Second Distribution of Berkeley Software for UNIX†

Installation Instructions

By following the directions here you should be able to bring up the software on the tape in a very short period of time, ranging from an hour (if you have a standard version 6 system and can use the precompiled binaries), to about 6 hours (if you have a version 6 or 7 system which requires recompilation.)

Extracting the tape

To extract the tape you will need a 800 BPI tape drive and a file system with 12000 blocks of free space. If your tape was written with *tar* or *cpio*, then extracting the tape is straightforward. If it was written with *tp* (the default) then some work will be required to fully unpack the tape. Unless you will be using *tp* here skip the next paragraph.

First extract the file **create** from the tape by doing:

```
tp xm ./create
```

This is a shell script of **mkdir** commands. Run the script and then do

```
tp xm
```

This will take several minutes. When it completes, you will have a skeletal tape. In most directories on the tape will be a file **cont.a** which is an archive of the contents of that directory. The best thing to do is to unarchive all the files so you can look at things easily. The find command

```
find . −name cont.a −a −print
```

will print the names of all the **cont.a** files. For each such file, change into its directory and do

```
ar x cont.a
rm cont.a
```

You can omit the **rm** if you have tons of space. If you have a very old *ar*, you may have to use the *ar* in **misc**.

† UNIX is a trademark of Bell Laboratories.

Installation Preparation

The first thing to determine is which version of UNIX you are running, and how much impact modifications you have made to the system will have on the software here.

Version 7

This is the new version of UNIX which has just been released by Bell Laboratories. Most of the software here has been running on version 7 for several months at Berkeley.

The binaries on the distribution tape will NOT run on version 7, as they were compiled on a PDP-11 version 6 system. Thus you must recompile from the source for a version 7 system. This will not be hard since almost all of the software thinks it is running on version 7. The one exception is the Pascal system, which has not been run on version 7 (since we don't have version 7 on our PDP-11s yet).

Version 6

If you have a standard version 6 UNIX system then you can just use the binaries on the tape and avoid the bother of recompiling. In fact, unless you have a late-model C compiler compilation may be troublesome or impossible.

On a ‘‘standard’’ system the *getuid* system call returns the user id in the low byte of its result word. If this is the case on your system, then you should have no trouble installing the binaries supplied.†

Other Version 6

If you have a version 6 UNIX system which has 16 bit user id's (such as the systems at Berkeley) then you will have to modify the **upgrade/libretro** version 7 simulator library and recompile the programs here.

If you have a PWB/UNIX system, the binaries supplied here should work (as far as I know). If they don't the *make* and *cc* from PWB should be adequate to recompile to repair any problems.

† I have compiled the code here with a library which uses ‘‘sys sleep’’ to implement *sleep()*, rather than the later, more efficient sleep implementation using new system calls pause and alarm. If you have the latter, you can recompile programs which sleep if you wish.

New files to be added.

The following are the major files and directories which will be created as you install the tape:

/bin/csh This is the new shell. It is not placed in the directory **/usr/ucb** because it is often linked to **/bin/makesh**, which is on a different file system than **/usr/ucb** on most systems.

/etc/htmp For version 6 systems, this forms a data base which simulates version 7 environments, storing home directories and (most importantly) terminal types for each terminal.

/etc/termcap This is a data base describing terminals, and is used by the *ex* editor, and the *tset* program.

/etc/ttytype This file maps terminal ports to their types, and indicates which ports are not hard wired. The *tset* program uses this to initialize the terminal type at login.

/usr/include/retrofit On version 6 systems, a directory of header files used to simulate version 7 UNIX.

/usr/lib/Mail.help* Help files for *Mail*.

/usr/lib/Mail.rc A startup file for *Mail*.

/usr/lib/ex2.0preserve Preserve command for *ex*.

/usr/lib/ex2.0recover Recover command for *ex*.

/usr/lib/ex2.0strings Error messages for *ex*.

/usr/lib/how_p* Help files for Pascal.

/usr/lib/libretro.a Library simulating some version 7 calls on version 6.

/usr/lib/libtermlib.a Library providing terminal independent functions.

/usr/lib/me [Directory] The dynamically loaded parts of the −*me* macros are placed here.

/usr/lib/pi1.2strings Error messages for *pi*, the Pascal translator.

