @ROOT.COM
```

9. The following files have to become symbolic links, as they cannot be shared:

```
/etc/passwd (only if users are different across nodes)
/etc/utmp
/usr/adm/*
/usr/spool/* (only if users are different across nodes)
```

10. Set up your default protection modes:

```
$ SET PROT=(G:RE,W:RE)/DEF
```

11. The following steps assume that you are bringing up EUNICE BSD on the node TARDIS. Copy the following files to the various nodes' local directories.

```
$ CREAT/DIR/PROT=(G:RE,W:RE) EUN_ROOT:[LOCAL_TARDIS.USR_ADM]
$ CREAT/DIR/PROT=(G:RE,W:RE) EUN_ROOT:[LOCAL_DALEK.USR_ADM]
$ SET DEF TWG$USR:[ADM]
$ BACK/LOG *.*;* EUN_ROOT:[LOCAL_TARDIS.USR_ADM]*
$ BACK/LOG *.*;* EUN_ROOT:[LOCAL_DALEK.USR_ADM]*
$ SET DEF TWG$ETC:[000000]
$ BACK/LOG UTMP. EUN_ROOT:[LOCAL_TARDIS]ETC_UTMP.
$ BACK/LOG UTMP. EUN_ROOT:[LOCAL_DALEK]ETC_UTMP.
```

12. This step needs to be done only if you have different users on the two nodes (ie. the SYS$SYSTEM:SYSUAF.DAT files are not identical). If the SYS$SYSTEM:SYSUAF.DAT file is identical on both nodes, then ignore this step.

```
$ SET DEF TWG$ETC:[000000]
$ COPY PASSWD. EUN_ROOT:[LOCAL_TARDIS]ETC_PASSWD.
$ COPY PASSWD. EUN_ROOT:[LOCAL_DALEK]ETC_PASSWD.
$ CREAT/DIR/PROT=(G:RE,W:RE) EUN_ROOT:[LOCAL_TARDIS.USR_SPOOL]
$ CREAT/DIR/PROT=(G:RE,W:RE) EUN_ROOT:[LOCAL_DALEK.USR_SPOOL]
$ SET DEF TWG$USR:[SPOOL]
$ BACK/LOG [...]*.*;* EUN_ROOT:[LOCAL_TARDIS.USR_SPOOL...]*.*;*
$ BACK/LOG [...]*.*;* EUN_ROOT:[LOCAL_DALEK.USR_SPOOL...]*.*;*
```

13. Invoke the EUNICE BSD cshell:

```
$ @TWG$ADMIN:CSHELL
```

If the installation is correct so far, the cshell prompt (%) will be displayed.

14. Set up the symbolic links, as follows:

```
% umask 022
% cd /usr
% /bin/rm -rf adm
% ln -s /eunlocal/usr_adm adm
% cd /etc
% /bin/rm -f utmp
% ln -s /eunlocal/etc_utmp utmp
```

Note: The % is the cshell prompt.

15. Perform this step **ONLY** if the two nodes have different users (i.e. the SYS$SYSTEM:SYSUAF.DAT files are not identical). If the two nodes have identical SYS$SYSTEM:SYSUAF.DAT files, proceed to Step 16.

```
% cd /etc
% /bin/rm -f passwd
% ln -s /eunlocal/etc_passwd passwd
% cd /usr
% /bin/rm -rf spool
% ln -s /eunlocal/usr_spool spool
```

DO NOT REPEAT STEPS 14, and 15 for the other node (DALEK). Once the symbolic link has been set up on one node, both nodes will be correctly referenced .

16. You may now proceed to bring up EUNICE BSD on the second node of the cluster.

Remember that the second node still needs to have its SYSGEN parameters tuned as specified in Appendix C of this manual.

8. INSTALLATION VERIFICATION

Although the installation has been completed, the EUNICE BSD software must now be tested. The following is a suggested "script" that will verify the basic structure of the system and, at the same time, verify some of the UNIX commands.

```
$ @TWG$ADMIN:CSHELL.COM
Last login:  <<system response>>
             <<Message of the Day >>
% date
  <<system display, date and time zone>>
% ls
  <<UNIX Directory Listing>>
% ls | wc -w
  <<test, pipes, system prints number of files in directory>>
% ed
 ^y
stopped
% jobs
 <<system prints information about editor>>
% fg
ed
q
% logout


$ @TWG$ADMIN:CSHELL.COM
Last login:  <<system response>>
             <<Message of the Day >>
% cd  /tmp
% /etc/eunice/tests.csh
  <<multiple system responses>>
% /etc/eunice/tests.sh
  <<multiple system responses>>
% cd <<to return to home directory>>
```

The last four lines execute two shell scripts which exercise the *csh* and *sh* shells. In exercising the two shells, the *csh* and *sh* shells are executed from the */tmp* directory. REMEMBER: UNIX is a case-sensitive operating system; when you enter the cshell, you should have your terminal keyboard "Caps Lock" key OFF.

9. SYSTEM BOOT PROCEDURE MODIFICATION

Once EUNICE BSD is working properly, you can add the call to STARTEUNICE.COM towards the end of a normal exit of the command file SYS$MANAGER:SYSTARTUP.COM. Edit SYSTARTUP.COM, and add the following line near its end:

```
$ @DUA0:[EUNICE.ETC.EUNICE]STARTEUNICE.COM
```

Remember to change the disk device name to reflect what is appropriate for your system. If you already had a call to restart EUNICE BSD in your SYSTARTUP.COM file, but commented it out as suggested in the beginning of this manual, check that the file specification is correct, and then re-activate the line by removing the exclamation mark from in front of it. Remember to rename the file from "REBOOT.COM" (as it has been named in previous EUNICE BSD releases) to "STARTEUNICE.COM".

Sites which are also running Wollongong's WIN/TCP product should place the call to start WIN/TCP **after** the call to EUNICE BSD. Since WIN/TCP uses the same REX kernel used by EUNICE BSD, they can share a common copy of the REX kernel in a combined installation. Always place the call to [NETDIST.MISC]STARTINET.COM after the call to [EUNICE.ETC.EUNICE]STARTEUNICE.COM when updating the SYSTARTUP.COM file.

A final step in the process of making the WIN/TCP installation work with EUNICE BSD is to set up the links to the TCP/IP commands. A shell script, *wintcp.csh*, which resides in */etc/eunice* has been provided to set up these links automatically once EUNICE BSD has been brought up. It is only necessary to run this script once - the setup it creates is permanent. Log into the cshell, and run the shell script as indicated below.

