

Unix Commands

Access Rights (AFS)

fs la dirname	list uniqnames/group names and rights in specified directory
fs sa . uniqname rl	grant read and list rights to individual in current directory
fs sa . uniqname none	remove access rights for individual in current directory
racl uniqname rl	grant read and list rights to individual in current directory and subdirectories

Aliases

edit file .cshrc.aliases (always realias after modifying)
alias alias_name cd /destination_directory

Change Directories

cd	cd to root/home directory
cd dirname	cd into directory cd /fusion from /group/boehnke/ changes directories to to /group/boehnke/fusion
cd ..	cd back one directory cd .. from /group/boehnke/fusion/ goes to /group/boehnke
cd ../../	cd back two directories cd ../../ from /group/boehnke/fusion/Genotypes goes to /group/boehnke/
pwd	display current working directory

Compare Files

diff	diff file1 file2 lines starting with < show lines in file1 different than file2 lines starting with > show lines in file2 different than file1 -i ignore case -B ignore blank lines -y output in 2 columns diff3 file1 file2 file3 – show differences in 3 files diff file1 file2 > diff.txt – output diffs to text file
tkdiff	tkdiff file1 file2 (must use Exceed if using Windows) ! marks differing lines

Compress/Uncompress Files

bzip2	bzip2 filename – zips files smaller than gzip
bunzip2	bunzip2 filename – unzips bzip file
gzip	gzip filename – zips file -r gzip recursively (subdirectories) -n gzip with specified level of compression. -1 is fast but less efficient compression and -9 is slow but most efficient compression (default is -6)
gunzip	gunzip filename – unzips gzip file gunzip in Perl (-c writes output to stdout) <pre>if (\$file1 =~ /\.gz\$/) { open(IN, "gunzip -c \$file1 ") die "Unable to uncompress '\$file1': \$!\n"; }</pre>
tar (tar files)	tar -options infile.1 infile.2 > output.file -c create -f file -t table of contents -v verbose
tar (untar files)	tar -options tarfilename -C specify directory to untar the file to tar xvf -C directory newtarname -f file -x extract -v verbose -z uncompress zipped tar file

Copy/Delete/Move Files

cp	cp file1 file2 – file1 is kept and file2 is created cp directory/filename . – copies file to current directory
mv	mv file1 file2 – file1 is renamed as file2
rm	rm filename – delete filename

Create Directories

mkdir	mkdir dirname mkdir -p – creates parent directory and subdirectories mkdir -p test/one/two creates test, test/one, and test/one/two
-------	---

Create Directories (cont.)

mkdir (cont.)

mkdir -v – verbose, prints message for each created directory

mkdir -v -p test/one/two

prints

mkdir: created directory `test'

mkdir: created directory `test/one'

mkdir: created directory `test/one/two'

mkdir -m – mode, set permissions for directory

mkdir -p -m 740 test/one

test receives existing permissions – one is group readable

Remove Directories

rmdir

rmdir dirname

/bin/rm -r dirname removes directories containing files

Directory/Memory Usage

df -h

list available space on fdata (compute19) and clusters

du

summarize disk usage of each file

-h prints in human readable format (50K, 23 M)

-s summarize

find

find -name "filename" (searches current directory)

find -iname "filename" (same as -name, case insensitive)

find -not -iname "filename" (list files other than filename)

find . -print

prints all files in current directory and subdirectories

find . | xargs grep "rs1004454"

prints file names (& location) containing rs1004454

find . -size +128M

lists all files (and locations) larger than 128 MB

find -maxdepth 2 -name "filename"

looks in current directory and one subdirectory

free

display memory usage

-m displays free memory size in MB

-t displays a line containing the total memory in MB

fs lq

fs lq dirname – list quota used in each AFS volume

ls

list files in directory

-a list hidden files

-l list dates, sizes of files, etc.

-F puts "/" after directories

-lt lists date/time stamps starting with recent files

Directory/Memory Usage (cont.)

mon	displays current information about MOSIX nodes as bar chart (first two are fantasia and snowwhite) For more information about the cluster, see http://csg.sph.umich.edu/docs/cluster/ (done by Terry Gliedt)
top	displays ongoing look at processor activity in real time
uptime	displays current time, how long the system has been running, how many users are currently logged in, and the system load averages for the past 1, 5, and 15 minutes
w	displays who is logged in and what they are running (header line is same line displayed with uptime)

