UNIX User Basics

Joshua Uziel <uzi@csua.ucla.edu> − 1/17/99

Getting Help

man − Look at manual page, do this to see almost any command’s options − example: "man ls" − **DO THIS WITH MOST OF THESE COMMANDS!!!**

Listing and Moving Files and Directories

ls − LiSt files − "−a" lists all files, "−l" give a long DOS−dir−like output − example: "ls −la"
cp − CoPy − copy file(s) from one location to another − example: "cp *.c another_directory/"
mv − MoVe − move file(s) from one location to another, same as cp except it removes files afterwards
rm − ReMove − delete files − example: "rm *.o"
cd − Change Directory − example "cd /tmp"
mkdir, rmdir − MaKe DIRectory and ReMove DIRectory
ln − LiNk − make a link to a file − two types of links, normal and symbolic − example: "ln −s original newlink"
Special Directories − ’.’ is current, ’..’ is current’s parent, ’~’ is home, and ’/’ is the root (or top) of all directories

Redirection and Shell Stuff

| − Pipe, one of UNIX’s best features, used to send output from one program to be input to another − example: "ls −l | grep uzi"
>, >>, < − Redirect output of a program to a file, append output to end of a file, and get input from a file − example: "./a.out < input > ouput"
***, ?** − Wildcards − used to specify any number (including zero) of any character for ’*’, and any one character for ’?’ − example: "ls *.?"
$var − An environment variable, where "var" is the name and a value can be seen using "echo $var"
set, setenv − used to set environment varibles, depending on shells.

Managing File Permissions

Sample "ls −l /bin/ls" output: **−rwxr−xr−x   1 root   root   29980 Apr 23 1998 /bin/ls**

First ten characters are the file’s permissions. First is file type, ’−’ for normal, ’d’ for directory, ’l’ for symlink, etc.
Three sets of three characters of "rwx", or Read, Write, Execute permission − first for owner, then group owner, then everyone.
Then we have number of links to file, owner of the file, group owner of file, file size, date it was last modified, and the file’s name.

chmod − Change a file’s mode or permissions − one way is to use octal numbering − example: "chmod 751 /bin/ls" for rwxr−x−−x
chgrp − Change a file’s group − example: "chgrp users /bin/ls"
chown − Change a file’s owner − example: "chgrp uzi /bin/ls"
umask − Set default file permissions (generally, do the opposite, or "777 minus what you want")

Process Management

ps − Get a listing of running "processes" or programs − example: "ps −aux"
kill − Kill a running process − example: "kill −CODE PID", where CODE is an optional kill code, and PID is the process ID #.
& − Run a program in the background − example "netscape &"
CTRL−Z − Suspend a running program
CTRL−C − Kill a running program
fg, bg − Put a suspended program in the foreground or background
jobs − List running programs of this terminal − each has a number that can be refered to as %#, or %1 for number one (useful with fg, bg and kill)
nice − Make a program use less computer time
nohup − Let a program run after you log out (NO Hang−UP)

Viewing Text Files

cat − Concatinate and display files, used to output a file’s contents without pausing − example: "cat textfile"
more − Like cat, but pauses every screen−full − example: "more prog.c"
less − Like more, but more powerful
head, tail − View the beginning or ending of a file, given a number option with display that many lines − example: "tail −25 /var/spool/mail/uzi"
wc − Get statistics of how many lines, words and characters in a file − example: "wc file.txt"

Searching and Comparing

find − Find a file, second argument is where to search from − example: "find . −name ’*.c’ −print"
grep − Look for text in a file − example: "grep variable *.c"
cmp − Compare two files − example: "cmp file1 file2"
diff − Output difference between two files − example: "diff prog.c.orig prog.c > prog.c.diff"

Printing − Depending on which Unix you’re using (BSD type, SYSV type) is which you may be using

lpr, lp − Print a file −examples: "lpr −P printer file" or "lp −d printer file"
lpq, lpstat − Get statistics on a printer(s)
lprm, cancel − Cancel a print job

Finding and Communicating with other Users

finger − Get information on another user on the system − example: "finger uzi"
w, who − Find out who else is on a system
write − Write a message to a user (user CTRL−D to stop) − example: "write uzi"
talk − Talk to a user (CTRL−C to stop, and there’s also "ytalk" which is an enhanced version) − example: "talk uzi"
mesg − Use "mesg y" or "mesg n" to allow or disallow others to write and talk to you

Remote Commands

telnet − Open a connection on another system − example: "telnet fire.csua.ucla.edu"
ftp − Retrive files with the File Transfer Protocol − example: "ftp ftp.kernel.org"
rlogin − Similar to telnet, but if you have a ".rhosts" file with allowed machines, you can log in without a password
rsh − Also using ".rhosts", lets you run one command on another machine without logging in
rcp − Another that uses ".rhosts" and lets you copy files from another machine