```
$ @TWG$ADMIN:CSHELL.COM
Last login: <<system response>>
            <<Message of the Day >>
%   /etc/eunice/wintcp.csh
```

After you have adjusted the SYSTARTUP.COM file as indicated above, you may wish to perform a trial system re-boot to make certain that SYSTARTUP.COM is functioning properly so as to be sure that EUNICE BSD will be brought up correctly if the system restarts while unattended.

9.1 RECORDING ALTERATIONS WITH SNAP.CSH

In */etc/eunice*, the shell script *snap.csh* is provided to make copies of all the files touched during the EUNICE BSD installation. It also runs AUTHORIZE (on the system and default accounts, as well as any others you request), and SYSGEN so that a record can be kept of any quotas and parameters changed. The output is redirected to a file called *snapshot* in the current directory. *Snap.csh* is run as follows:

```
$ @TWG$ADMIN:CSHELL.COM
% cd /tmp
% /etc/eunice/snap.csh
% lpr snapshot
or
% vms print snapshot.;
```

Either of the last two commands will send the snapshot file to the line printer.

NOTE: You must have SYSPRV privilege for *snap.csh* to work properly. If you run the script from an unprivileged account, the AUTHORIZE and SYSGEN portions of the *snapshot* will be empty.

10. ADJUSTING EUNICE BSD AFTER MODIFYING VAX/VMS

Modifications to the VAX configuration or to the VMS operating system can have a direct impact on EUNICE BSD. This chapter will cover what EUNICE BSD modifications are required if devices are added or changed, a new directory is added to the UNIX root, memory is added to the VAX, new users are added to the system, or VMS is updated with a new release.

10.1 NEW DEVICES

When new devices are to be accessed from the EUNICE BSD environment, the EUNICE BSD startup and accounting files must be modified.

10.1.1 Terminals

The device information for EUNICE BSD should be changed when any of the following are added to the VAX:

A. Additional or different terminal controller boards

B. WIN/TCP for VMS networking software

C. DECnet networking software

For new terminals to be known to the EUNICE BSD environment, they should be added to the device directory, */dev*. Entries for the new terminals should also be added to the UNIX terminal files. The three files to be changed are as follows:

A. */etc/eunice/dev.com*

B. */etc/ttys*

C. */etc/locations*

The following examples will show sample files which assume that a system is configured with eight hard-wired login lines, called TXA0: through to TXA7:, and allow for four logins each on DECnet (RTAn:) and WIN/TCP (NTYn:).

10.1.1.1 /etc/eunice/dev.com

Logical name assignments for each device are made in */etc/eunice/dev.com*. Terminal names for direct login lines are in the form *tty??*. They must start with *tty0* and continue with increasing decimal numbers for each terminal on the system.

First, there should ALWAYS be a line which says:

```
$ ASSIGN/SYSTEM TT:   TTY
```

After that, there should be an explicit logical assignment for each terminal on the system. Your system may use TT terminals, TX terminals or a combination of both; whichever it is, add the appropriate assignments to DEV.COM using the same syntax as already shown in that file. For a complete discussion of the device-naming conventions, see Section 4.5.

VMS terminal names for DECnet connections are in the form *rty??:*. They start with *rty1:* and increment for each DECnet login. Add as many entries as there will be logins at any one time to the VAX over DECnet.

Terminal names for WIN/TCP connections are in the form *nty??:*. They start with *nty1:* and increment for each WIN/TCP login. Add as many entries as there will be logins at any one time to the VAX over WIN/TCP. The EUNICE BSD software, in order to look more like native UNIX, prefers to see virtual devices called *ttyp??*. The logical names should be adjusted accordingly (as shown below).

For example, the terminal section in */etc/eunice/dev.com* for the above described system would look like the following:

```
$ ASSIGN/SYSTEM TXA0:  TTY0   ! /dev/tty0
$ ASSIGN/SYSTEM TXA1:  TTY1   ! /dev/tty1
$ ASSIGN/SYSTEM TXA2:  TTY2   ! /dev/tty2
$ ASSIGN/SYSTEM TXA3:  TTY3   ! /dev/tty3
$ ASSIGN/SYSTEM TXA4:  TTY4   ! /dev/tty4
$ ASSIGN/SYSTEM TXA5:  TTY5   ! /dev/tty5
$ ASSIGN/SYSTEM TXA6:  TTY6   ! /dev/tty6
$ ASSIGN/SYSTEM TXA7:  TTY7   ! /dev/tty7
$ ASSIGN/SYSTEM _NTY1: TTYP1  ! /dev/ttyp1, 1st WIN/TCP login line
$ ASSIGN/SYSTEM _NTY2: TTYP2  ! /dev/ttyp2, 2nd WIN/TCP login line
$ ASSIGN/SYSTEM _NTY3: TTYP3  ! /dev/ttyp3, 3rd WIN/TCP login line
$ ASSIGN/SYSTEM _NTY4: TTYP4  ! /dev/ttyp4, 4th WIN/TCP login line
$ ASSIGN/SYSTEM _RTA1: RTY1   ! /dev/rty1, 1st DECnet login line
$ ASSIGN/SYSTEM _RTA2: RTY2   ! /dev/rty2, 2nd DECnet login line
$ ASSIGN/SYSTEM _RTA3: RTY3   ! /dev/rty3, 3rd DECnet login line
$ ASSIGN/SYSTEM _RTA4: RTY4   ! /dev/rty4, 4th DECnet login line
```

After adding the logicals for the new devices, it is necessary to add the devices to the */dev* "directory". Use the existing command in DEV.COM as an example for the format, and add your new devices to the "*/dev*" search list. Note that numerical sequences of devices are often specified by ranges of values.

Execute */etc/eunice/dev.com* so that the logical assignments for the new terminals are made. It is very important that the new logical assignments are made to the system before continuing. To make the new assignments, type the following:

```
$ SET PROT=(S:RWED, O:RWED, G:RE, W:RE)/DEFAULT
$ SET VERIFY
$ @TWG$ADMIN:DEV.COM
        << multiple system responses >>
$ SET NOVERIFY
```

10.1.1.2 /etc/ttys

Additional terminals must be added to */etc/ttys* which must be readable by all EUNICE BSD users. The above described system would have the following */etc/ttys*.

```
console none su on
tty0 none vt100 on
tty1 none vt100 on
tty2 none vt100 on
tty3 none vt100 on
tty4 none vt100 on
tty5 none vt100 on
tty6 none vt100 on
tty7 none vt100 on
ttyp1 none vt100 on
ttyp2 none vt100 on
ttyp3 none vt100 on
ttyp4 none vt100 on
rty1 none vt100 on
rty2 none vt100 on
rty3 none vt100 on
rty4 none vt100 on
```

10.1.1.3 /etc/locations

An entry for each of the terminals on the system must be added to */etc/locations*, with terminal device names given in VMS style. The device name should be followed by a colon **and** a comma. If the location field (the space after the comma) is optional. If it is filled out, the *finger* utility will list the location of the terminal. The */etc/locations* file must be readable by all EUNICE BSD users. The */etc/locations* file for the above system would be of the following form:

```
opa0:,VAX console
txa0:,Rm# 207
txa1:,Rm# 203
txa2:,Rm# 205
txa3:,Rm# 202
txa4:,Rm# 206
txa5:,Rm# 209
txa6:,Rm# 192
txa7:,Rm# 102
nty1:,WIN/TCP Virtual Terminal
nty2:,WIN/TCP Virtual Terminal
nty3:,WIN/TCP Virtual Terminal
nty4:,WIN/TCP Virtual Terminal
rta1:,DECnet Virtual Terminal
rta2:,DECnet Virtual Terminal
rta3:,DECnet Virtual Terminal
rta4:,DECnet Virtual Terminal
```

10.1.2 Adding New Disks and Top Level UNIX Directories

When a new disk is to be included in the UNIX file system, an entry needs to be added to the UNIX root. In order to create a directory under "/", a logical assignment for that directory must be made, and the logical which emulates the UNIX root directory must be modified.

The top-level directory of the new disk and the top-level directory of the disk with the UNIX executables must be EXECUTE to everyone using EUNICE BSD, (usually WORLD), as follows:

```
$ SET DEF DRA0:[000000]                ! CONTAINS EUNICE
$ SET PROT 000000.DIR/PROT=W:E
$ SET DEF DRA2:[000000]                ! NEW DISK
$ SET PROT 000000.DIR/PROT=W:E
```

Add an appropriate logical name assignment to */etc/eunice/root.com*. Choose a unique name for the new directory. In the following example, a new user disk, DRA2:, has been added. The disk will have the users' directories at the top level, with the new directory called */u2*. The entry for this directory in */etc/eunice/root.com* would be:

```
$ ASSIGN/SYSTEM/EXEC/TRANS= CONC   DRA2:     TWG$U2    ! /u2
```

Remember to edit the assignment line for the root directory search list after adding any new directory(s).

As another example, assume that it is necessary to move some of the EUNICE BSD software and some user directories to the new disk. In this example, the users will all be placed under a directory called [USER] at the top of the disk. The UNIX directory, */u2*, will point to that directory on DRA2:, as indicated below:

```
$ SET  DEF  DRA2:[000000]
$ CREATE/DIR  [.USER]
```

Add the logical assignment to */etc/eunice/root.com* for the new user directory.