Permissions

chgrp	change default group ID of file (owner of file can do) chgrp pritzker variables.txt – group pritzker can read chgrp fusion variables.txt – group fusion can read chgrp -c pritzker –variables.txt outputs what was changed home0>changed group of `variables.txt' to pritzker chgrp -R – recursive on files and directories
chmod	change access permissions chmod +r filename – grant permission to read the file chmod +w filename – grant permission to write/delete the file chmod +x filename – make file executable chmod u+rx directory/ – owner has full rights chmod ugo+x – owner, group, others can execute chmod g+rx directory/ – group has read and execute rights chmod o-rwx * – others others have no rights chmod g=r – group can read chmod a+X – add executable permissions to others for files already executable by the owner (avoid text files recursively set to executable) chmod n1n2n3 – n1 is owner, n2 is group, n3 is others read = 4, write = 2, execute = 1 (add numbers to 7) chmod 750 – owner has full rights, group has read and execute, others have none

Permissions (cont.)

chmod (cont.)

chmod 755 – owner has full rights, group has read and execute, others have read and execute

chmod 775 – owner has full rights, group has full rights, others have read and execute

chmod n1n2n3 -R – chmod recursively (subdirectories)

Directories must have executable permissions to be read by group.

chown

change group ownership of file/directory

chown :fusion filename/directory (fusion group)

chown :pritzker filename/directory (pritzker group)

-R – chown recursively (chown -R :group Directory/)

umask

read = 4, write = 2, execute = 1 (subtract numbers from 7)

create directories with rwxr-xr-x – set umask to 022:

owner read, write, execute: 7-4-2-1=0

group read, execute: 7-4-1=2

others read, execute: 7-4-1=2

002 – owner is all, group is all, others are read, execute

007 – owner is all, group is all, others are none

027 – owner is read, write, execute – group is read, execute – others are none

077 – owner is all, group is none, others is none

set umask in .cshrc.aliases (umask nnn)

execute is set on directories; execute has to be manually changed on files after they are created (read and write is unchanged)

Printing (use aliases)

enscript -FCourier10 -fCourier7 -2r -h -G -P33

(2 portrait pages side by side with date, time, title, etc.)

a2ps --landscape -2 --sides=1 --line-numbers=1

(same format but prettier with line numbers)

a2ps -1 --line-numbers=1

(same format but 1 portrait page)

display available printers - cd /etc and view “printcap”

lpq

lpq -P33 (see queued jobs on HP 8100)

lprm

lprm -P33 job# (cancel print job on HP 8100)

Tokens

Default AFS authentication (tokens) is 25 hours

>tokens

Tokens held by the Cache Manager:

User's (AFS ID 47778) tokens for afs@sph.umich.edu

[Expires Aug 28 18:44]

To extend tokens to 7 days:

kinit -l 7d; aklog (-l – lifetime)

The shell script afs.sh issues “kinit; aklog”

View Files

head

head filename – view first 10 lines of file

head -20 filename – view first 20 lines

head -20 filename > newfile – output to file

less

less filename – display output one screen at a time

more

more filename – display output one screen at a time

tail

tail filename – view last 10 lines of file

tail -20 filename – view last 20 lines of file

tail -20 filename > newfile – output to file

grep

grep texttofind filetogrep (use quotes for more than 1 word)

-A # – outputs # lines from file after grepped output

(-A 2 outputs 2 lines following grepped text)

-B # - outputs # lines from file before grepped output

(-B 2 outputs 2 lines preceeding grepped output)

--color – outputs grepped text in red

Use quotes and backslashes to grep for periods,

commas, and dashes (grep ‘\.’ – grep ‘\,’ – grep ‘\.’)

grep ‘^SNP’ filetogrep – output lines beginning with SNP

grep ‘SNP\$’ filetogrep – output lines ending with SNP

grep -i – ignore case of pattern

grep -o ‘pattern’ – print the part of matching lines that contain ‘pattern’

grep -v ‘pattern’ – print lines not containing pattern

grep -n ‘pattern’ – prints line number with the pattern

View Files (cont.)

grep (cont.)

grep for lines containing pattern in tab delimited file

```
File contains: one      6      two
                three   10     four
                five    15     six
                seven   6      eight
```

grep Ctrlv Tab 6 CtrlV filename (Ctrlv is Control + v and Tab is the Tab key) returns

```
one      6      two
seven    6      eight
```

grep (use file)

grep -f file_of_text_to_find filetogrep (-f is filename)
grep -wf mylist filetogrep > outputfile (-w is word/string)

grep (wc)

grep texttofind filetogrep | wc -l
(gives count of lines containing pattern)

egrep

grep on more than 1 word/string
egrep -w 'rs12739426|rs640742|rs10889577' filetogrep
(can also use -A2, -B2, --color)

zgrep

grep for text in compressed or gzipped file (Linux only)

