For People New to Both FreeBSD and Unix

Annelise Anderson
andrsn@andrsn.stanford.edu

August 15, 1997

Congratulations on installing FreeBSD! This introduction is for people new to both FreeBSD *and* Un*x—so it starts with basics. It assumes you're using version 2.0.5 or later of FreeBSD as distributed by Walnut Creek or FreeBSD.ORG, your system (for now) has a single user (you)—and you're probably pretty good with DOS/Windows or OS/2.

0.1 Logging in and Getting Out

Log in (when you see `login:`) as a user you created during installation or as *root*. (Your FreeBSD installation will already have an account for root; root can go anywhere and do anything, including deleting essential files, so be careful!) The symbols % and # in the following stand for the prompt (yours may be different), with % indicating an ordinary user and # indicating root.

To log out (and get a new `login:` prompt) type

```
# exit
```

as often as necessary. Yes, press `enter` after commands, and remember that Unix is case-sensitive—`exit`, not `EXIT`.

To shut down the machine type:

```
# /sbin/shutdown -h now
```

Or to reboot type

```
# /sbin/shutdown -r now
```

or

```
# /sbin/reboot
```

You can also reboot with `Ctrl-Alt-Delete`. Give it a little time to do its work. This is equivalent to `/sbin/reboot` in recent releases of FreeBSD, and is much, much better than hitting the reset button. You don't want to have to reinstall this thing, do you?

0.2 Adding A User with Root Privileges

If you didn't create any users when you installed the system and are thus logged in as root, you should probably create a user now with

```
# adduser
```

The first time you use adduser, it might ask for some defaults to save. You might want to make the default shell csh instead of sh, if it suggests sh as the default. Otherwise just press enter to accept each default. These defaults are saved in `/etc/adduser.conf`, an editable file.

Suppose you create a user *jack* with full name *Jack Benimble*. Give jack a password if security (even kids around who might pound on the keyboard) is an issue. When it asks you if you want to invite jack into other groups, type **wheel**

```
Login group is ‘‘jack’’. Invite jack into other groups: wheel
```

This will make it possible to log in as *jack* and use the `su` command to become root. Then you won't get scolded any more for logging in as root.

You can quit `adduser` any time by typing `Ctrl-C`, and at the end you'll have a chance to approve your new user or simply type `n` for no. You might want to create a second new user (jill?) so that when you edit jack's login files, you'll have a hot spare in case something goes wrong.

Once you've done this, use `exit` to get back to a login prompt and log in as *jack*. In general, it's a good idea to do as much work as possible as an ordinary user who doesn't have the power—and risk—of root.

If you already created a user and you want the user to be able to `su` to root, you can log in as root and edit the file `/etc/group`, adding jack to the first line (the group wheel). But first you need to practice `vi`, the text editor--or use the simpler text editor, `ee`, installed on recent version of FreeBSD.

To delete a user, use the `rmuser` command.

0.3 Looking Around

Logged in as an ordinary user, look around and try out some commands that will access the sources of help and information within FreeBSD.

Here are some commands and what they do:

`id`
: Tells you who you are!

`pwd`
: Shows you where you are—the current working directory.

`ls`
: Lists the files in the current directory.

`ls -F`
: Lists the files in the current directory with a * after executables, a / after directories, and an @ after symbolic links.

`ls -l`
: Lists the files in long format—size, date, permissions.

`ls -a`
: Lists hidden ‘‘dot’’ files with the others. If you're root, the‘‘dot’’ files show up without the -a switch.

`cd`
: Changes directories. cd .. backs up one level; note the space after cd. cd /usr/local goes there. cd ~ goes to the home directory of the person logged in—

e.g., /usr/home/jack. Try cd /cdrom, and then ls, to find out if your CDROM is mounted and working.

`view` *filename*
Lets you look at a file (named filename without changing it. Try `view /etc/fstab. :q to quit.`

`cat` *filename*
Displays filename on screen. If it's too long and you can see only the end of it, press `ScrollLock` *and use the* `up-arrow` *to move backward; you can use* `ScrollLock` *with man pages too. Press* `ScrollLock` *again to quit scrolling. You might want to try* `cat` *on some of the dot files in your home directory—*`cat .cshrc, cat .login, cat .profile.`

You'll notice aliases in .cshrc for some of the ls commands (they're very convenient). You can create other aliases by editing .cshrc. You can make these aliases available to all users on the system by putting them in the system-wide csh configuration file, /etc/csh.cshrc.