Miscellaneous

alias − Aliases can be used to make a shortcut for a common command − example: "alias dir ’ls −l’" or "alias dir=’ls −l’", depending on shell
at − Lets you have a program run "at" a certain time
crontab − Schedule regular programs to run at certain times (some let all users have their own crontab file)
cal − Print a calendar for a given month − example: "cal 8 1976" or just "cal" for this month
date − Print current time and date
df − Find out how much free disk space is available − example: "df −k ." for current directory in kilobytes
du − Find out disk usage of a file or directory − example: "du −ks ~" for size of your entire home directory
echo − Repeats or echoes the argument − example: "echo hello" prints "hello"
tar − Tape ARchive − allows you to pack amny files together − example: "tar −xvf file.tar" to extract
gzip, gunzip − Compress or uncompress *.gz files − example: "gzip file.tar"
passwd − Change your password on a system
spell − Spellcheck a text file against system dictionary − example: "spell paper.txt"
sort − Sort contents of a file
time − Get runtime of a program − example: "time ls −l"
uname − Get info on a machine − example: "uname −a"
uptime − Get how long a machine has been up
which, whence, where − File out which program you’ll be running, or where all occurences of a program is in your path − example: "which ls"
vi, pico, emacs, jed, joe − Text editors.
pine, elm, mutt, mailx, mail − Email programs

Using Vi

Vi is a powerful editor common to all UNIXes. Learning it would be extremely valuable. Vi got it’s name from being a "VIsual" editor, and although it seems quite archaic, it is very useful. It’s actually built on a line editor called "Ex", which you use through commands starting with a colon (’:’). Getting to know Ex is a good way to become a Vi master, but not required to use Vi well.

Vi has two modes: command and insert mode.

Command Mode

In command mode, you can invoke insert mode (see below), issue editing commands, move the cursor, invoke ex commands, invoke a UNIX shell, and do file operations.

Insert Mode

Is the other mode, in which you can enter new text. You return to the command mode with the ESCAPE key. These commands take you into insert mode:

a − Append after cursor
C − Change to end of line
o − Open a line below current line
S − Substitute entire line
A − Append at end of line
i − Insert befor cursor
O − Open a line above current line
R − Begin overwriting text
c − Begin change operation
I − Insert at beginning of line
s − Substitute a character

List of Commands

c, d, y − Change, Delete, Yank operators ... these are important and can be used with other commands to make a wide assortment of possibilities.

h, j, k, l − Move left, down, up, right (Arrow keys tend to work as well)

w, W; b, B; e, E − Forward a word; Backards a word; End of word

), (; }, {;]], [[− Beginning of next, current sentence; paragraph; section

$, 0; ^; −, + − Last, first position of current line; first nonblank character of current line; First character of previous, next line

/*text*, ?*text* − Search forward, backaward for *text* ... doing without *text* searches for last thing searched for

% − Find match of current parenthesis, brace, or bracket

CTRL−G; *n*G, :*n* − Display current line number; Go to line *n* (**G** alone goes to last line)

m*x*, ’*x* − Mark current position with character *x*, go to mark *x*

cw, cc, C − Change word, change line, change text to end of line

dd, *n*dd, D, dw, d/*text*, dG − delete current line, delete *n* lines, delete to end of line, delete current word, delete to *text*, delete to end of file

p, P − insert last delete after, before cursor

u, U, . , ~ − Undo last change, Restore current line, repeat last change, reverse case

x, X − Delete the character on, before cursor

Y, yy, *n*yy − Copy current line, copy *n* lines

:w − write file (:w! Writes overriding protection)

:q − quit (:q! Quites discarding changes)

ZZ, :x, :wq − write and quit

:e *file* − Edit *file*

:r *file* − Reads in contents of *file* after cursor

J − Join two lines

<<, >> − Shift current line one shift width (default 8 spaces) to the left, right

>} − Shift right to end of paragraph

<% − Shit left until matching parenthesis, brace, or bracket

:0,$s/*old/new/*gc −Substitute *old* text for *new* text (gc = globally and confirm) throughout the file

(There are by far more commands than this, but a complete listing would be huge.)

There are also many optional settings that you can change using **:set**. Typing **:set all** will list all options and their settings. Default settings can be put into a **$HOME/.exrc** file.

I would highly recommend getting a good UNIX book or two. UNIX is huge, has a lot to learn and remember, and takes years to get good at. Don’t be discouraged by the fact that there’s so much to learn... take is as a rewarding challenge. There are some good UNIX websites, such as http://www.ugu.com to look at, as the Internet holds a wealth of UNIX knowledge. Some good books are O’Reilly’s nutshell books (such as "UNIX in a Nutshell" and "Learning UNIX") and the "UNIX System Administration Handbook". (Generally, books published by O’Reilly, Prentice Hall, and Addison−Wesley are good, and others aren’t ... but there are a few exceptions.) Install Linux and play around. Ask for help from someone in the LUG (Linux User Group) on campus if you get stuck.

As you become more comfortable with UNIX, you’ll be able to make use of it’s great power. Trust me, it’s worth it. Good luck!