Other

↑

see/use/modify previous commands

&

“command &” keeps the window the command was issued in available for use

>

put command output into a file (see cat, head, tail)

|

pipe commands together (see cut)

;

equivalent to a return – command; command; command;
cd directory; ls; pico filename

&&

combine two commands where the second one is run only if the first one is successful.

```
cd sample && grep s1 stage1.txt
```

returns

```
sample: No such file or directory (directory is Sample)
```

```
cd Sample && grep s1 stage1.txt will run
```

Other (cont.)

<code>\</code>	use for long input commands (helps with readability) echo This could end up being \ ? a really really long line \ ? to type! This could end up being a really really long line to type! (The entry will appear on a single line with ↑) or cd directory1/directory2/directory3/directory4/ && \ zip *.txt
tab	use to auto-complete command
cal/ncal	cal – show calendar with current date highlighted cal -3 – show previous month, current month, and next month cal -m – show calendar with Monday as first day of week cal -j – show Julian days (one-based starting January 1) cal 12 2008 – show December 2008 calendar ncal – show calendar in vertical format
cat	cat file1 file2 > newfile – newfile contains file1 and file2 cat -n file1 file2 > newfile – number lines in newfile cat file – display contents of file find DOS characters in a Unix file cat -v filename more marker,panels^M rs340874,SNP0422_Geno7_Metsim^M rs4675095,SNP0422_Geno7_Metsim^M rs11708067,SNP0422_Geno7_Metsim^M ^M is DOS newline
clear	clear terminal screen
control characters	ctrl c – kill job ctrl d – end of file ctrl s – stop output ctrl q – continue output ctrl z – suspend current job
cut	cut -f1 -d, filename (-d ‘ ’ for space delimited file) outputs column 1 from comma delimited file cut -f1-3,5 -d, filename outputs columns 1-3 and 5 from comma delimited file

Other (cont.)

cut (cont.)

cut -f1-6,8- filename

outputs columns 1-6, and 8 to the end of the row

cut -f1 -d, filename | uniq -c

outputs column1, prefixes lines by # of occurrences

cut -f1 -d, filename | sort | uniq -c

same output but sorted by column 1

cut -c 1-8 filename cuts columns 1-8

cut default delimiter is tab

date

display date and time (24 hour time)

dos2unix

dos2unix filename – convert DOS file to UNIX file

expand

converts tabs to spaces

expand filename > newfilename

expand -1 filename > newfilename

converts tab characters to a single space

(default is 8 spaces including the column text)

unexpand converts spaces to tabs

file

Determine file type (or directory)

file shell_script_name returns

shell_script_name: C shell script text executable

file perl_file_name returns

perl_file_name: perl script text executable

file text_file_name returns

text_file_name: ASCII text

file sybolic_link returns

link: symbolic link to 'linked file'

groups

prints group/groups a user is in

groups ppwhite returns

ppwhite : fusion f-dfo pritzker

history

view last 100 commands issued

!history# will repeat command

join

join files on common field (must sort files on join field)

join -j1 2 -j2 1 -e 'null' -t, -a1 file1 file2 > output.csv

Other (cont.)
join (cont.)

-j1 2 – file1, field2
-j2 1 – file2, field1
-e 'null' – replace empty fields with null
-t, – -t is field delimiter (comma in this example)
-a1 – output lines only matching file1
-v1 – output lines in file 1 but not in file 2
-v2 – output lines in file 2 but not in file 1
-o list – specify fields to output (if not all fields are used)
-o 0,1.1,1.2,1.6,2.2,2.3,2.4 – join field, file1 field 1, file1
field2, file1 field6, file2 field2, etc. (no spaces)

kill

kill -9 process#
(type “ps -f to see jobs and process number)

ln -s (symbolic link)

ln -s source_file new_file
source_file is existing file, new_file is name of symbolic link
(full path to source_file is required)

man

Unix man(ual) – “man grep” will display manual for grep

nl

nl filename – numbers lines in filename

paste

reattach columns of data
paste file1 file2 > newfile
paste -d, file1 file2 > newfile (comma delimiter)
paste default delimiter is tab

ps2pdf

ps2pdf ps_filename – convert postscript file to pdf file

rename

bulk rename with Perl one-liner

rename chr01.txt, chr02.txt, etc. to chr01.csv, chr02.csv, etc.

```
rename -n 's/txt/csv/' chr*.txt
prints
chr01.txt renamed as chr01.csv
chr02.txt renamed as chr02.csv
```