```
$ ASSIGN/SYSTEM/TRANS=CONC   DRA2:[USER.]  TWG$U2    ! /u2
```

To put the */usr* directory on the new disk, create DRA2:[EUNICE.USR] and move the */usr* directory structure there.

```
$ SET DEF DRA2:[000000]
$ SET PROT 000000.DIR/PROT=(S:RWED,O:RWE,G:E,W:E)
$ BACKUP/TRUN DRA1:[EUNICE.USR...]*.*;*   DRA2:[EUNICE.USR...]*.*;*
```

The only change necessary for the */usr* directory is to modify the file TWG$ADMIN:ROOT.COM, which defines TWG$USR:. Type the new logical assignment for TWG$USR into the system and then run ROOT.COM, as follows:

```
$ ASSIGN/SYSTEM/TRANS=CONCEALED TWG$USR DRA2:[EUNICE.]
$ @TWG$ADMIN:ROOT.COM
        << any errors observed are real errors >>
```

A new utility called TWG$ADMIN:EDITROOT.COM has been added to this release of EUNICE BSD, which creates these directory logicals (and the attendant search list) interactively by means of a question and answer session. Please see Appendix E of this Guide for details about the running of this command file.

If, during this process, any new user accounts have been added, or account directories have changed disks, please read the following section.

10.2 ADDING NEW USERS OR ALTERING THE USER AUTHORIZATION FILE

For new users, run AUTHORIZE to create their VMS accounts. Give them at least the minimum quotas specified in Table 2-2. Create their VMS directory and give them disk quotas as necessary. There may be other site-specific requirements, such as adding the user to "a computer-use accounting file". If the user directory is in a directory for which there is no logical assignment in ROOT.COM, this directory needs to be added to the UNIX root. (Refer to Section 2.6, "Selecting Disk Storage Space For EUNICE BSD Files".)

Certain modifications to the authorization file for existing users necessitate the creation of a new */etc/passwd* file. These include changes to the owner, UIC number, default device, or default directory. Run */etc/eunice/adduser.com* to create new */etc/passwd* and */etc/group* files. Verify that the home directory field in the */etc/passwd* file translates to a valid path name. If double slashes appear in the path, edit */etc/eunice/adduser.csh* to change it to a valid path and re-run */etc/eunice/adduser.com.* Refer to Section 5.2, "Setting Up the UNIX User Database Files".

10.3 CHANGING THE VALUE OF MAXPRO

The value of the SYSGEN parameter, MAXPRO, must be at least 64 for EUNICE BSD to operate. SYS$SYSTEM:MODPARAMS.DAT should be modified to have the new value. (Otherwise, an AUTOGEN RE-BOOT will reset MAXPRO to the old value specified in the MODPARAMS.DAT file.) The number of processes, *nproc,* is now automatically read by *initgbl* when TWG$ADMIN:EUNICE.COM is executed. (Refer to Section 2.3, "Verification/Modification of System and Users' Accounts", and Section 4.6, "Set up the EUNICE BSD Run-Time Environment (EUNICE.COM)".) To modify the MAXPRO type the following:

```
$ SET DEF SYS$SYSTEM
$ RUN SYSGEN
SYSGEN> SHOW MAXPRO
SYSGEN> SET MAXPRO 64
SYSGEN> WRITE CURRENT
SYSTEN> EXIT
```

In order for changes to the value of non-dynamic SYSGEN parameters, such as MAXPRO, to take effect, it is necessary to re-boot the system. This will also create a new VFORKCOMM global section file which knows about the new number of processes.

10.4 NEW VERSION OF THE VMS OPERATING SYSTEM

EUNICE BSD is supported on the version of VMS which is current at the time of release, unless otherwise specified. In most cases, a minor release of VMS will not affect the ability to use EUNICE BSD. EUNICE BSD, Version 4.3.1 is compatible with VMS 4.2 and later versions, but not with earlier versions of VMS.

No modifications to EUNICE BSD files are necessary when a new version of VMS is installed, but you need to be sure that the VMS SYSGEN parameters are still set up appropriately. Remember that MAXPRO has to be at least 64 for EUNICE BSD to run correctly, as indicated in the previous section.

When non-dynamic SYSGEN parameters such as MAXPRO are modified, the system must be re-booted for them to take effect. Re-boot before running the EUNICE BSD startup file, TWG$ADMIN:STARTEUNICE.COM.

Some sites find it necessary to increase other SYSGEN parameters, such as the KFILSTCNT or GBLPAGES. Since AUTOGEN is usually run after a new release is added, these values have probably been decreased. If they need to be increased, error messages will be returned when the EUNICE BSD startup files are run. It is a good idea to take the parameters of the old SYS$SYSTEM:MODPARAMS.DAT, and immediately include them in the new one - they have usually been set up to allow use of particular software packages (such as EUNICE BSD).

10.5 ADDING MORE INTERNAL MEMORY TO THE VAX MACHINE

When more memory is added to the VAX, some SYSGEN parameters may be modified. Please refer to Section 2.4 for advice regarding the insertion of important SYSGEN parameters in the MODPARAMS.DAT file.

10.6 INCREASING AVAILABLE DISK STORAGE SPACE

The following directories are areas where unnecessary files tend to build up, or which contain non-essential files. (See Table 2-4 for further details.)

/usr/lib/learn/play: These files are maintained by *learn(1)* so users may continue their lessons from the place they left off. If a user is no longer using *learn(1)*, the files belonging to that user should be deleted.

/usr/man/man1 through */usr/man/man8:* These directories contain the unformatted version of the *UNIX User's Reference Manual [URM]*, Section 1. If there is a severe lack of disk space, these files may be deleted and the hard copy referenced.

/usr/man/cat1 through */usr/man/cat8:* These files are the formatted versions of Section 1 of the *UNIX User's Reference Manual [URM].* When a user requests a manual page with the *man(1)* command, the file from */usr/man/man1* through */usr/man/man8* is formatted. A copy is sent to the terminal. The formatted copy is reserved in the associated */usr/man/cat* directory, so it will not need to be re-formatted for the next user. It is usually a good idea to keep the commonly accessed manual pages in the *cat* directories. It may be desirable to delete rarely used or lengthy manual pages.

/usr/doc: These files are mostly sources for documents in the *UNIX User's Supplementary Documents [USD]* and the *UNIX Programmer's Supplementary Documents [PS1]* and *[PS2],* and can be deleted without affecting the performance of EUNICE BSD.

/usr/games: Although important to many people, these programs are rarely regarded as indispensable. There are many library files under this directory too, so a recursive VMS DELETE is recommended.

/usr/lib/font: The directory, and others at the level (fontinfo, me, ms, tmac), are used by *troff(1)* and contain many small files.

/usr/lib/uucp: The directory only uses approximately 370 blocks. However, if the directory is never used, it can be removed.

11. INSTALLATION AND ADMINISTRATION OF UUCP

This section describes the VMS part of *uucp(1)* administration. For the UNIX part, see the *UNIX System Manager's Manual [SMM].*

The VMS system administrator should:

A. Increase the value of the nondynamic SYSGEN parameter TTY_TYPAHDSZ and re-boot the system so that the new value takes effect.

B. Create a UUCP user account.

C. Edit the lines containing MODEMIN and MODEMOUT in the file TWG$ADMIN:UUCP.COM to reflect the serial lines used by *uucp*. Set UUCP_HOST_NAME to the desired *uucp*-name of your machine in the TWG$ADMIN:EUNICE.COM file.

11.1 REBOOT THE SYSTEM TO INCREASE THE VALUE OF TTY_TYPAHDSZ

To run *uucp* reliably, you will have to adjust the size of the typeahead buffer TTY_TYPAHDSZ. The default value of 78 should be increased to 255 to improve performance and allow you to run *uucp* at high speed (up to 19.2K).

To change TTY_TYPAHDSZ:

1. Edit the file SYS$SYSTEM:MODPARAMS.DAT to add the following line:

   TTY_TYPAHDSZ = 255 ! For the reliable use of uucp

2. Issue the following command:

   $ @SYS$SYSTEM:SHUTDOWN

11.2 USE AUTHORIZE TO CREATE THE UUCP USER ACCOUNT

1. Run the following commands:

   ```
   $ SET DEF SYS$SYSTEM
   $ RUN AUTHORIZE
   ```

2. Once you get the "UAF>" prompt, modify the SYSUAF.DAT record for the UUCP user by entering the AUTHORIZE requests below:

   ```
   UAF> ADD UUCP /PASS=EUNICE /UIC=[020,001]
     <<user record copied>>
   UAF> MODIFY UUCP/DIR=[LIB.UUCP]
     /DEV = TWG$USR
     /OWNER = UUCP
     /PRIV = (TMPMBX, NETMBX, SYSPRV)
     /LGICMD = TWG$USR:[LIB.UUCP]LOGIN.COM
     /FLAGS = (DISWELCOME, DISREPORT)
    <<user record updated>>
   UAF> EXIT
   ```

Just as for any new user account, remember to run ADDUSER.COM to create an entry for *uucp* in the */etc/passwd* file.

11.3 EDIT TWG$ADMIN:UUCP.COM

Edit the lines containing MODEMIN and MODEMOUT entries in TWG$ADMIN:UUCP.COM substituting your *uucp* serial ports for the incoming and outgoing lines.

Example

