

# Intermediate UNIX

**Course Description:** This is an intermediate level course on the UNIX operating system. Topics covered include newsgroups and other ways to connect to other computers, job control, and using intermediate level UNIX commands.

**Prerequisites:** Previous basic experience with the UNIX operating system or the equivalent of the *Introduction to UNIX* course. A UNIX shell account is recommended.

This document has been prepared for you by W&MF staff so that you can better understand the UNIX operating system. The document is meant to serve as a supplement to the class and as a future reference for you. Not all of the information will be covered in the *Intermediate UNIX* class.

## A BRIEF REVIEW

---

In the *Introduction to UNIX* class students learn how to log in, change passwords, manipulate files and directories, and find basic information. We also give brief explanations of pine, pico and dot files. Students in the *Intermediate UNIX* course should already be familiar with the commands found in the Appendix to this document. If you would like further explanation of any of these commands, please ask the instructor or the roamer, or use the **man** command.

Remember that UNIX is an operating system. Each time you log in, you run a shell, such as tcsh or csh, that provides the programs that you will use during your session. There are quite a few different shells and they are all slightly different. One very popular shell is **tcsh**. Here is the man page definition:

**tcsh** An enhanced but completely compatible version of the Berkeley UNIX C shell, csh(1). It is a command language interpreter usable both as an interactive login shell and a shell script command processor. It includes a command-line editor (see The command-line editor), programmable word completion (see Completion and listing), spelling correction (see Spelling correction), a history mechanism (see History substitution), job control (see Jobs) and a C-like syntax.

As you learn more about and become more comfortable with UNIX, you will see that the **man** pages provide much of the information you will need to know.

## NEW COMMANDS

---

### I. FILE MANAGEMENT

1. \*, ? Wildcards are symbols that represent unknowns.  
\* = 0 or more characters  
? = exactly one character

In this list: apple apple1 apple2 orange orange1 orange2 orange10

orange\* matches all of the list items except the apples  
 orange? matches only orange1 and orange2

```
unix@hmfmac32:~ [3:35pm - 1] find *.doc
stats.doc
apples.doc
intermediate_unix.doc
unix.doc
unix1.doc
unix2.doc
unix10.doc
unix@hmfmac32:~ [3:35pm - 2] find unix?.doc
unix1.doc
unix2.doc
```

## 2. **chmod** Set access permissions on your files and directories

In order for other people to have permission to view a file in your directory, you must first set the permissions on that file to allow them to see it. You have your own permission to read write or execute a file or folder, and you can also set permissions for people in your group (those people with accounts on your server), and for all other people (people who might want to view your web page, for example).

To see what the permissions of your files are, use the list command with the option `—l`:

```
unix@hmfmac32:~ [3:35pm - 3] ls -l
total 1530
drwx----- 2 abersot      512 Jun 13 20:52 Mail/
drwx----- 2 abersot      512 Dec  2 1999 News/
-rw----- 1 abersot      704 Jun 14 14:54 resource.doc
-rw-r--r-- 1 abersot     9305 Jul 13 15:09 spaceman.jpg
```

The `d` in front of an entry means that it is a directory. The other nine spots are consecutively for the read (`r`), write (`w`), and execute (`x`) permissions for you, the group, and for other users. To change the access permissions of a file, use **chmod**, which stands for change the mode of a file. Use `u` for yourself, `g` for the group, `o` for other users, and `a` for all users. To add permissions, use a `+`, and to take away permissions, use a `-`.

For example, if I want to change the file `resource.doc` so that I can read, write and execute it, and so that the group and other users can only read it:

```
unix@hmfmac32:~ [3:35pm - 4] chmod u+x,a+r resource.doc
unix@hmfmac32:~ [3:35pm - 5] ls -l
total 1530
drwx----- 2 abersot      512 Jun 13 20:52 Mail/
drwx----- 2 abersot      512 Dec  2 1999 News/
-rwxr--r-- 1 abersot      704 Jun 14 14:54 resource.doc
-rw-r--r-- 1 abersot     9305 Jul 13 15:09 spaceman.jpg
```

The other method for changing permissions utilizes the binary and octal equivalents of these three-letter combinations. You can find out about this method in any good UNIX text, or for a brief explanation type **man chmod** at your UNIX prompt.

