SETTING UP UNIX – Sixth Edition

Enclosed are:

1. ‘UNIX Programmer’s Manual,’ Sixth Edition.
2. Documents with the following titles:

   Setting Up UNIX – Sixth Edition
   The UNIX Time-Sharing System
   C Reference Manual
   Programming in C – A Tutorial
   UNIX Assembler Reference Manual
   A Tutorial Introduction to the ED Text Editor
   UNIX for Beginners
   RATFOR – A Preprocessor for a Rational Fortran
   YACC – Yet Another Compiler-Compiler
   NROFF Users’ Manual
   The UNIX I/O System
   A Manual for the Tmg Compiler-writing Language
   On the Security of UNIX
   The M6 Macro Processor
   A System for Typesetting Mathematics
   DC – An Interactive Desk Calculator
   BC – An Arbitrary Precision Desk-Calculator Language
   The Portable C Library (on UNIX)
   UNIX Summary

3. The UNIX software on magtape or disk pack.

If you are set up to do it, it might be a good idea immediately to make a copy of the disk or tape to guard against disaster. The tape contains 12100 512-byte records followed by a single file mark; only the first 4000 512-byte blocks on the disk are significant.

The system as distributed corresponds to three fairly full RK packs. The first contains the binary version of all programs, and the source for the operating system itself; the second contains all remaining source programs; the third contains manuals intended to be printed using the formatting programs roff or nroff. The ‘binary’ disk is enough to run the system, but you will almost certainly want to modify some source programs.

Making a Disk From Tape

If your system is on magtape, perform the following bootstrap procedure to obtain a disk with the binaries.

1. Mount magtape on drive 0 at load point.
2. Mount formatted disk pack on drive 0.
3. Key in and execute at 100000

TU10	TU16
012700	(to be added)
172526	
010040	
012740	
060003	
000777	

The tape should move and the CPU loop. (The TU10 code is *not* the DEC bulk ROM for tape; it reads block 0, not block 1.)

4. Halt and restart the CPU at 0. The tape should rewind. The console should type '='.

5. Copy the magtape to disk by the following. This assumes TU10 and RK05; see 6 below for other devices. The machine's printouts are shown in italic (the '=' signs should be considered italic). Terminate each line you type by carriage return or line-feed.

```
= tmrk
disk offset
0
tape offset
100         (See 6 below)
count
1           (The tape should move)
= tmrk
disk offset
1
tape offset
101         (See 7 below)
count
3999        (The tape moves lots more)
=
```

To explain: the *tmrk* program copies tape to disk with the given offsets and counts. Its first use copies a bootstrap program to disk block 0; the second use copies the file system itself onto the disk. You may get back to '=' level by starting at 137000.

6. If you have TU16 tape say 'htrk' instead of 'tmrk' in the above example. If you have an RP03 disk, say 'tmrp' or 'htrp', and use a 99 instead of 100 tape offset. If you have an RP04 disk, use 'tmhp' or 'hthp' instead or 'tmrk', and use a 98 instead of 100 tape offset. The different offsets load bootstrap programs appropriate to the disk they will live on.

7. This procedure generates the 'binary' disk; the 'source' disk may be generated on another RK pack by using a tape offset of 4101 instead of 101. The 'document' disk is at offset 8101 instead of 101. Unless you have only a single RK drive, it is probably wise to wait on generating these disks. Better tools are available using UNIX itself.

Booting UNIX

Once the UNIX 'binary' disk is obtained, the system is booted by keying in and executing one of the following programs at 100000. These programs correspond to the DEC bulk ROMs for disks, since they read in and execute block 0 at location 0.

RK05	RP03	RP04
012700	012700	(to be added)
177414	176726	
005040	005040	
005040	005040	
010040	005040	
012740	010040	
000005	012740	
105710	000005	
002376	105710	
005007	002376	
	005007	

Now follow the indicated dialog, where '@' and '#' are prompts:

```
@ rkunix        (or 'rpunix' or 'hpunix')
mem = xxx
login:  root
#
```

The *mem* message gives the memory available to user programs in .1K units. Most of the UNIX software will run with 120 (for 12K words), but some things require much more.

UNIX is now running, and the 'UNIX Programmer's manual' applies; references below of the form X-Y mean the subsection named X in section Y of the manual. The '#' is the prompt from the UNIX Shell, and indicates you are logged in as the super-user. The only valid user names are 'root' and 'bin'. The root is the super-user and bin is the owner of nearly every file in the file system.