-n – no action, show what files will be renamed to
-v – verbose, print names of files successfully renamed
-f – force, overwrite existing files

```
rename -n 's/c/snp_c/' chr*.txt
prints
chr01.txt renamed as snp_chr01.txt
chr02.txt renamed as snp_chr02.txt
```

Other (cont.)

run commands in
subshell with ()

redirect or collect output of a series of commands

```
( cd 080213/Association/Stage1/ ; cat out* > both.out ; \  
uuencode both.out myfile.out ) | mailx ppwhite@umich.edu \  
-s "Test Unix Command"
```

sends an e-mail with the subject "Test Unix Command"
with the file myfile.out attached

screen

create any number of virtual windows within the same
terminal window. Type 'screen' to create an initial window.

screen (cont.)

ctrl-a c – create a window

ctrl-a spacebar – go back to first window (this command
pages through open windows)

ctrl-a 0 – go to window 0 (ctrl-a 2 – go to window 2)

ctrl-a backspace or del – go to previous window

ctwl-a w – list windows

screen -d – detach a session and run the session independent
of your login window

screen -d -r reattach a session (if necessary, detach it first)

screen -list – shows status

to exit a window, exit from the shell in that window and the
window is killed.

if the screen appears to freeze, type ctrl-q (ctrl-s causes the
terminal window to stop)

sed

```
sed -e 's/X/x/g' infile > outfile
```

substitute x for X in outfile (see also tr)

```
sed -e 's/\-/\N/g' infile > outfile
```

substitute \N for - in outfile for SQL input

```
sed -e 's/\.\N/g' infile > outfile
```

substitute \N for . in outfile for SQL input

sed works with strings of any length (tr on single characters)

seq

print numeric sequence

```
seq 2 returns
```

1

2

```
seq 20 22 returns
```

20

21

22

Other (cont.)

sleep	delay for a specified amount of seconds (default) sleep 60 – delay for 1 minute sleep 120 – delay for 2 minutes sleep 1m – delay for 1 minute sleep 2m – delay for 2 minutes sleep 1h – delay for 1 hour
sort	sort filename > sorted.output sort -g – numeric sort with numbers in exponential format sort -n – numeric sort sort -r – reverse sort sort -k# – sort on column number (k2 is column 2) sort -c – see if a file is sorted (error message returned if not) sort -u – suppress duplicate lines sort -k2,2g – forces sorting in exponential format (may be necessary for large files) sort -t, – sort using comma field delimiter
touch	touch filename – creates empty file filename
tr	tr '[X]' '[x]' < infile > outfile substitute x for X in outfile (see also sed) tr works on single characters (sed on strings of any length)
uniq	uniq filename > uniqued.output (best if used after sort) uniq -c – prefixes lines with # of occurrences uniq -d only shows duplicate lines
uuencode	uuencode (uu = Unix to Unix) uuencode input_file output_file converts binary file into text file that can be used as e-mail attachment
which	which -program_name/command displays location of program/command
who	displays usernames of who is logged in
xargs	build and execute command lines from standard input find -name "README*" xargs grep "pedigree" returns list of filenames beginning with README that include the word "pedigree"

Other (cont.)

xargs (cont.)

```
find . | xargs grep "rs1004454"  
prints file names (& location) containing rs1004454  
ls | xargs file  
prints file/directory names on left and types on right  
(directory, ASCII English text, C shell script, etc.)
```

General Unix/Linux/Perl Web Pages:

fosswire.com/2007/08/02/unixlinux-command-cheat-sheet/
computerhope.com/unix.htm
digilife.be/quickreferences/QRC/The%20One%20Page%20Linux%20Manual.pdf
linux.about.com/?once=true&
pixelbeat.org/cmdline.html

Perl One-Liner Web Pages:

sial.org/howto/perl/one-liner/
sysbio.harvard.edu/csb/resources/computational/scriptome/UNIX/
ajs.com/ajswiki/Perl_one-liners
perlhowto.com/one_liners
unixguide.net/unix/perl_oneliners.shtml

Screen Cheat Sheet:

catonmat.net/blog/wp-content/plugins/wp-downloadMonitor/user_uploads/screen.cheat.sheet.pdf

SED One-Liners:

sed.sourceforge.net/sed1line.txt

AWK Tutorial:

http://csg.sph.umich.edu/docs/awk_hints.pdf