3. **quota -v** display file system disk quota and usage
- du <directory>** summarize disk usage

Don't be a disk hog. Keep track of the amount of space you use.

```
unix@hmfmac32:~ [3:35pm - 6] quota -v
Disk quotas for unix (uid 005):
Filesystem      usage  quota  limit
/u              15873  50000  60000
```

**quota** shows your total usage and your limit, and **du** shows the usage for a specific directory.

```
unix@hmfmac32:~ [3:35pm - 7] du mail
6565      mail
```

## II. PROCESSES

### 1. Mapping Characters

- a) **Control-U (^u)** kills the current line (a way to erase what you just wrote)

**^z** and **q** ways to get out of a process  
**q** means quit, so try that first. **^z** will be explained later. just remember that it won't quit a process, it will temporarily suspend it.

- b) **>** means output

```
unix@hmfmac32:~ [3:35pm - 8] find *.doc > docref.txt
unix@hmfmac32:~ [3:35pm - 9] ls
unix@hmfmac32:~ [3:35pm - 10] cat docref.txt
```

When you list (**ls**) your files you will see a file called docref.txt. Use **cat** to see that docref.txt is a text file that consists of a list of all of the .doc files in the directory I am in.

**>>** means append to file

```
unix@hmfmac32:~ [3:35pm - 11] cat docref.txt >> exampledoc.txt
unix@hmfmac32:~ [3:35pm - 12] tail exampledoc.txt
```

The data in docref.txt is now attached (appended) to the data in exampledoc.txt. Use **tail** to see the last lines of exampledoc.txt.

- c) | this character is a pipe. It pipes the output of one command to the input of another.

For example, if I want to display information about currently logged on users, I would use the **w** command. This command's output is often a very long list that I will not be able to read unless I can scroll up the page (scrolling within the page is not a feature that all telnet programs offer). The **more** command allows a user to see screenfulls of the output of a command.

```
unix@hmfmac32:~ [3:35pm - 13] w | more
12:28pm up 80 day(s), 23:01, 46 users, load average: 0.02, 0.04,
0.04
User      tty          login@  idle   JCPU   PCPU   what
john     pts/20       Sun 3pm 20:37    5     5   pine -z -i
aaron   pts/17       Sun 2am 10:39           pine -z -i
fsmith  pts/3        9Jul00 8days           pine -z -i
ralphk  pts/25       8:40am 3:48           pine -z -i
jimb    pts/38       11:52am 15            pine -z -i
pheller pts/39       9:00am 15            pine -z -i
root    pts/49       9:08am           pine -z -i
--More--
```

To read the next screen you press the **space bar**, and to read one more line you press **enter** (return). To stop reading press **q**.

## 2. Managing jobs

- a) **& command** (**<Cntrl-Z> (^z)**) suspends a process (job) such as pine  
Using either of these commands upon issuing the original command will push the process into the background so that you can multi-task, or run simultaneous processes.
- b) **jobs, bg** control process execution  
Use these commands to see what processes are suspended (in the background).

```
unix@hmfmac32:~ [3:35pm - 14] jobs
[1] + Suspended (signal) pine -z -I
```

```
unix@hmfmac32:~ [3:35pm - 15] bg
[1] pine -z -i &
```

**ps** report process status

```
unix@hmfmac32:~ [3:35pm - 16] ps
PID  TT  S  TIME COMMAND
24513 pts/49 S  0:00 -tcsh
24820 pts/49 T  0:00 pine -z -i
```

- c) **fg** control process execution

Use **fg** to bring a process to the foreground.

```
unix@hmfmac32:~ [3:35pm - 17] fg 1
```

**d) kill [pid]** terminate or signal processes  
In place of **[pid]**, enter the process ID you are shown when you use **ps**.

**kill —9 [pid]** Only use **kill —9 [pid]** when you cannot close a process with **kill**. **kill —9** may terminate other processes you are running along with the unwanted process.