0.4 Getting Help and Information

Here are some useful sources of help. *Text* stands for something of your choice that you type in—usually a command or filename.

`apropos` *text*
Everything containing string text in the whatis database.

`man` *text*
*The man page for text. The major source of documentation for Un*x systems.* `man ls will tell you all the ways to use the ls command. Press Enter to move through text, Ctrl-b to go back a page, Ctrl-f to go forward, q or Ctrl-c to quit.`

`which` *text*
Tells you where in the user's path the command text is found.

`locate` *text*
All the paths where the string text is found.

`whatis` *text*
Tells you what the command text does and its man page. Typing `whatis *` *will tell you about all the binaries in the current directory.*

`whereis` *text*
Finds the file text, giving its full path.

You might want to try using `whatis` *on some common useful commands like* `cat`, `more`, `grep`, `mv`, `find`, `tar`, `chmod`, `chown`, `date`, *and* `script`. `more` *lets you read a page at a time as it does in DOS, e.g.,* `ls -l | more` *or* `more` *filename. The* `*` *works as a wildcard—e.g.,* `ls w*` *will show you files beginning with* `w`.

Are some of these not working very well? Both `locate` *and* `whatis` *depend on a database that's rebuilt weekly. If your machine isn't going to be left on over the weekend (and running FreeBSD), you might want to run the commands for daily, weekly, and monthly maintenance now and then. Run them as root and give each one time to finish before you start the next one, for now.*

```
# /etc/daily
output omitted
# /etc/weekly
output omitted
# /etc/monthly
output omitted
```

If you get tired waiting, press `Alt-F2` *to get another virtual console, and log in again. After all, it's a multi-user, multi-tasking system. Nevertheless these commands will probably flash messages on your screen while they're running; you can type* `clear` *at the prompt to clear the screen. Once they've run, you might want to look at* `/var/mail/root` *and* `/var/log/messages`.

Basically running such commands is part of system administration—and as a single user of a Unix system, you're your own system administrator. Virtually everything you need to be root to do is system administration. Such responsibilities aren't covered very well even in those big fat books on Unix, which seem to devote a lot of space to pulling down menus in windows managers. You might want to get one of the two leading books on systems administration, either Evi Nemeth et.al.'s UNIX System Administration Handbook (Prentice-Hall, 1995, ISBN 0-13-15051-7)—the second edition with the red cover; or Æleen Frisch's Essential System Administration (O'Reilly & Associates, 1993, ISBN 0-937175-80-3). I used Nemeth.

0.5 Editing Text

To configure your system, you need to edit text files. Most of them will be in the `/etc` directory; and you'll need to `su` to root to be able to change them. You can use the easy `ee`, but in the long run the text editor `vi` is worth learning. There's an excellent tutorial on vi in `/usr/src/contrib/nvi/docs/tutorial` if you have that installed; otherwise you can get it by ftp to ftp.cdrom.com in the directory FreeBSD / FreeBSD-current / src / contrib / nvi / docs / tutorial.

Before you edit a file, you should probably back it up. Suppose you want to edit `/etc/rc.conf`. You could just use `cd /etc` to get to the `/etc` directory and do:

```
# cp rc.conf rc.conf.orig
```

This would copy `rc.conf` to `rc.conf.orig`, and you could later copy `rc.conf.orig` to `rc.conf` to recover the original. But even better would be moving (renaming) and then copying back:

```
# mv rc.conf rc.conf.orig
# cp rc.conf.orig rc.conf
```

because the `mv` command preserves the original date and owner of the file. You can now edit `rc.conf`. If you want the original back, you'd then **mv rc.conf rc.conf.myedit** (assuming you want to preserve your edited version) and then

```
# mv rc.conf.orig rc.conf
```

to put things back the way they were.

To edit a file, type

```
# vi filename
```

Move through the text with the arrow keys. `Esc` (the escape key) puts `vi` in command mode. Here are some commands:

`x`
: delete letter the cursor is on

`dd`
: delete the entire line (even if it wraps on the screen)

`i`
: insert text at the cursor

`a`
: insert text after the cursor

Once you type `i` or `a`, you can enter text. `Esc` puts you back in command mode where you can type

`:w`
: to write your changes to disk and continue editing

`:wq`
: to write and quit

`:q!`
: to quit without saving changes

/text
: *to move the cursor to text;* `/Enter (the enter key) to find the next instance of` *text*`.`

`G`
: `to go to the end of the file`

n`G`
: *to go to line n in the file, where n is a number*

`Ctrl-L`
: *to redraw the screen*

`Ctrl-b` *and* `Ctrl-f`
: *go back and forward a screen, as they do with* `more` *and* `view`.