```
$ MODEMIN="TXB8:"      ! Incoming line
$ MODEMOUT="TXB9:"     ! Outgoing line
```

Set the name of your machine in TWG$ADMIN:EUNICE.COM by modifying the line which sets the **UUCP_HOST_NAME** logical. Please edit the field between quotation marks as indicated in the instructions in Section 4.6 of this manual.

11.4 UUCP-SPECIFIC SET-UP PROCEDURES

Turn to your *UNIX System Manager's Manual [SMM]*, and set up the UNIX part of *uucp* according to the instructions detailed there.

11.4.1 Polling Sites

EUNICE BSD has the capacity of polling various sites on an hourly basis at prescribed hours. This can be specified in the file */usr/lib/uucp/poll_sites*. A sample file is provided. The lines from this file are shown below:

```
twg3b20   4 6   11 13 15   19   23 #
twg-ap 5 10   20   22 #
vax1 3 #
```

This would cause the site twg3b20 to be polled at 4:00AM, 6:00AM, 11:00AM, 1:00PM, 3:00PM, 7:00PM and 11:00PM. The site twg-ap would be polled at 5:00AM, 10:00AM, 8:00PM and 10:00PM; and the site vax1 at 3:00AM.

Set up this file according to the sites to be polled.

11.4.2 Fix sendmail.cf

The file # */usr/lib/sendmail.cf* should be aware of the UUCP sites with which you wish to communicate. This involves changing the line following the one that says:

```
#INSTALLER UPDATE THIS LINE FOR UUCP SITES
```

Add the UUCP site names here. The line shown is only a sample. Note this is the "CU" macro. Refer to the "Sendmail Installation and Operation Guide" in the *System Manager's Manual [SMM]*.

11.5 EDIT STARTEUNICE.COM

Remove the comment lines from the line that starts UUCP.COM in the main EUNICE startup file, TWG$ADMIN:STARTEUNICE.COM. The entry for UUCP.COM is near the end of the file.

When everything is done, *uucp* will start automatically. In the event that you have not had to change any of the SYSGEN parameters, you can initiate *uucp* by hand immediately, by issuing the following command.

```
$ @TWG$ADMIN:UUCP.COM
```

12. LICENSING OBLIGATIONS

The EUNICE BSD environment provides a transportation link which enables of software written for the UNIX operating system to run in a VMS environment. This link is referred to as REX (Runtime Executive), by The Wollongong Group. REX provides a cost-effective solution for migrating applications to other VMS systems.

Porting UNIX-based software to run under VMS is a very powerful ability, and is the main reason EUNICE BSD was purchased at some sites. However, it is important that the programmer using EUNICE BSD be aware of licensing obligations if these programs are to be moved to another VAX.

Both the UNIX and EUNICE BSD code are licensed on a per-CPU basis. This means that if any portion of the distributed software from the EUNICE BSD tape is moved to a machine other than the licensed CPU, a license agreement is necessary for the target machine. The licensed CPU serial number appears on your UNIX and/or EUNICE BSD license agreement(s).

The libraries which contain the Section 2 system calls, (see the *UNIX Programmer's Reference Manual [PRM]*, Section 2 are the heart of this transportation link. These libraries are:

SYS$SHARE:TWG_LIBC_43.EXE
TWG$USR:[LIBVMS]LIBC.OLB
/lib/libc.a

When programs which were originally written to run in the UNIX operating environment are compiled with these libraries, they can then be run under the VMS operating system. If a program is compiled by *cc(1)* or *f77(1)* with the alias *vmsobj* set, the code for the system calls included in the executable program will come from the VMS object format libraries. Because these libraries contain EUNICE BSD code, a REX license is required to move any portion of the code in *libc* or the shareable libraries to another machine.

In addition, if any of the following EUNICE BSD files are to be placed on a machine which does not have a EUNICE BSD license, a REX license is required.

CLISETUP.EXE	INITGBL
VFORKCLI.EXE	USERS.COM
FORKDUMY.EXE	ROOT.COM
FORKDUMY1.EXE	DEV.COM
SHELL	EUNICE.COM
	SUCHMOD.COM
	RC.COM
	STARTEUNICE.COM

Use of the UNIX executables is not included in the REX license. As a special case, a company can also obtain the right to use a selected list of the UNIX executables. For example, if commands such as *date(1)* or *csh(1)* are used in a program to be run on another VAX, a special agreement can be constructed which includes REX and the selected UNIX executables.

If the compilers provided with EUNICE BSD are only being used as a development tool, and no EUNICE BSD or UNIX code is being included in the executable, there is no need for a REX license for the target VAX. In this case, the compiled version for the target machine must be compiled with a non-UNIX compiler, such as the DEC™ C compiler. Note that if you choose to use the DEC C compiler, The Wollongong Group will not support the resultant code. It is important to recognize that no EUNICE BSD or UNIX proprietary code can legally be placed on a machine which does not have a valid license for that code.

Each REX distributorship is arranged individually with The Wollongong Group. If you have any questions regarding REX, please contact your sales representative.

Appendix A. EUNICE BSD WITH VMS 4.x

With the release of the 4.x series, the VMS operating system has several new features which affect the operation of EUNICE BSD.

System Logical Name Table Size Limits

Logical name tables now have size limits which are set using SYSGEN. The system logical name table size is limited by the parameter LNMSHASHTBL, which has a default value of 128. This size should be reviewed, since most sites will need to increase this value; a more practical value is 256.

Process Logical Name Table Size Limits

The process logical name table size, LNMPSHSHTBL, needs at least 10 entries per EUNICE user.

File Names

Under VMS 4.x, filenames can have 39 characters with up to 39 characters in the extension (39chars.39chars) and contain many special characters which were not allowed in the past.

EUNICE BSD, Version 4.3.1 will allow names up to 39 lowercase characters, and a 39 lower case character extension. Any filenames bigger than this will be hashed by EUNICE BSD.

The hashing algorithm introduced with EUNICE BSD Release 4.3.1 takes advantage of the longer file names. If you are just moving from VMS 3.x to 4.2 or better, be sure to convert the old filenames to the new algorithm. See the *UNIX User's Reference Manual [URM]*, Section 1, *cvtfnames(1)*.

Installing Programs with Privilege

As a new security feature, the INSTALL utility now requires that a program which is installed with privilege not have a debug entry point. To make a program installable with privilege, either load with the -notrace option or use *trpatch(1)*.

Unless the file specification used to install and run a utility with privilege is set up in a special manner, the privileges will not be retained when the utility is used. The file specification must be recognized as starting from a device.

If the disk is not clustered, the device specification might look like this:

```
DRA0:[USER.LOCAL]MYPROG/PRIV=(WORLD,SYSLCK)
```

On a clustered disk, simply use the full device name, followed by the directory and file name:

```
HSC000$DRA0:[USER.LOCAL]MYPROG/PRIV=(WORLD,SYSLCK)
```

Of course, one may use the logical equivalent for part of the specification. For example, if SITE_USERS has been assigned to _DRA0:[USER.], the following could be used to install the utility:

```
SITE_USERS:[LOCAL]MYPROG/PRIV=(WORLD,SYSLCK)
```

The exact file specification scheme used to install the utility is what must be used to reference it when setting up the symbol for a foreign command. For instance, if:

```
SITE_USERS:[LOCAL]MYPROG/PRIV=(WORLD,SYSLCK)
```

is used to install MYPROG.EXE, the symbol should be set up as:

```
MYPROG :== $SITE_USERS:[LOCAL]MYPROG      MYPROG
```

New Terminal Drivers

The new terminal drivers under VMS 4.x have changed the control character for end-of-file (EOF) to ^Z instead of ^D. To send an EOF to terminate I/O redirection with utilities such as *cat*, *echo* or *mail*, use a ^Z.

VAX Clusters

The following points should be noted:

A. EUNICE BSD should be installed on a non-shared disk. If not, AT&T requires a license for each CPU which can access EUNICE BSD on shared disks.

B. EUNICE BSD will not use cluster search lists.

Appendix B. GUIDE TO INSTALLATION PROBLEM SOLVING

NOTE: For operational (run-time) errors, please refer to the *EUNICE BSD Reference Manual*, Section 5, Guide to Operations Problem Solving.

ADDUSER.COM,
adduser.csh

symptom: home directory pathnames in */etc/passwd* are not valid for your site, uses */u*.

solution: After creating */etc/passwd*, TWG$ADMIN:ADDUSER.CSH edits it using *ed(1)*. Change a line in this section to change */u* to the appropriate directory name for the local site.

ADDUSER.COM,
adduser.csh

symptom: cannot find device

solution: This script calls *cvtuaf(8)*, which creates the UNIX *passwd* file. (Nested logical name translation will now be performed on the specification of the users' default directory as defined in the authorization file.) If the pathnames in */etc/passwd* have been somehow corrupted, you can still have *adduser.csh* edit the */etc/passwd* file to change double slashes in the home directory name to contain the appropriate home directory path. With the latter solution, you will end up with a valid */etc/passwd*, but will continue to see the error each time ADDUSER is run.

It is most likely that the "/" search list does not match the logicals that have been set up in the ROOT.COM file. Check carefully that they agree.

ADDUSER.COM,
adduser.csh
(cvtuaf)

symptom: running *cvtuaf* returns:
device DRA0: not found

solution: The *cvtuaf* program was unable to find a logical name for the specified device (here DRA0:) as found in the SYS$SYSTEM:SYSUAF.DAT file. Make sure that ROOT.COM has a logical name translation for this device. If not, add one to ROOT.COM, re-run ROOT.COM, and then re-run ADDUSER.COM.

ADDUSER.COM
adduser.csh
passwd

symptom: */tmp/sh##* no such file or directory

solution: This error will be produced by *vi(1)* due to a high RMS multi-block buffer count. From a system account, decrease the blocking factor, trying values from 8 down to 2:

$ SET RMS/SYSTEM/BLOCK=8

cd(1)	symptom: *cd ..* cannot find directory solution: The authorize file has the user's device set to a logical assigned to a non-existent directory. Check that the "/" search list and the logicals defined in ROOT.COM match correctly.
EUNICE.COM	symptom: global page table full solution: increase the SYSGEN parameter GBLPAGES and reboot. Also edit MODPARAMS.DAT with the new value. EUNICE BSD requires a minimum of 661 global pages.
EUNICE.COM	symptom: not able to create global section $CRMPSC error solution: The account used to run EUNICE.COM must have SYSGBL privilege. In addition to CMKRNL, CMEXEC, SYSNAM, SYSPRV, WORLD, OPERATOR. Type: **$ SET PROC/PRIV=SYSGBL**
@TWG$ADMIN:CSHELL	symptom: null no match when trying to get into the cshell solution: permissions on */etc/passwd* must be W:R or the user did not use the command file provided which should call the TWG$BIN:CSH executable. It is a good idea to put an assignment for this command file in the system-wide login file.
@TWG$ADMIN:CSHELL	symptom: cannot find an entry for you in */etc/passwd* solution: Need to re-run ADDUSER.COM or set /PROT to W:R on */etc/passwd.*
eunlogin(1)	symptom: 'eunlogin' returns: EUNLOGIN LCK FAILED solution: Either EUNLOGIN is not installed properly or the symbol is not set up properly. It should be installed with WORLD and SYSLCK privileges. See TWG$ADMIN:EUNICE.COM and TWG$ADMIN:USERS.COM.
EUN_ROOT	symptom: why is this necessary? solution: This should be assigned to a rooted file specification. It is used by the EUNICE BSD startup command files. It must have the directory name ending in a dot. The */bin*, */etc*, and */lib* directories should be under the directory named in EUN_ROOT. solution: this must be executed by the SYSTEM with all privileges except BYPASS.

shell tests

symptom: cat with input from a document and output cat.xxx hangs or responds with:

/tmp/sh550: cannot create

solution: Decrease RMS multi-block count. SET RMS/BLOCK=8 or lower from system account.

SUCHMOD.COM

symptom: global page table full when installing images

solution: Increase the SYSGEN parameter GBLPAGES and reboot. Also edit SYS$SYSTEM:MODPARAMS.DAT with the new value. EUNICE BSD requires at least 661 global pages.

SUCHMOD.COM

symptom: VMS returns error about running out of slots.

solution: Increase KFILSTCNT and reboot. Edit SYS$SYSTEM:MODPARAMS.DAT to reflect this change. EUNICE BSD installs images from six directories.

SUCHMOD.COM

symptom: installing utilities returned:
no name, no message

solution: Run SYSGEN to increase the NPAGEDYN (NON-PAGED DYNAMIC MEMORY).

Appendix C. FINE-TUNING SYSTEM PARAMETERS

Fine tuning the system parameters involves possibilities too numerous for all of them to be included in this publication. This appendix includes procedures for tuning system parameters which most often require fine tuning, and reflects the collective experience of the EUNICE BSD installers at Wollongong.

References are made to the Wollongong system. This particular configuration was:

VAX 11/750
8MB memory
Disk drives: 300MB, 120MB, 80MB, 10MB

If, during installation, situations are encountered which you would like to see detailed in a future version of this publication, please write to The Wollongong Group.

A. Process Information

VMS alone uses one or two processes per user. VMS/EUNICE BSD uses one process for the shell, and one process per command. These are called "the basic two".

NOTE: Some commands use more than "the basic two" processes.

- *cc* uses an additional process for each pass of the C compiler.
- Each background job uses one (or more) additional processes per command.
- *man* uses two more processes than "the basic two" because it uses pipes.

1. Quantum - The Scheduling Quantum
   a. Decides when a program is interactive or compute bound.
   b. The value should be large enough to satisfy I/O and interactive requests to reflect the non-compute-bound nature of the environment, otherwise programs will run more slowly. (Compute-bound programs receive a lower priority.)
   c. **$ RUN SYSGEN**
   d. **SYSGEN> SHO QUANTUM**

      NOTE: The default of 20 is acceptable.

      Clock = 200 (20 x 10) milliseconds.

B. Record Management System Parameters

SYSGEN> SHO /RMS

To enable read ahead and write behind.

1. SYSGEN> **SHO RMS_DFMBC** - The multi block count
   a. When doing a read, read "x" blocks at once.
2. SYSGEN> **SHO RMS_DFMBFSDK** - The multi buffer count
   a. Use "x" buffers and keep them filled.
   b. RMS_DFMBC default = 16
   c. RMS_DFMBFSDK default = 0 (which means 1)

   For example:
   The Wollongong Group system, RMS_DFMBC 7
   The Wollongong Group system, RMS_DFMBFSDK 2

C. Paging Parameters

1. Pages in page file: SYSGEN> **SHOW /PQL**
   a. If there are not enough pages, the system will lock up, which means that the system will go into mwait and wait for pages to be freed.
   b. PQL_DWSEXTENT - Working set extent
      1. This new parameter was introduced in the upgrade from VMS 2.x to VMS 3.x,
      2. If there is a lot of memory available, the working set extent will be higher.
      3. The maximum amount of memory you can access measured in *n* number of pages.