**e) top** display and update information about top cpu processes. Give the **fg** command to come back.

```
unix@hmfmac32:~ [3:35pm - 18] top
```

```
198 processes: 193 sleeping, 3 zombie, 1 stopped, 1 on cpu
CPU states: 98.2% idle, 0.3% user, 0.8% kernel, 0.7% iowait, 0.0% swap
Memory: 384M real, 16M free, 75M swap in use, 403M swap free
PID USERNAME THR PRI NICE SIZE RES STATE TIME CPU COMMAND
24886 unix      1  33   0 2112K 1720K cpu/0  0:00 0.66% top
   1 root        1  33   0 1336K  224K sleep  3:40 0.00% init
 236 root       18  33   0 3304K 1120K sleep  2:27 0.00% syslogd
 286 root        1  33   0 1992K  608K sleep  2:25 0.00% ls3
```

Ironically, **top** is a very cpu-intensive process.

### III. CONNECTING TO OTHER COMPUTERS

#### 1. Newsgroups

**tin, rtin** a Netnews reader

```

unix@hmfmac32:~ [3:35pm - 19] rtin
rtin 1.3 950824BETA PL0 [UNIX] (c) Copyright 1991-94 Iain Lea.
Reading config file...
Connecting to agate.berkeley.edu...
Reading news active file...-
Checking for new groups...
Subscribe to new group bln.hochschulen.chipkarte (Yy/Nn) [n]: n

Group Selection (agate.berkeley.edu 60) h=help

42   59  alt.music.psychedelic  All types of psychedelic music.
44   16  alt.music.producer     Discussion about record product
45  499  alt.music.radiohead      alt.music.radiohead
46   95  alt.music.ramones        For discussion of the kings of
47   37  alt.music.smiths        Discussion about the Smiths, an
48  530  alt.music.sonic-youth   Loud enough to make you deaf.

<n>=set current to n, TAB=next unread, /=search pattern, c)atchup,
g)oto, j=line down, k=line up, h)elp, m)ove, q)uit, r=toggle
all/unread, s)ubscribe, S)ub pattern, u)nsubscribe, U)nsub pattern,
y)ank in/out

```

The man pages really do a good job of describing **tin**: tin is a full-screen easy to use Netnews reader. It can read news locally (i.e. /usr/spool/news) or remotely (rtin or tin -r option. Try **man rtin** if you want a longer description.

The first time you use **tin** or **rtin** you will not be subscribed to any news groups yet (makes sense since you haven't asked to subscribe to any). To browse the news groups that are offered, type **y** (yank). A list of hundreds of news groups will be prepared for you to look through and they will all have a **u**, for unsubscribed, next to them. Type **s** to subscribe to newsgroups that interest you. Type **r** to leave the general list of news groups and read the ones that you have subscribed to. These and all of the other commands you'll need for tin are written out at the bottom of the screen. Try out the newsgroups. They're pretty fun.

## 2. Web browsing

**lynx** a general purpose distributed information browser for the World Wide Web

```

unix@hmfmac32:~ [3:35pm - 20] lynx www.berkeley.edu
                                University of California, Berkeley (p1 of 4)
Welcome to the University of California, Berkeley. For a text-only version of the campus home
page, please follow this link.

18 Jul 2000 / 10:29 AM
Sather Gate (follow this link for online tour)
Campus News
Summertime, and the living is...hectic

-- press space for next page --
Arrow keys: Up and Down to move. Right to follow a link; Left to go back.
H)elp O)ptions P)rint G)o M)ain screen Q)uit /=search [delete]=history list
    
```

When using lynx it's a good idea to follow links that say text-only version because lynx is a text-only browser. Therefore, text-only web pages are much easier to read and navigate through than their graphical versions. The commands to use while in lynx are written out at the bottom of the screen. It isn't an ideal way to surf the web, but in a pinch it will do.