/usr/lib/pi1:2strings Messages for two process *pi* for 34's and 40's.

/usr/lib/pi1 Second pass of two process *pi* translator.

/usr/lib/*px_header Header files which *pi* prepends to *obj* files.

/usr/lib/tabset [Directory] Terminal initialization files for *tset*.

/usr/lib/tmac.e The −*me* macros themselves, on version 6 systems.

/usr/lib/tmac/tmac.e The −*me* macros themselves, on version 7 systems.

/usr/msgs [Directory] The *msgs* program places messages here.

/usr/preserve [Directory] Editor temporaries are preserved here after system crashes.

/usr/ucb [Directory] Most of the binaries on the tape are placed here. They can be linked elsewhere (i.e. **/usr/bin**) but the makefiles which create the tape software expect them in **/usr/ucb** so they should be left there also.

Installation procedure.

Now follow the following procedure:

1. Run the **setup** script in this directory to create needed files and directories.

2. If you have a version 6 system then run the **install** script in the directory **upgrade/include** to put a copy of the retrofitting header files in **/usr/include/retrofit**.

3. If you have a standard version 6 system (with 8 bit user id’s) then run the **install** script in the directory **bin** on the tape. Then skip to step 6.

4. If you have a non-standard version 6 system which uses 16 bit user-id’s or has other modifications which would destroy binary compatibility, then:

   a. Look at the retrofit library source directory **upgrade/libretro** and make needed changes. Recreate the library and install it. If you have *make* you can use the **makefile**; otherwise use your shell with **make.script**.

   b. Recompile the termlib library **src/termlib**, using **makefile.v6** and ‘‘make install’’, or the shell script **make.script** if you don’t have *make*.

   c. Recompile the programs in **upgrade/src** using **make.script** or **makefile**. These are versions of some programs in **src** which are different for version 6.

   d. Follow the rest of the instructions for making a version 7 compilation, using **makefile.v6** or **make.script** whenever they exist rather than **makefile**. (You can skip part *a* since you have done it already.)

5. If you have a version 7 system:

   a. Run *make* in **src/termlib**, since this makes an important library which you will need right away.

   b. Then start in the **src** directory, and run *make* there and then in each subdirectory (see below). Look at the **READ_ME** files in each directory to get an idea of what is going on. After creating the binaries ‘‘make install’’ will install them in **/usr/ucb**. Some makefiles also install things in **/usr/lib/** or **/etc**; use ‘‘make −n’’ to see what *make* will do.

      The following is a reasonable order to do the subdirectories in: (omitting Pascal for now):

      Mail, csh, ex, me

   c. Install the Pascal system. Some of the parts of the Pascal system will require special treatment on version 7 as they use the older i/o library of version 6. See the file **misc/v7pascal** for more details.

      It is not necessary to compile **eyacc** or to run *eyacc* in the **pi** and **pxp** directories; rather just use the supplied **y.tab.c** files. (The supplied **makefile**s don’t run *eyacc*.)

   d. Now prepare the utilities for the Pascal system in the directory **pascal**. Then prepare the Pascal translator **pi**, the interpreter **px** and, finally, the profiler **pxp**.

      If you have a non-separate I/D machine, or if you do not have hardware floating point, then prepare **pi0** and **pi1** rather than **pi**, and use the **px34** and **pxp34** (NOID) versions of *px* and *pxp*.† You should, on these machines:

      ```
      mv /usr/ucb/pi34 /usr/ucb/pi
      mv /usr/ucb/px34 /usr/ucb/px
      mv /usr/ucb/pxp34 /usr/ucb/pxp
      ```

6. Install the manual sections in **man** copying them to **/usr/man/manu**. If you have version 6, follow the instructions in **upgrade/man** on adapting to the different manual macros used.

† You can run *pi* (instead of *pi34* from **pi0** and **pi1**) on a non-floating point machine with separate I/D by adding a system call to simulate a mfpi instruction (see **misc/fetchi.sys**). This *pi* will run slightly faster, and allow slightly larger programs to be written.