Practice with `vi` *in your home directory by creating a new file with* `vi` *filename and adding and deleting text, saving the file, and calling it up again.* `vi` *delivers some*

surprises because it's really quite complex, and sometimes you'll inadvertently issue a command that will do something you don't expect. (Some people actually like `vi`*—it's more powerful than DOS EDIT—find out about the* `:r` *command.) Use* `Esc` *one or more times to be sure you're in command mode and proceed from there when it gives you trouble, save often with* `:w`*, and use* `:q!` *to get out and start over (from your last* `:w`*) when you need to.*

Now you can `cd` *to* `/etc`*,* `su` *to root, use* `vi` *to edit the file* `/etc/group`*, and add a user to wheel so the user has root privileges. Just add a comma and the user's login name to the end of the first line in the file, press* `Esc`*, and use* `:wq` *to write the file to disk and quit. Instantly effective. (You didn't put a space after the comma, did you?)*

0.6 Printing Files from DOS

At this point you probably don't have the printer working, so here's a way to create a file from a man page, move it to a floppy, and then print it from DOS. Suppose you want to read carefully about changing permissions on files (pretty important). You can use the command `man chmod` to read about it. The command

```
% man chmod | col -b > chmod.txt
```

will remove formatting codes and send the man page to the `chmod.txt` file instead of showing it on your screen. Now put a dos-formatted diskette in your floppy drive a, `su` to root, and type

```
# /sbin/mount -t msdos /dev/fd0 /mnt
```

to mount the floppy drive on `/mnt`.

Now (you no longer need to be root, and you can type `exit` to get back to being user jack) you can go to the directory where you created chmod.txt and copy the file to the floppy with:

```
% cp chmod.txt /mnt
```

and use `ls /mnt` to get a directory listing of `/mnt`, which should show the file `chmod.txt`.

You might especially want to make a file from `/sbin/dmesg` by typing

```
% /sbin/dmesg > dmesg.txt
```

and copying `dmesg.txt` to the floppy. `/sbin/dmesg` is the boot log record, and it's useful to understand it because it shows what FreeBSD found when it booted up. If you ask questions on freebsd-questions@FreeBSD.ORG or on a USENET group—like "FreeBSD isn't finding my tape drive, what do I do?"—people will want to know what `dmesg` has to say.

You can now dismount the floppy drive (as root) to get the disk out with

```
# /sbin/umount /mnt
```

and reboot to go to DOS. Copy these files to a DOS directory, call them up with DOS EDIT, Windows Notepad or Wordpad, or a word processor, make a minor change so the file has to be saved, and print as you normally would from DOS or Windows. Hope it works! Manual pages come out best if printed with the dos

`print` command. (Copying files from FreeBSD to a mounted dos partition is in some cases still a little risky.)

Getting the printer printing from FreeBSD involves creating an appropriate entry in `/etc/printcap` and creating a matching spool directory in `/var/spool/output`. If your printer is on lpt0 (what dos calls LPT1), you may only need to go to `/var/spool/output` and (as root) create the directory `lpd` by typing: `mkdir lpd`, if it doesn't already exist. Then the printer should respond if it's turned on when the system is booted, and lp or lpr should send a file to the printer. Whether or not the file actually prints depends on configuring it, which is covered in the FreeBSD handbook.[1]

0.7 Other Useful Commands

`df`
shows file space and mounted systems.

`ps aux`
shows processes running. `ps ax` is a narrower form.

`rm` *filename*
remove filename.

`rm -R` *dir*
removes a directory dir and all subdirectories—careful!