Before UNIX is turned up completely, a few configuration dependent exercises must be performed. At this point, it would be wise to read all of the manuals and to augment this reading with hand to hand combat. It might be instructive to examine the Shell run files mentioned below.

Reconfiguration

The UNIX system running is configured to run on an 11/40 with the given disk, TU10 magtape and TU56 DECtape. This is almost certainly not the correct configuration. Print (cat-I) the file /usr/sys/run. This file is a set of Shell commands that will completely recompile the system source, install it in the correct libraries and build the three configurations for rk, rp and hp.

Using the Shell file as a guide, compile (cc-I) and rename (mv-I) the configuration program 'mkconf'. Run the configuration program and type into it a list of the controllers on your system. Choose from:

pc	(PC11)
lp	(LP11)
rf	(RS11)
hs	(RS03/RS04)
tc	(TU56)
rk	(RK03/RK05)
tm	(TU10)
rp	(RP03)
hp	(RP04)
ht	(TU16)
dc*	(DC11)
kl*	(KL11/DL11-ABC)
dl*	(DL11-E)
dp	(DP11)
dn	(DN11)
dh	(DH11)
dhdm	(DM11-BB)

The devices marked with * should be preceded by a number specifying how many. (The console typewriter is automatically included; don't count it in the kl specification.) Mkconf will generate the two files l.s (trap vectors) and c.c (configuration table). Take a careful look at l.s to make sure that all the devices that you have are assembled in the correct interrupt vectors. If your configuration is non-standard, you will have to modify l.s to fit your configuration.

In the run Shell file, the 11/45 code is commented out. If you have an 11/45 you must also edit (ed-I) the file /usr/sys/conf/m45.s to set the assembly flag fpp to reflect if you have the FP11-B floating point unit. The main difference between an 11/40 and an 11/45 (or 11/70) system is that in the former instruction restart after a segmentation violation caused by overflowing a user stack must be handled by software, while in the latter machines there is hardware help. As mentioned above, the 11/45 and 11/70 systems include conditionally-enabled code to save the status of the floating point unit when switching users. The source for such things is in one of the two files m40.s and m45.s.

Another difference is that in 11/45 and 11/70 systems the instruction and data spaces are separated inside UNIX itself. Since the layout of addresses in the system is somewhat peculiar, and not directly supported by the link-editor *ld,* the *sysfix* program has to be run before the loaded output file can be booted.

There are certain magic numbers and configuration parameters imbedded in various device drivers that you may want to change. The device addresses of each device are defined in each driver. In case you have any non-standard device addresses, just change the address and recompile. (The device drivers are in the directory /usr/sys/dmr.)

The DC11 driver is set to run 14 lines. This can be changed in dc.c.

The DH11 driver will only handle a single DH with a full complement of 16 lines. If you have less, you may want to edit dh.c.

The DN11 driver will handle 3 DN's. Edit dn.c.

The DP11 driver can only handle a single DP. This cannot be easily changed.

The KL/DL driver is set up to run a single DL11-A, -B, or -C (the console) and no DL11-E's. To change this, edit kl.c to have NKL11 reflect the total number of DL11-ABC's and NDL11 to reflect the number of DL11-E's. So far as the driver is concerned, the difference between the devices is their addresses.

The line printer driver is set up to print the 96 character set on 80 column paper (LP11-H) with indenting. Edit lp.c.

All of the disk and tape drivers (rf.c, rk.c, rp.c, tm.c, tc.c, hs.c, hp.c, ht.c) are set up to run 8 drives and should not need to be changed. The big disk drivers (rp.c and hp.c) have partition tables in them which you may want to experiment with.

After all the corrections have been made, use /usr/sys/run as a guide to recompile the changed drivers, install them in /usr/sys/lib2 and to assemble the trap vectors (l.s), configuration table (c.c) and machine language assist (m40.s or m45.s). After all this, link edit the objects (ld-I) and if you have an 11/45, sysfix the result. The final object file (a.out) should be renamed /unix and booted. See Boot Procedures-VIII for a discussion of booting. (Note: remember, before booting, always perform a sync-VIII to force delayed output to the disk.)

Special Files

Next you must put in all of the special files in the directory /dev using mknod-VIII. Print the configuration file c.c created above. This is the major device switch of each device class (block and character). There is one line for each device configured in your system and a null line for place holding for those devices not configured. The block special devices are put in first by executing the following generic command for each disk or tape drive. (Note that some of these files already exist in the directory /dev. Examine each file with ls-I with −l flag to see if the file should be removed.)

