Filesystem

bzip2 [*opts*] [*filepattern*] ·**bzip2 Compression (better)**
cd [-] [*directory*] ·**Change directory**
- :Change to the previous directory you were in
chmod [*opts*] <*mode*> <*filepattern*> ·**Change permissions**
-R :Change permissions recursively
chown [opts] <*user*>[.*group*] <*file*> ·**Change ownership**
-R :Change ownership recursively
cp [*opts*] <*from*> <*to*> ·**Copy files and directories**
-i :Interactive mode. Prompt before overwriting
-p :Preserve file permissions and ownership
-R :Copy directories recursively
df [*opts*] [*device name*] ·**Print filesystem usage info**
-a :Show all filesystems
-h :Human readable format/Quantify byte information
-i :Show inode usage info
du [*opts*] [*pattern*] ·**Show space usage on files and dirs**
-c :Produce a grand total for all arguments
-h :Human readable format/Quantify byte information
-s :Summarize. Only show a total for each argument
find <*path*> [*opts*] ·**Search for a file**

Learning find, once and for all!

Find all non-world readable html/htm files and change their user ownership to fred using chmod:

```
find / -type f -name '*.html' -o -name '*.htm' -a
  -perms -644 -exec chown fred {} \;
```

gzip [*opts*] <*filepattern*> ·**Compress a file or files**
-1..9 : Set compression level. 9=highest, 1=lowest
-d :Decompress file. Same as the **gunzip** command
-v :List the statistics for a compressed file
ln [*opts*] <*tofile*> <*linkfile*> ·**Create a sym/hard link**
-s :Create a symbolic link between files (alias name)
-f :Force creation, even if the link file exists
ls [*opts*] [*pattern*] ·**List file and directory entries**
-a :List all files including ones that start with '.'
-d :List directories themselves, not their contents
-l :Long list. Shows permissions and modified time
-r :Recusively list files in directories
-S :Sort output by file size
-h :Human readable format/Quantify byte information
-X :Sort by filename extension
-1 :Print output files one per line
--time=atime :Show last access timestamp for file
mkdir [*opts*] <*dirname*> ·**Make a new directory**
-p :Create parent directories if they don't exist
mv [-i] <*frompattern*> <*tofile*> ·**Move/Rename a file**
-i :Interactive move (Prompt before moving files)
rm [*opts*] <*filepattern*> ·**Remove a file**
-f :Force removal (Don't ask if it's ok to remove)
-i :Interactive remove (Ask before removing each file)
-r :Recusively delete directories an their contents
shred [*opts*] <*filepattern*> ·**Delete file data securely**
-n :Number of pattern iterations to run (default 25)
-u :Truncate and remove the file after overwriting
-z :Add a final overwrite with zeros to hide shreding
tar [*opts*] [*tarfile*] [*pattern*] ·**Create an archive**
c :Create mode. Create a tar archive
x :Extract mode. Untar archive contents
t :List mode. List the contents of the archive
f :Specify a tarfile to use
v :Verbose mode. Show files being added or untared
z,j :De/compress. Send i/o through gzip(z) or bzip2(j)
touch [*opts*] <*pattern*> ·**Update the timestamp on a file**
-t :Specify a timestamp to use instead of current time

Network

ifconfig [*devicename*] [*action*] [*options*]
ipchains [*opts*] ·**Manip. ipchains firewall(kernel 2.2+)**
iptables [*opts*] ·**Manip. iptables firewall(kernel 2.4+)**
-F: Flush current set of rules (Careful!)
-L: List the current rules
-n: Display rules without doing DNS lookups (faster)
mail [*opts*] [*address*] ·**Send mail from the command line**
-s *subject* :Specify the subject as *subject*
-c *list* :Send carbon copy to *list* of users
-b *list* :Send blind carbon copy to *list* of users
Ex: echo "Meet me at noon." | mail -s "Reminder" -c bob@company.com,suzy@company.com jack@company.com
netstat [*opts*] ·**Print network connections and info**
-a :Show both listening and non-listening sockets
-n :Do not attempt to resolve IP addresses
-t :Only show tcp socket connection table
ping [*opts*] [*host*] ·**Send ICMP packets to network hosts**
-c *count* :Send *count* number of packets and then quit
-i *sec* :Wait *sec* seconds between sending packets
route [*opts*] [*target*] ·**Show/Manipulate IP routing table**
-n :Show numerical addresses instead of hostnames
scp [*opts*] [[*host:*]*fromfile*] [[*host:*]*to*] · **Secure copy**
-C :Compresses the data that is sent over the session
-r :Recursively copy directories
ssh [*opts*] [[*user@*]*host*] [*command*] · **Secure shell/login**
-C :Compresses the data that is sent over the session
sniffit [*opts*] ·**Record TCP network traffic**
tcpdump [*opts*] [*expression*] ·**Dump traffic on a network**