`ls -R`
lists files in the current directory and all subdirectories; I used a variant, `ls -AFR > where.txt`*, to get a list of all the files in* `/` *and (separately)* `/usr` *before I found better ways to find files.*

`passwd`
to change user's password (or root's password)

`man hier`
man page on the Unix file system

Use `find` *to locate filename in* `/usr` *or any of its subdirectories with*

```
% find /usr -name "filename"
```

You can use `*` *as a wildcard in* "*filename*" *(which should be in quotes). If you tell find to search in* `/` *instead of* `/usr` *it will look for the file(s) on all mounted file systems, including the CDROM and the dos partition.*

An excellent book that explains Unix commands and utilities is Abrahams & Larson, Unix for the Impatient (2nd ed., Addison-Wesley, 1996). There's also a lot of Unix information on the Internet. Try the Unix Reference Desk[2] *.*

---

1. `http://www.freebsd.org/handbook/handbook.html`
2. `http://www.eecs.nwu.edu/unix.html`

0.8 Next Steps

You should now have the tools you need to get around and edit files, so you can get everything up and running. There is a great deal of information in the FreeBSD handbook (which is probably on your hard drive) and FreeBSD's web site[3] . A wide variety of packages and ports are on the Walnut Creek[4] CDROM as well as the web site. The handbook tells you more about how to use them (get the package if it exists, with `pkg_add /cdrom/packages/All/`*packagename, where packagename is the filename of the package). The cdrom has lists of the packages and ports with brief descriptions in* `cdrom/packages/index`, `cdrom/packages/index.txt`, *and* `cdrom/ports/index`, *with fuller descriptions in* `/cdrom/ports/*/*/pkg/DESCR`, *where the *s represent subdirectories of kinds of programs and program names respectively.*

If you find the handbook too sophisticated (what with `lndir` *and all) on installing ports from the cdrom, here's what usually works:*

Find the port you want, say `kermit`. *There will be a directory for it on the cdrom. Copy the subdirectory to* `/usr/local` *(a good place for software you add that should be available to all users) with:*

```
# cp -R /cdrom/ports/comm/kermit /usr/local
```

This should result in a `/usr/local/kermit` *subdirectory that has all the files that the* `kermit` *subdirectory on the CDROM has.*

Next, create the directory `/usr/ports/distfiles` *if it doesn't already exist using* `mkdir`. *Now check check* `/cdrom/ports/distfiles` *for a file with a name that indicates it's the port you want. Copy that file to* `/usr/ports/distfiles`*; in recent versions you can skip this step, as FreeBSD will do it for you. In the case of* `kermit`, *there is no distfile.*

Then `cd` *to the subdirectory of* `/usr/local/kermit` *that has the file* `Makefile`. *Type*

```
# make all install
```

During this process the port will ftp to get any compressed files it needs that it didn't find on the cdrom or in `/usr/ports/distfiles`. *If you don't have your network running yet and there was no file for the port in* `/cdrom/ports/distfiles`, *you will have to get the distfile using another machine and copy it to* `/usr/ports/distfiles` *from a floppy or your dos partition. Read* `Makefile` *(with* `cat` *or* `more` *or* `view`*) to find out where to go (the master distribution site) to get the file and what its name is. Its name will be truncated when downloaded to DOS, and after you get it into* `/usr/ports/distfiles` *you'll have to rename it (with the* `mv` *command) to its original name so it can be found. (Use binary file transfers!) Then go back to* `/usr/local/kermit`, *find the directory with* `Makefile`, *and type* `make all`

---

3. `http://www.freebsd.org/`

4. `http://www.cdrom.com/`

`install`.

The other thing that happens when installing ports or packages is that some other program is needed. If the installation stops with a message `can't find unzip` *or whatever, you might need to install the package or port for unzip before you continue.*

Once it's installed type `rehash` *to make FreeBSD reread the files in the path so it knows what's there. (If you get a lot of* `path not found` *messages when you use* `whereis` *or* `which`*, you might want to make additions to the list of directories in the path statement in* `.cshrc` *in your home directory. The path statement in Unix does the same kind of work it does in DOS, except the current directory is not (by default) in the path for security reasons; if the command you want is in the directory you're in, you need to type* `./` *before the command to make it work; no space after the slash.)*

You might want to get the most recent version of Netscape from their ftp site[5] *. (Netscape requires the X Window System.) There's now a FreeBSD version, so look around carefully. Just use* `gunzip` *filename and* `tar xvf` *filename on it, move the binary to* `/usr/local/bin` *or some other place binaries are kept,* `rehash`*, and then put the following lines in* `.cshrc` *in each user's home directory or (easier) in* `/etc/csh.cshrc`*, the system-wide csh start-up file:*

```
setenv XKEYSYMDB /usr/X11R6/lib/X11/XKeysymDB
setenv XNLSPATH /usr/X11R6/lib/X11/nls
```

This assumes that the file `XKeysymDB` *and the directory* `nls` *are in* `/usr/X11R6/lib/X11`*; if they're not, find them and put them there.*

If you originally got Netscape as a port using the CDROM (or ftp), don't replace `/usr/local/bin/netscape` *with the new netscape binary; this is just a shell script that sets up the environmental variables for you. Instead rename the new binary to* `netscape.bin` *and replace the old binary, which is* `/usr/local/lib/netscape/netscape.bin`.