/etc/mknod /dev/NAME b MAJOR MINOR

The NAME is selected from the following list:

c.c	NAME	device
rf	rf0	RS fixed head disk
tc	tap0	TU56 DECtape
rk	rk0	RK03 RK05 moving head disk
tm	mt0	TU10 TU16 magtape
rp	rp0	RP moving head disk
hs	hs0	RS03 RS04 fixed head disk
hp	hp0	RP04 moving head disk

The major device number is selected by counting the line number (from zero) of the device's entry in the block configuration table. Thus the first entry in the table bdevsw would be major device zero.

The minor device is the drive number, unit number or partition as described under each device in section IV. The last digit of the name (all given as 0 in the table above) should reflect the minor device number. For tapes where the unit is dial selectable, a special file may be made for each possible selection.

The same goes for the character devices. Here the names are arbitrary except that devices meant to be used for teletype access should be named /dev/ttyX, where X is any character. The files tty8 (console), mem, kmem, null are already correctly configured.

The disk and magtape drivers provide a 'raw' interface to the device which provides direct transmission between the user's core and the device and allows reading or writing large records. The raw device counts as a character device, and should have the name of the corresponding standard block special file with 'r' prepended. Thus the raw magtape files would be called /dev/rmtX.

When all the special files have been created, care should be taken to change the access modes (chmod-I) on these files to appropriate values.

The Source Disk

You should now extract the source disk. This can be done as described above or the UNIX command dd-I may be used. The disk image begins at block 4100 on the tape, so the command

dd if=/dev/mt0 of=/dev/rk1 count=4000 skip=4100

might be used to extract the disk to RK drive 1.

This disk should be mounted (mount-VIII) on /usr/source; it contains directories of source code. In each directory is a Shell file run that will recompile all the source in the directory. These run files should be consulted whenever you need to recompile.

Floating Point

UNIX only supports the 11/45 FP11-B floating point unit. For machines without this hardware, there is a user subroutine available that will catch illegal instruction traps and interpret floating point operations. (See fptrap-III.) The system as delivered has this code included in all commands that have floating point. This code is never used if the FP hardware is available and therefore does not need to be changed. The penalty is a little bit of disk space and loading time for the few floating commands.

The C compiler in /usr/source/c probably should be changed if floating point is available. The fpp flag in c0t.s should be set and C should be recompiled and reloaded and installed. This allows floating point C programs to be compiled without the −f flag and prevents the floating point interpreter from getting into new floating programs. (See /usr/source/c/run.)

Time Conversion

If your machine is not in the Eastern time zone, you must edit (ed-I) the subroutine /usr/source/s4/ctime.c to reflect your local time. The variable ‘timezone’ should be changed to reflect the time difference between local time and GMT. For EST, this is 5*60*60; for PST it would be 8*60*60. This routine also contains the names of the standard and Daylight Savings time zone; so ‘EST’ and ‘EDT’ might be changed to ‘PST’ and ‘PDT’ respectively. Notice that these two names are in upper case and escapes may be needed (tty-IV). Finally, there is a ‘daylight’ flag; when it is 1 it causes the time to shift to Daylight Savings automatically between the last Sundays in April and October (or other algorithms in 1974 and 1975). Normally this will not have to be reset. After ctime.c has been edited it should be compiled and installed in its library. (See /usr/source/s4/run.) Then you should (at your leisure) recompile and reinstall all programs performing time conversion. These include: (in s1) date, dump, ls, cron, (in s2) mail, pr, restor, who, sa and tp.

Disk Layout

If there are to be more file systems mounted than just the root, use mkfs-VIII to create the new file system and put its mounting in the file /etc/rc (see init-VIII and mount-VIII). (You might look at /etc/rc anyway to see what has been provided for you.)

There are two considerations in deciding how to adjust the arrangement of things on your disks: the most important is making sure there is adequate space for what is required; secondarily, throughput should be maximized. The RK disk (or its image) as distributed has 4000 blocks for file storage, and the remainder of the disk (872 blocks) is set aside for swap space. In our own system, which allows 14 simultaneous users, this amount of swap space is not quite enough, so we use 1872 blocks for this purpose; it is large enough so running out of swap space never occurs.

Many common system programs (C, the editor, the assembler etc.) create intermediate files in the /tmp directory, so the file system where this is stored also should be made large enough to accommodate most high-water marks. In an idle state, we have about 900 free blocks on the file system where /tmp resides, and hit the bottom every few days or so. (This causes a momentary disruption, but not a crash, as swap-space runout does.) All the programs that create files in /tmp try to take care to delete them, but most are not immune to events like being hung up upon, and can leave dregs. The directory should be examined every so often and the old files deleted.