## 3. FTP

**ftp** file transfer program

```

unix@hmfmac32:~ [3:35pm - 21] ftp ocf.berkeley.edu
Connected to ocf.berkeley.edu.
220 AppleShare IP FTP Server; Service Ready
Name (ocf.berkeley.edu:abersot): unix
331 User name ok, need password.
Password:
230 User logged in, proceed
ftp>
    
```

In ftp, you can use the same commands that you use at your UNIX shell prompt to navigate around. Use **pwd**, **ls**, and **cd**, to figure out where you are and where you want to be. Type **help** for a list of commands. When you are finally in the directory that contains the file you want, type **get <filename>** to download the file to your account. Type **quit** to leave the program.

Your system administrator may have installed a more user-friendly ftp program such as **ncftp**. Learn what your options are and try them out.

#### 4. Logging on

a) **telnet <hostname>** user interface to a remote system using the TELNET protocol

**rlogin <hostname>** remote login  
Use **rlogin -l <username> <hostname>** to login with a different username than the local one (the username you are already logged in under), otherwise rlogin will default to use the same username.

**ssh <hostname>** secure shell client (remote login program)

So, what's the difference and why would you do it at all? Basically, they all do the same thing. Use these commands when you want to log in to another UNIX account from the one you are already logged into.

b) **set term=<term\_name>** resets terminal so that it will work with the remotely logged in host

It's possible that you will not have to use this command. You will only have to use it if the remote host does not have the same terminal settings as the local host.

**stty <command>** set the options for a terminal  
e.g., **stty erase ^H** set the backspace key to erase.

**stty** is a command that's good to know about in general, and we'll talk about it in the More Dot Files section.

#### 5. Finding servers

**nslookup** query name servers interactively.

Use nslookup to if you want to find the IP (Internet protocol) address or name of a server. If you put in the address, the output is the name and if you put in the name, the output is the IP address.

## IV. INTERACTING WITH OTHER USERS

### 1. **.plan** and **.project**

```
unix@hmfmac32:~ [3:35pm - 22] finger unix
```

```

Login name: unix                      In real life: UNIX teacher
Directory: /accounts/unix            Shell: /usr/local/bin/tcsh
On since Jul 1 10:11:53 on pts/22 from hmfmac32.berkeley.edu
New mail received Sat Jul 1 12:10:59 2000;
  unread since Sat Jul 1 12:05:09 2000
Project: I'm creating UNIX classes
Plan: To teach UNIX to everyone

```

From your home directory, you can use a text editor like `pico` to create files called `.plan` and `.project`. Use them to describe yourself to other users on your server.

2. **last** —`n` `Nogin_name`            shows the last `n` (number) of times a person logged in and where they were
  
3. **finger** —`person@somewhere`        displays information about remote users including their real name and their `.plan` and `.project` files, if they have them.

## V. MORE DOT (.) FILES

### 1. `.cshrc`

The `.cshrc` (stands for "C shell run control script) is a script that initializes each C shell created. In general, this file is used to set local aliases, set the prompt, set terminal characteristics, and run processes associated with each shell you might want to create. It isn't practical to explain it all here, but this will get you started.

```

unix@hmfmac32:~ [3:35pm - 23] pico .cshrc
setenv NNTPSERVER      agate.berkeley.edu
set noclobber
set history=100
set mail=/var/mail/$user

set prompt="`whoami`@\`hostname` [\!] "

                                [ Read 47 lines ]
^G Get Help   ^O WriteOut  ^R Read File  ^Y Prev Pg   ^K Cut Text
^C Cur Pos    ^X Exit      ^J Justify    ^W Where is  ^V Next Pg
^U UnCut Text ^T To Spell

```

**setenv NNTPSERVER**        sets the server where you get your news (in this case the server is `agate.berkeley.edu`)

**set history=100**            sets history to remember your last 100 commands

**set prompt=** sets your prompt (in this case it is set to equal  
 ""whoami"@hostname" [!])

Your local system administrator probably sets up a default .cshrc file for anyone who has an account on the server. Before you modify yours, make sure you create a backup copy to use in the event that you don't like your modifications.