      4. SHO PQL_DWSEXTENT (make it large!, e.g., 2000)
         The Wollongong Group system, 1000.

   This parameter permits the subprocess to be passed the maximum number of pages allowed by the PQL_DWSEXTENT.

2. Working Set
   a. Every user has a working set which is a circular list of pages for each process.
   b. When a page fault occurs (there are too many pages on the circular list), the working set throws a page either to the Free list, or the Modified list which provides a system-wide cache of pages.
   c. WSMAX - Working set max SYSGEN> **SHO WSMAX**
      1. On a 1 mb system (2,048 pages) a working set of 1000 gives users 1/2 of memory.
      2. On a 2 mb system (4,496 pages) a working set of 2000 gives users 1/2 of memory.

      The Wollongong Group system, 977.

   NOTE: Each user's UAF may limit that user's memory access further.

   NOTE: The WSMAX setting, depends on the system's amount of physical memory. For example, the cc program uses working set of 300 pages. Use ^T to see the working set (multiply that number by 2).

3. The Modified Page Writer

   a. Several parameters are affected when pages get moved from the Modified list (by writing to disk) to the Free list, as indicated below:

      MPW_HILIM
      MPW_LOLIM
      MPW_WRTCLUSTER
      MPW_WAITLIM
      FREELIM
      FREEGOAL
      BALSETCNT
      KFILSTCNT
      VIRTUALPAGECNT
      NPAGDYN

   b. These parameters may be set to optimize system performance; see below.

4. MPW_HILIM - The Modified Page Writer High Limit

   1. The MPW_HILIM triggers the Modified Page Writer to write pages on disk.
   2. When the Modified Page List grows larger than the MPW_HILIM, the Modified Page writer is triggered to begin writing Modified Pages out to the disk, thus freeing those pages for adding to the Free List.
   3. This parameter is set according to the total memory on the system.

      The Wollongong Group system, 8192 (which is 2/3 of total memory).

5. MPW_LOLIM - The Modified Page Writer Lo Limit.

   1. The MPW_LOLIM tells the Modified Page Writer when to stop writing pages to disk.

      The Wollongong Group system, 400.

6. MPW_WRTCLUSTER - The Modified Page Writer Write Cluster Size

   1. The MPW_RWTCLUSTER tells the Modified Page Writer how many pages to put in a cluster for each I/O (write to disk) operation.
   2. This allows the write call to move pages to disk more efficiently.

      The Wollongong Group system, 96 (blocks). (This means that to move 2 mb to disk takes 90 I/O operations of 96 blocks each time; this equals FAST operations.)

7. MPW_WAITLIM - The Modified Page List Wait Limit

   1. If one process generates a lot of Modified Pages, it can fill up the Modified Page List and force the Modified Page List to contain only the process' Modified Pages. This impacts system performance by filling up the system-wide cache.
   2. SHO MPW_WAITLIM

      The Wollongong Group system, 8192 (blocks).

8. FREELIM - The low threshold on the Free Page List

1. If the free page list is low, it is time to wake up the Modified Page Writer (to write some pages to disk and free the pages).
2. SHO FREELIM

   The Wollongong Group system, 50.

9. FREEGOAL - The goal set for writing free pages in the Free Page List
   1. When the FREELIM is hit, the Modified Page Writer begins writing modified pages to disk in order to provide free pages to the Modified Free List. The FREEGOAL is set to tell the Modified Page Writer that there are enough pages in the Free List and that it has reached its goal.
   2. SHO FREEGOAL

      The Wollongong Group system, 150.
10. BALSETCNT - Balance set count is the maximum number of processes swapped in at one time.
    1. The BALSETCNT is used to force processes to be swapped out.
    2. A symptom of having the BALSETCNT set poorly occurs when lots of processes sitting in memory and lots of paging is going on (use monitor to see activity).
    3. SHO BALSETCNT

       The Wollongong Group system, 50 (50 processes in memory at one time).

    NOTE: As there are more users on the system, BALSETCNT should be increased slowly.
11. KFILSTCNT - Known files count.
    1. Used to limit the number of directories from which programs can be installed in memory.
    2. SHO KFILSTCNT
    3. EUNICE BSD uses 7.
12. VIRTUALPAGECNT - Virtual page count
    1. Amount of virtual memory for any process.
    2. SHO VIRTUALPAGECNT

       The Wollongong Group system, 8192.

       NOTE: A large program, such as some LISP programs, needs a lot of virtual memory.
    3. $ SHOW MEM (see Main Memory)
13. NPAGDYN - Non-paged dynamic
    1. Size of non-paged virtual memory; used for making pipes.
    2. If all pages are used the system locks!
    3. SYSGEN> **SHO NPAGDYN**

4. $ SHO MEM -- tells how many bytes are free.

   NOTE: The number of users makes a difference, e.g., approximately 15 users keep 30,000 bytes. Use SYSGEN to add number of pages.

   SYSGEN> USE CURRENT
   SYSGEN> SHO NPAGDYN
   SYSGEN> SET NPAGDYN n

   The Wollongong Group system, 90,624.

D. File Manager Parameters

This section deals with the size of caches in the file manager used to reduce disk traffic.

SYSGEN> SHO /ACP
MONITOR FCP (The cache hit rate should be high and close to the call rate.)
(The create and write rates should be low.)

Parameters	The Wollongong Group's System Values	Description
1. ACP_MULTIPLE	1	All of read-only sections
2. ACP_SHARE	1	If more than one disk controller
3. ACP_HDRCACHE	100	Each file has a header (similar to *inode*). Here it is 100 of the most recently accessed file headers.
4. ACP_DIRCACHE	100	Number of directory blocks
5. ACP_WORKSET	0	File manager can have as much physical memory as needed.
6. ACP-WINDOW	32	#window pointers to keep in cache

Appendix D. SUPPORT SERVICES

The Wollongong hotline is staffed from 7:30am to 5pm PST, Monday through Friday. The direct line to Product Support is 415/962-7140. Support calls should be made by the system administrator or an appointed representative only. When phoning Wollongong, please be ready to provide your license number, the EUNICE BSD version, and information about your system's configuration. Questions cannot be addressed without this information.

The EUNICE BSD documentation set should be consulted before contacting Product Support. Sections listing the 4.3 BSD commands supported by EUNICE BSD and guides to problem solving are in the *EUNICE BSD Reference Manual.* The *EUNICE BSD Administrator's Guide* contains additional information on installation problem solving. In addition, both the on-line and hard copies of the UNIX manual pages have EUNICE NOTES indicating circumstances under which utilities behave in a way other than expected on native 4.3 BSD.

If customers encounter EUNICE BSD behavior which they believe requires modification, they are encouraged to submit the details thereof on the Modification Request (MR) form. MRs are always welcome, as they assist in providing products better suited to customer needs.

The initial and extended support services are described in the *System Administrator's Guide, Volume V.* The Eunice Support Information section of this manual describes the Modification Request Services program in detail and includes blank Modification Request forms. A copy of the Support Services agreement is included for your convenience.

The EUNICE BSD Support Information section in the *System Administrator's Guide, Volume V* describes the initial support you receive with product purchase and the extended support services renewal procedure. A copy of the Support Services Agreement is included for your convenience. This section also describes the Modification Request Services program in detail and includes blank Modification Request forms.

Appendix E: EDITING ROOT.COM COMMAND FILE USING EDITROOT.COM

The command file ROOT.COM assigns logicals to the logical name tables that are used by EUNICE BSD. This file can be interactively edited using EDITROOT.COM. EDITROOT.COM opens ROOT.COM and locates all lines pertaining to logical name assignments. You are prompted to determine whether the assignment is to be modified, and, if so, the modification is done and the line is written out to a new file called NEWROOT.COM.

To use EDITROOT.COM, set your default to TWG$ADMIN (or [EUNICE.ETC.EUNICE]), give your account privilege, and type:

```
$ @EDITROOT.COM
```

to which the program returns:

```
**************************
Checking root.com for
logical assignments.
**************************
```

This indicates that EDITROOT.COM has opened the command file ROOT.COM to search each line for assignment statements.

Assigning EUNICE BSD Directories

All directories which come with the distribution must be installed under the [EUNICE] top-level directory. If the line is an assignment to a EUNICE BSD sub-directory, only the device assignment is allowed to be modified. (These are the usual five system directories - */lib*, */etc*, */bin*, */usr* and */vos*.) EDITROOT.COM will prompt you for the new device name (if you desire to change it):

```
Keep filesystem TWG$ETC on DRA1:?[Y/N]: Y
```

In the example above, the assignment contained the filesystem */etc* and the device name DRA1:. If the response is 'yes', as it is in the example, the device name is kept the same. If the response is 'no' you will be prompted for a new device name.

If an assignment is encountered which involves a non-EUNICE-distribution directory the command file will automatically prompt you for both the device name and the directory specification.

Note that these directory specifications have to be ROOTED specifications in the new releases of EUNICE BSD (i.e. DUA0:[EUNICE.ETC.] rather than DUA0: [EUNICE.ETC]).

Root Directory Search List Modification and Assignments

When the command file encounters a search list definition, it parses out the filesystems in the search list and prompts you for input. The device name (the disk logical) and the directory to which the filesystem is assigned are used to create an assignment line for that filesystem in NEWROOT.COM.

The first line in ROOT.COM which starts the root directory search list assignment (the 'define/sys.."/" line) defines necessary members of the list and is not parsed for filesystems' names. Starting with the next line after the 'define' statement, filesystem names are parsed out. Each name is presented with a query asking you if you want to keep it in the list:

```
tmp >> Keep this filesystem in search list? [yes]: Y
Keep tmp on which device? [enter device]: DRA3:
Keep tmp on which directory (e.g. [user.]])? [RET for none]: [EUNICE.TMP.]
```

In the above example, the filesystem TMP is going to be assigned to the device DRA3: and the directory [EUNICE.TMP.]. The filesystem is ignored and no new device/directory assignment is made for it in NEWROOT.COM.

If the filesystem was previously assigned to a logical by ROOT.COM then an assignment statement for that filesystem is not made again.

Adding New File Systems to the Search List

After the current list is done you are prompted for additions to the list. Each addition also requires a device name and directory.

```
Do you want to add new members to your search list?: Y

New filesystem search list member? [RET for last entry]: max
What device will it be on?: DRA3:
What directory is it on [RET for none]: [USER.MAX.]
```

The example above illustrates the prompting for new list members. Each new member is the name of a filesystem you intend to have available in the root directory. The filesystem name entered is 'MAX' and it will reside on DRA0:[USER.MAX.]

Output of EDITROOT.COM

The command file EDITROOT.COM writes out a new command file name NEWROOT.COM. This file is a combined version of the original ROOT.COM file, and the changes you entered in response to the prompting. This output file should be checked before using it in place of ROOT.COM. Once you have checked the contents, if you wish to use this command file at system reboot time (which is almost always the case), you must rename it ROOT.COM. This way it will automatically be called by STARTEUNICE.COM when your system restarts.

An Introduction to the Revision Control System

Walter F. Tichy

Department of Computer Sciences
Purdue University
West Lafayette, IN 47907

ABSTRACT

The Revision Control System (RCS) manages software libraries. It greatly increases programmer productivity by centralizing and cataloging changes to a software project. This document describes the benefits of using a source code control system. It then gives a tutorial introduction to the use of RCS.

Functions of RCS (Revision Control System)

RCS manages software libraries. It greatly increases programmer productivity by providing the following functions.

1. RCS stores and retrieves multiple revisions of program and other text. Thus, one can maintain one or more releases while developing the next release, with a minimum of space overhead. Changes no longer destroy the original -- previous revisions remain accessible.
   a. Maintains each module as a tree of revisions.
   b. Project libraries can be organized centrally, decentralized, or any way you like.
   c. RCS works for any type of text: programs, documentation, memos, papers, graphics, VLSI layouts, form letters, etc.
2. RCS maintains a complete history of changes. Thus, one can find out what happened to a module easily and quickly, without having to compare source listings or having to track down colleagues.
   a. RCS performs automatic record keeping.
   b. RCS logs all changes automatically.
   c. RCS guarantees project continuity.
3. RCS manages multiple lines of development.
4. RCS can merge multiple lines of development. Thus, when several parallel lines of development must be consolidated into one line, the merging of changes is automatic.
5. RCS flags coding conflicts. If two or more lines of development modify the same section of code, RCS can alert programmers about overlapping changes.
6. RCS resolves access conflicts. When two or more programmers wish to modify the same revision, RCS alerts the programmers and makes sure that one modification won't wipe out the other one.

7. RCS provides high-level retrieval functions. Revisions can be retrieved according to ranges of revision numbers, symbolic names, dates, authors, and states.
8. RCS provides release and configuration control. Revisions can be marked as released, stable, experimental, etc. Configurations of modules can be described simply and directly.
9. RCS performs automatic identification of modules with name, revision number, creation time, author, etc. Thus, it is always possible to determine which revisions of which modules make up a given configuration.
10. Provides high-level management visibility. Thus, it is easy to track the status of a software project.
    a. RCS provides a complete change history.
    b. RCS records who did what when to which revision of which module.
11. RCS is fully compatible with existing software development tools. RCS is unobtrusive -- its interface to the file system is such that all your existing software tools can be used as before.
12. RCS' basic user interface is extremely simple. The novice need to learn only two commands. Its more sophisticated features have been tuned towards advanced software development environments and the experienced software professional.
13. RCS simplifies software distribution if customers maintain sources with RCS also. This technique assures proper identification of versions and configurations, and tracking of customer modifications. Customer modifications can be merged into distributed versions locally or by the development group.
14. RCS needs little extra space for the revisions (only the differences). If intermediate revisions are deleted, the corresponding differences are compressed into the shortest possible form.
15. RCS is implemented with reverse deltas. This means that the latest revision, which is the one that is accessed most often, is stored intact. All others are regenerated from the latest one by applying reverse deltas (backward differences). This results in fast access time for the revision needed most often.

How to get started with RCS

Suppose you have a file f.c that you wish to put under control of RCS. Invoke the checkin command:

```
ci f.c
```

This command creates f.c,v, stores f.c into it as revision 1.1, and deletes f.c. It also asks you for a description. The description should be a synopsis of the contents of the file. All later checkin commands will ask you for a log entry, which should summarize the changes that you made.

Files ending in ,v are called RCS files ("v" stands for "versions"), the others are called working files. To get back the working file f.c in the previous example, use the checkout command:

```
co f.c
```

This command extracts the latest revision from f.c,v and writes it into f.c. You can now edit f.c and check it in back in by invoking:

```
ci f.c
```

Ci increments the revision number properly. If *ci* complains with the message

```
ci error: no lock set by <your login>
```

then your system administrator has decided to create all RCS files with the locking attribute set to 'strict'. In this case, you should have locked the revision during the previous checkout. Thus, your last checkout should have been

```
co -l f.c
```

Locking assures that you, and only you, can check in the next update, and avoids nasty problems if several people work on the same file. Of course, it is too late now to do the checkout with locking, because you probably modified f.c already, and a second checkout would overwrite your modifications. Instead, invoke

```
rcs -l f.c
```

This command will lock the latest revision for you, unless somebody else got ahead of you already. In this case, you'll have to negotiate with that person.

If your RCS file is private, i.e., if you are the only person who is going to deposit revisions into it, strict locking is not needed and you can turn it off. If strict locking is turned off, the owner off the RCS file need not have a lock for checkin; all others still do. Turning strict locking off and on is done with the commands:

```
rcs -U f.c      and      rcs -L f.c
```

You can set the locking to strict or non-strict on every RCS file.

If you don't want to clutter your working directory with RCS files, create a subdirectory called RCS in your working directory, and move all your RCS files there. RCS commands will look first into that directory to find needed files. All the commands discussed above will still work, without any modification*.

To avoid the deletion of the working file during checkin (in case you want to continue editing), invoke

```
ci -l f.c
```

This command checks in f.c as usual, but performs an additional checkout with locking. Thus, it saves you one checkout operation. There is also an option **-u** for *ci* which does a checkin followed by a checkout without locking. This is useful if you want to compile the file after the checkin. Both options also update the identification markers in your file (see below).

Automatic Identification

RCS can put special strings for identification into your source and object code. To obtain such identification, place the marker

```
$Header$
```

into your text, for instance inside a comment. RCS will replace this marker with a string of the form

```
$Header: filename revisionnumber date time author state $
```

You never need to touch this string, because RCS keeps it up to date automatically. To propagate the marker into your object code; simply put it into a literal character string. In C, this is done as follows:

```
static char rcsid[] = "$Header$";
```

The command *ident* extracts such markers from any file, even object code. Thus, *ident* helps you to find out which revisions of which modules were used in a given program.

---

* Pairs of RCS and working files can actually be specified in 3 ways: a) both are given, b) only the working file is given, c) only the RCS file is given. Both files may have arbitrary path prefixes; RCS commands pair them up intelligently.

You may also find it useful to put the marker

Log

into your text, inside a comment. This marker accumulates the log messages that are requested during checkin. Thus, you can maintain the complete history of your file directly inside it. There are several additional identification markers; see *co* (1) for details.

Numbering of Revisions

Revisions are organized in a tree that grows from the initial revision. The tree has a main trunk, along which the revisions are normally numbered 1.1, 1.2, 1.3, ... 2.1, 2.2, ... Every revision can sprout several branches. A branch starting at revision X has number X.y, and revisions on that branch have numbers X.y.z. For example, branches starting at revision 1.3 are numbered 1.3.1, 1.3.2, and 1.3.3, and revisions on branch 1.3.1 are numbered 1.3.1.1, 1.3.1.2, 1.3.1.3, etc. Note that revisions on branches may sprout new branches, and the numbering works analogously.

Revisions and branches may also be labelled symbolically. For instance, branch 1.3.1 could be labelled "Experimental". Revisions on a labelled branch can then be identified using the branch label as a prefix. For example, revision 1.3.1.1 is identified with "Experimental.1". Of course, it is also possible to give a symbolic name to an individual revision. This label can then be used to identify the revision and as a prefix for branches starting with that revision. Note that labels are mapped to revision numbers. Labels start with letters and are followed by letters, digits, and underbars.

How to combine MAKE and RCS

If your RCS files are in the same directory as your working files, you can put a default rule into your makefile. Do not use a rule of the form .c,v.c, because such a rule keeps a copy of every working file checked out, even those you are not working on. Instead, use this:

```
.SUFFIXES: .c,v

.c,v.o:
        co -q $*.c
        cc $(CFLAGS) -c $*.c
        rm -f $*.c

prog:   f1.o f2.o .....
        cc f1.o f2.o ..... -o prog
```

This rule has the following effect. If a file f.c does not exist, and f.o is older than f.c,v, MAKE checks out f.c, compiles f.c into f.o, and then deletes f.c. From then on, MAKE will use f.o until you change f.c,v.

If f.c exists (presumably because you are working on it), the default rule .c.o takes precedence, and f.c is compiled into f.o, but not deleted.

If you keep your RCS file in the directory ./RCS, all this won't work and you have to write explicit checkout rules for every file, like

```
f1.c:   RCS/f1.c,v; co -q f1.c
```

Unfortunately, these rules do not have the property of removing unneeded .c-files. A modification of MAKE which understands RCS directories will be available soon.

Additional Information on RCS

If you want to know more about RCS, for example how to work with a tree of revisions and how to use symbolic revision numbers, read the following paper:

Walter F. Tichy, "Design, Implementation, and Evaluation of a Revision Control System," in *Proceedings of the 6th International Conference on Software Engineering*, IEEE, Tokyo, Sept. 1982.

The manual (*man*) pages associated with RCS are located in both the *UNIX User's Reference Manual* and the *UNIX Programmer's Reference Manual.* Taking a look at the manual page RCSFILE (5) should also help to understand the revision tree permitted by RCS.