Exhaustion of user-file space is certain to occur now and then; the only mechanisms for controlling this phenomenon are occasional use of du-I and threatening messages of the day and personal letters.

The efficiency with which UNIX is able to use the CPU is largely dictated by the configuration of disk controllers. For general time-sharing applications, the best strategy is to try to split user files, the root directory (including the /tmp directory) and the swap area among three controllers. In our own system, for example, we have user files on an RP, the root on an RF fixed-head disk, and swap on an RK. This is best for us since the RK has a faster transfer rate than the rather slow RF, and in swapping the transfer rate rather than access time is the dominant influence on throughput.

Once you have decided how to make best use of your hardware, the question is how to initialize it. If you have the equipment, the best way to move a file system is to dump it (dump-VIII) to magtape, use mkfs-VIII to create the new file system, and restore the tape. If you don’t have magtape, dump accepts an argument telling where to put the dump; you might use another disk or DECtape. Sometimes a file system has to be increased in logical size without copying. The super-block of the device has a word giving the highest address which can be allocated. For relatively small increases, this word can be patched using the debugger (db-I) and the free list reconstructed using icheck-VIII. The size should not be increased very greatly by this technique, however, since although the allocatable space will increase the maximum number of files will not (that is, the i-list size can't be changed). Read and understand the description given in file system-VI before playing around in this way.

If you have only an RP disk, see section rp-IV for some suggestions on how to lay out the information on it. The file systems distributed on tape, containing the binary, the source, and the manuals, are each only 4000 blocks long. Perhaps the simplest way to integrate the latter two into a large file system is to extract the tape into the upper part of the RP, dump it, and restore it into an empty, non-overlapping file system structure. If you have to merge a file system into another, existing one, the best bet is to use ncheck-VIII to get a list of names, then edit this list into a sequence of mkdir and cp commands which will serve as input to the Shell. (But notice that owner information is lost.)

New Users

Install new users by editing the password file /etc/passwd (passwd-V). You'll have to make current directories for the new users and change their owners to the newly installed name. Login as each user to make sure the password file is correctly edited. For example:

```
ed /etc/passwd
$a
joe::10:1::/usr/joe:
w
q
mkdir /usr/joe
chown joe /usr/joe
login joe
ls −la
login root
```

This will make a new login entry for joe. His default current directory is /usr/joe which has been created. The delivered password file has the user *ken* in it to be used as a prototype.

Multiple Users

If UNIX is to support simultaneous access from more than just the console teletype, the file /etc/ttys (ttys-V) has to be edited. For some historical reason tty8 is the name of the console typewriter. To add new typewriters be sure the device is configured and the special file exists, then set the first character of the appropriate line of /etc/ttys to 1 (or add a new line). Note that init.c will have to be recompiled if there are to be more than 20 typewriters. Also note that if the special file is inaccessible when init tries to create a process for it, the system will thrash trying and retrying to open it.

File System Health

Periodically (say every day or so) and always after a crash, you should check all the file systems for consistency (icheck, dcheck-VIII). It is quite important to execute sync (VIII) before rebooting or taking the machine down. This is done automatically every 30 seconds by the update program (VIII) when a multiple-user system is running, but you should do it anyway to make sure.

Dumping of the file system should be done regularly, since once the system is going it is very easy to become complacent. Just remember that our RP controller has failed three times, each time in such a way that all information on the disk was wiped out without any error status from the controller. Complete and incremental dumps are easily done with the dump command (VIII) but restoration of individual files is painful. Dumping of files by name is best done by tp (I) but the number of files is limited. Finally if there are enough drives entire disks can be copied using cp-I, or preferably with dd-I using the raw special files and an appropriate block size. Note that there is no stand-alone program with UNIX that will restore any of these formats. Unless some action has been taken to prevent destruction of a running version of UNIX, you can find yourself stranded even though you have backup.

Odds and Ends

The programs dump, icheck, dcheck, ncheck, and df (source in /usr/source/s1 and /usr/source/s2) should be changed to reflect your default mounted file system devices. Print the first few lines of these programs and the changes will be obvious.

If you would like to share any UNIX compatible software with others, please let us know about it. If you find bugs in the software or the documentation, again let us know.

Lastly, there is a UNIX users' group forming. To get on their mailing list, send your name(s) and address to:

Prof. Melvin Ferentz
Physics Dept.
Brooklyn College of CUNY
Brooklyn, N.Y. 11210

Good luck.
Ken Thompson
Dennis Ritchie