## 2. .login

The .login file controls what you see when you log in to your account. It gets executed once at login or window startup. In general, the .login file is used to identify you to the system, to set your home directory, and to set your mail path. You can also use it to set terminal characteristics and your prompt.

```

unix@hmfmac32:~ [3:35pm - 24] pico .login
echo ""
if (-z /var/mail/$USER || ! -e /var/mail/$USER) then
  echo "You have NO mail."
else
  echo ---Last 15 messages:-----
  from | tail -15
  echo -----
endif
echo ""
biff y
  
```

The part of this that begins and ends with "echo" lets you know whether or not you have mail, and if you do, prints a list of who the last 15 messages are from. If you have this in your .login, it will print out above your initial prompt right when you login.

**biff y** sets the terminal so that it will temporarily interrupt what you are doing when you get a mail message. The interruption is a concatenated version of the message itself. This option bugs me, so in my .login it says **biff n** ("n" for no biffing).

## 3. .aliases

An alias is exactly what you might think it is - a false name. If you create a .aliases (say dot aliases) file in your home directory, you can add any kind of alias to it that you want. Everyone uses different commands, so everyone has a different .aliases file. The examples below are purely examples. Find the commands that you use the most and then create an alias to use as a shortcut.

```

unix@hmfmac32:~ [3:35pm - 25] pico .aliases
  
```

```

alias  rm      rm -i
alias  s       ls -s
alias  m       mail
alias  mv     mv -i
alias  d      fls -la
alias  h      history
alias  mr     'more -cs'
alias  lo     exit
alias  d      fls -la
alias  la     ls -al
alias  ll     ls -lF
alias  m      more
alias  bye    exit
alias  vt100  'set term = vt100;setenv TERM vt100'
alias  del    rm

[ Read 47 lines ]
^G Get Help  ^O WriteOut  ^R Read File  ^Y Prev Pg   ^K Cut Text
^C Cur Pos   ^X Exit      ^J Justify    ^W Where is  ^V Next Pg
^U UnCut Text      ^T To Spell

```

You can use a `.aliases` file for your aliases or you can add them to your `.cshrc` file.

4. **source <filename>** tells the shell to read and interpret a file even though you have already started using the shell.
  - source .aliases**
  - source .cshrc**
  - source .login**

If you modify your `.login` file the changes will not take effect unless you either log out and log back in or use the `source` command. The reason for this is because this file is only read once per session, at login. Same goes for the `.cshrc` file.

## VI. MISCELLANEOUS

### 1. More Commands

**grep** The `grep` utility searches files for a pattern and prints all lines that contain that pattern.

```

unix@hmfmac32:~ [3:35pm - 26] grep unix docref.txt
intermediate_unix.doc
unix.doc
unix1.doc
unix2.doc
unix10.doc

```

**grep** is like the command **find** only it is much more powerful. At your prompt, type **grep <pattern> <file>**. In the case above, I used "unix" as the pattern, and "docref.txt" as the file. The output was a list of all of the documents in my directory that have the word "unix" in their title.

(If you flip back a few pages, you will remember that I created docref.txt by typing **find \*.doc > docref.txt**)

**sort** sort, merge, or sequence check text files

## 2. Check spelling

**spell <filename>** checks the spelling in a text file and lists words it doesn't recognize. Don't count on it! **spell** doesn't always know the answer!

```
unix@hmfmac32:~ [3:35pm - 27] spell example.txt
ethernet
index.html
upload
yup
```

The text for this example was taken from an e-mail. The output of the command is an alphabetical list of what the computer thinks are misspelled words.

## 3. Math

**wc <filename>** displays a count of lines, words and numbers in a file  
**factor <number>** displays the prime factors of a number  
**bc** arbitrary precision arithmetic language

```
unix@hmfmac32:~ [3:35pm - 28] bc
5+5
10
6-2
4
8*8
64
10/5
2
```

# TIPS AND FAQS

---

## I. CREATING WEBPAGES

If you want to create a web page using your UNIX account there are a few necessary steps to follow:

1. In your own top-level directory (type **cd** to get there) create a directory called **public\_html** (type **mkdir public\_html**).
2. Type **cd public\_html** to get into this new directory.