0.9 Your Working Environment

Your shell is the most important part of your working environment. In DOS, the usual shell is `command.com`. The shell is what interprets the commands you type on the command line, and thus communicates with the rest of the operating system. You can also write shell scripts, which are like DOS batch files: a series of commands to be run without your intervention.

Two shells come installed with FreeBSD: csh and sh. csh is good for command-line work, but scripts should be written with sh (or bash). You can find out what shell you have by typing `echo $SHELL`.

The csh shell is okay, but tcsh does everything csh does and more. It allows you to recall commands with the arrow keys and edit them. It has tab-key completion of filenames (csh uses the escape key), and it lets you switch to the directory you were last in with `cd -`. It's also much easier to alter your prompt with tcsh. It makes life a lot easier.

Here are the three steps for installing a new shell:

---

5. `ftp://ftp.netscape.com`

1. Install the shell as a port or a package, just as you would any other port or package. Use `rehash` and `which tcsh` (assuming you're installing tcsh) to make sure it got installed.

2. As root, edit `/etc/shells`, adding a line in the file for the new shell, in this case `/usr/local/bin/tcsh`, and save the file. (Some ports may do this for you.)

3. Use the `chsh` command to change your shell to tcsh permanently, or type `tcsh` at the prompt to change your shell without logging in again.

Note: It can be dangerous to change root's shell to something other than `sh` or `csh` on early versions of FreeBSD and many other versions of Unix; you may not have a working shell when the system puts you into single user mode. The solution is to use `su -m` to become root, which will give you the tcsh shell as root, because the shell is part of the environment. You can make this permanent by adding it to your `.tcshrc` file as an alias with

```
alias su su -m
```

When tcsh starts up, it will read the `/etc/csh.cshrc` and `/etc/csh.login` files, as does csh. It will also read the `.login` file in your home directory and the `.cshrc` file as well, unless you provide a `.tcshrc` file. This you can do by simply copying `.cshrc` to `.tcshrc`.

Now that you've installed tcsh, you can adjust your prompt. You can find the details in the manual page for tcsh, but here is a line to put in your `.tcshrc` that will tell you how many commands you have typed, what time it is, and what directory you are in. It also produces a > if you're an ordinary user and a # if you're root, but tsch will do that in any case:

```
set prompt = "%h %t %~ %# "
```

This should go in the same place as the existing set prompt line if there is one, or under

```
if($?prompt) then
```

if not. Comment out the old line; you can always switch back to it if you prefer it. Don't forget the spaces and quotes. You can get the `.tcshrc` reread by typing `source .tcshrc`.

You can get a listing of other environmental variables that have been set by typing `env` at the prompt. The result will show you your default editor, pager, and terminal type, among possibly many others. A useful command if you log in from a remote location and can't run a program because the terminal isn't capable is `setenv TERM vt100`.

0.10 Other

As root, you can dismount the CDROM with `/sbin/umount /cdrom`, take it out of the drive, insert another one, and mount it with `/sbin/mount_cd9660 /dev/cd0a /cdrom` assuming cd0a is the device name for your CDROM drive. The most recent versions of FreeBSD let you mount the cdrom with just `/sbin/mount /cdrom`.

Using the live file system—the second of FreeBSD's CDROM disks—is useful if you've got limited space. What is on the live file system varies from release to release. You might try playing games from the cdrom. This involves using `lndir`, which gets installed with the X Window System, to tell the program(s) where to find the necessary files, because they're in the `/cdrom` file system instead of in `/usr` and its subdirectories, which is where they're expected to be. Read `man lndir`.

0.11 Comments Welcome

If you use this guide I'd be interested in knowing where it was unclear and what was left out that you think should be included, and if it was helpful. My thanks to Eugene W. Stark, professor of computer science at SUNY-Stony Brook, and John Fieber for helpful comments.

Annelise Anderson, andrsn@andrsn.stanford.edu