7. Add a line of the form

   /usr/lib/ex2.0preserve −a

   to the file **/etc/rc**, before it cleans files out of **/tmp**. This will preserve the editor temporaries from **/tmp** after system crashes, and implements the editor crash recovery mechanism.†

7. So that the *msgs* program can receive messages which are sent via **mail** change, change your mail program to execute ‘‘/usr/ucb/msgs −s’’ with the message on the standard input whenever mail is sent to ‘‘msgs’’. A version 6 **mail** program which does this is in ‘‘mail.c’’ in the directory **misc**.

8. Make sure that the programs **/usr/lib/ex2.0preserve** and **/usr/lib/ex2.0recover** can write the directory **/usr/preserve**. For security, these programs should be owned by ‘‘root’’, mode 4755, and the directory **/usr/preserve** should be mode 755.

   The programs **/usr/ucb/setenv** and **/usr/ucb/tset** must be able to write **/etc/htmp**. It is wise to have **/etc/htmp** mode 644 and **sethome** and **ttytype** mode 4755 to a user who owns **/etc/htmp** (this doesn’t have to be ‘‘root’’, but it can).

9. Initialize the **/etc/ttytype** data base with the types of the terminals on your system. The file contains one line per terminal. On version 6, each line has the (one character) terminal name, and then a 2 character code. On version 7 each line has a two character code, a space, and then the (arbitrary length) terminal name. See **misc/ttytype.v6** and **misc/ttytype** for samples. The codes are defined by the file **/etc/termcap**.

10. Initialize the Mail file **/usr/lib/Mail.rc** defining any *alias* groups for distribution of mail you wish. A line of the form

    alias staff bill kurt eric

    will cause ‘‘Mail staff’’ to send copies to *bill*, *kurt*, and *eric*.

Software not installed by the above procedure

The modifications to the standard i/o library **src/libNS**, the Berkeley network **src/net**, and the *finger* program **src/finger.c** are not installed by the above procedure.

The standard I/O library modifications may require some care to make as several slightly different versions of this library are extant. See the **READ_ME** file in the **src/libNS** directory.

If you wish to run the Berkeley network, read the material in the **src/net** directory. The network is not hard to set up, but this will require a bit of preparation.

The *finger* program requires preparation of some data bases, and perhaps modifications to the *login* program as well as to *finger* itself to work. See the comments at the beginning of the program **src/finger.c** and its manual page for details.

Problems you may encounter (Version 6 only)

1. Recompiling the editor will overflow the standard compiler symbol table. See **upgrade/c** for instructions on a trivial change to make a C compiler with a bigger symbol table, which you can make available via the **−t0** flag to **cc**. Some scripts on the tape also reference a **−t1** version of the C compiler, which puts *switch* statement code out as instructions rather than as data. This makes for programs with larger text spaces but smaller per-user data. See **upgrade/c** for the C compiler change which implements this.

2. If you use the binaries on the tape, some will print times in Pacific time. They will work in your time zone if you recompile them.

3. *Csh* uses an *access* system call which is not part of a bare version 6 system. Its manual page and C interface are in the directory **misc**, as well as a file **access.sys** containing information on how to add it to your system. The *access* call is in later version 6 and version 7 systems.

† If **/usr** is a mounted filesystem, be sure it is mounted before you try to run **/usr/lib/ex2.0preserve**.

4. If you don’t get mail in the file “.mail” in your login directory you’ll have to finagle the *Mail* program to know where you do. Look at its **local.c** and **local.h** files. The *from* program in **upgrade/src** will also have to be changed.

5. If you have changed the *times* system call as per 50 changes, returning long integers for *proc_user_time* and *proc_system_time*, the supplied *csh* binary will dump when it calls *times*. You’ll have to change some declarations in the shell and recompile.

Maintenance

/etc/termcap If you get have or get terminals which aren’t described in this data base, you will have to add entries. The manual page for *termcap* explains how to write new entries.

/etc/ttytype This file tells the types of hardwired ports and which lines are dialups, and is used with *tset*. It must be edited when the system configuration changes.

/usr/preserve Editor temporaries are saved here after a system crash, when /usr/lib/ex2.0preserve is run out of /etc/rc. If no one cleans this directory out, it can get very large. You can periodically run a find command of the form

```
find /usr/preserve −mtime +7 −a −exec rm −f {} \;
```

to clean out old junk. It is usefully run by the daemon *cron*.

/usr/msgs Must be cleaned out periodically (every few months).