3. Type **pico index.html** and create your homepage in HTML. You can use any text editor for this - pico is only one choice.
4. FTP any images you want on your page to the public\_html directory.
5. Change the permissions on index.html, any other .html pages, and all of your images so that they are world-readable and world-executable. To do this type **chmod a+rx index.html**. Then use **ls** to make sure the files are public. If they are public, there will be an asterisk (\*) next to their names.
6. Your web page address should be something like <http://server.host.edu/~username/index.html>. For example, if your username is blastoff and you have an account at ocf.berkeley.edu, your address would be <http://ocf.berkeley.edu/~blastoff/index.html>.

## II. SCRIPTS

What are they? A script is simply a series of commands that you want to execute. The .cshrc and .login files are special login scripts. You can create your own scripts to make your life easier. For example, you could script the whole process for setting permissions and creating your webpage instead of typing in the series of commands outlined above. Scripts are a little difficult to get started with so it is beyond the scope of this class to into them, but the reference list below shows some places you can look for more information.

## III. X WINDOWS, LINUX, AND OTHER "WEIRD-SOUNDING" STUFF

As you start to use UNIX more, you will hear about all kinds of strange-sounding UNIX related stuff. Common questions from beginning and intermediate level UNIX users are "what is X Windows?" "and what is linux?"

X Windows is a standard for graphical user interfaces for \*x operating systems (i.e. the different breeds of UNIX). There are actually many implementations of X Windows such as GNOME, CDE, and KDE, that are equally good to use.

Linux is a particular type of UNIX that has become very popular recently. To put it simply, it is a desktop version of UNIX that offers a wide selection of applications and also provides excellent performance and stability. Learn more about Linux at <http://home.earthlink.net/~michaelburns/>, or at any of the other zillion sites about it on the Web.

See the Resources section below for places to start looking for answers about all of this weird-sounding stuff.

## CONCLUSION

---

By this time, we hope that you have a fairly good idea of some of the commonly used features within the UNIX operating system. Even though there were plenty of features that were not covered, we hope that you will find all of the information presented useful. If you have any questions, please ask the instructor or roamer. Also, try to experiment with the topics covered, and see what you can create as well.

Remember to fill out an evaluation before you leave, and thank you for attending Intermediate UNIX.

## UNIX RESOURCES

---

- **online resources:**  
[http://www.yahoo.com/Computers\\_and\\_Internet/Software/Operating\\_Systems/Unix/](http://www.yahoo.com/Computers_and_Internet/Software/Operating_Systems/Unix/)  
<http://unix.oreilly.com>
- **Books: (O Reilly & Associates, books famous for the unique animals on their covers)**  
Learning the UNIX Operating System, 4th Edition  
UNIX in a Nutshell: System V Edition  
UNIX Power Tools, 2nd Edition

**Appendix: Introduction to UNIX command reference**

Command	Brief Explanation
<b>Directories and File Manipulation:</b>	
ls [-a] [-F] [-l] [-R] [-s]	list files in your directory
cp	copy file
mv	move file
rm	remove (delete) file
<b>Viewing Files:</b>	
Cat	concatenate-displays a file on your screen
More	displays a file with page breaks
Head —number	displays the first (number of) lines in a file
Tail —number	displays the last (number of) lines in a file
<b>Directories:</b>	
Pwd	tells what your current working directory is
Cd	change directory
Mkdir	make directory
Rmdir	remove (delete) directory
<b>Finding Information:</b>	
Finger	get account information on another user
Ping	sends a message to a remote machine to determine if it is alive (up)
which	shows you which program in your path will be called with a command that you choose
whereis	searches the machine and displays any files that begin with a command you choose
who	shows who is currently logged in on the local machine
man	displays the manual page for a command
whoami	tells you who you are logged in as
<b>Dot (.) Files:</b>	
.plan	if created, a .plan adds its contents to the bottom of your finger
.project	if created, a .project adds its contents before your .plan
.signature	if created, a signature file adds its contents to the bottom of any email you send
<b>Programs:</b>	
pico	a simple text editor
pine	an email program for unix
mail, elm	other (not as user friendly as pine) email programs for unix
vi, emacs, jove	more powerful but less user friendly than pico text editors
<b>other useful commands:</b>	
passwd	changes your password
logout, quit, exit	end your unix session
cal	shows the current's month's calendar
date	shows today's date