Some examples of how to use tcpdump

tcpdump host foo (To or from host 'foo')
tcpdump not host foo (Not to or from host foo)
tcpdump port http (All to or from port 80)
tcpdump ip and not net localnet (non-local net.)

telnet [*opts*] [*host*] [*port*] ·**Open TCP socket to a host**
-n <file> : Opens *file* for recording trace information
-x :Turns on encryption of the data stream if possible
traceroute [*opts*] [*host*] ·**Show the route packets take**
-n :Don't do DNS lookups of the IP addresses
wget [*opts*] [*URL*] ·**Make a HTTP request from the shell**
-r :Recursive get the URL and all it's links
-k :Convert the non-relative links to relative ones
whois [*opts*] <*arg[@server]*> ·**Query a whois database**
Ex: whois domain.com
whois a.b.c.d (IPv4 address)

Informational

cat [*opts*] [*filepattern*] ·**Print file contents on STOUT**
-E :Display a $ at the end of each line
-T :Show tabs as ^I
-v :Show non-printing characters
date [*opts*] ·**Print or set the system date and time**
--date=*STRING* :display time described by *STRING*
--set=*STRING* :set time described by *STRING*
dmesg [*opts*] ·**Print or control the kernel ring buffer**
-c :Clear the contents of the ring buffer
file [*opts*] [*filepattern*] ·**Determine the file type**
-z :Try to look inside compressed files
finger [*opts*] [*userpattern*] ·**Show info about system users**
-m :Match the exact username specified
free [*opts*] ·**Display free and used memory in the system**
-b :Display the information in bytes
hexdump [*opts*] ·**Show all the characters of a file**
-c :Display the input offset in hexidecimal
last [*opts*] [*username*] ·**Show last system logins for users**
-*num* :Show last *num* of sessions
-a :Display the hostname in the last column
-d :Translates IP numbers to their hostname
-f <file> :Use file as last log
less [*opts*] [*filepattern*] ·**View a file a page at a time**
-i :Do case insensitive searching
-S :Don't wrap long lines
+[less commands] :Pass initial commands to less
lsof [*opts*] [*names*] ·**List all open files**

Try these useful tasks with lsof

When the CD-ROM is "busy":	**lsof /dev/cdrom**
Programs using audio:	**lsof /dev/dsp**
List open ipv4 network files:	**lsof -i 4 -a**

man [*opts*] [*section*] <*manpage*> ·**View software manual pages**
-a :View all available manual pages for name
Ex: 'man ls' or 'man -a nice' or 'man 5 crontab'
md5sum [*opts*] [*filepattern*] ·**Show the uniqueness of files**
-c :Check MD5 sums of files against md5sum listfile
ps [*opts*] ·**Show what processes are running on the system**
a :Select all processes on a terminal
u :Display user oriented format. More columns
x :Select processes without a controlling TTY
w :Show an extra line of process entry per w specified
Ex: ps auxwww =Displays all process information on system
quota [*opts*] [*user*] ·**Display disk usage and limits**
-v :Display filesystems where no quota is set
random **</*numpattern*/>** ·**Print out a random number from numpattern**
Ex: random /500..1000/ (print a random number between 500 and 1000)
slocate [*opts*] [*pattern*] ·**Locate pattern in file index db**
-i : Case insensitive search
-r : Search the database using POSIX regular expressions
time [*opts*] [*command*] ·**Show resource usage for a command**
top [*opts*] ·**Display top CPU processes every X seconds**
-d *sec* :Set the delay to *sec* seconds before refreshing
umask [*opts*] [*mode*] ·**Set the default file permissions**
-S :Show current symbolic umask
uname [*opts*] ·**Show OS and system information**
-a :Show everything
uptime ·**Show system uptime and load**
w [*opts*] [*user*] ·**Show who is logged in/what they are doing**
whereis [*command*] ·**Locate the related files for a command**
which [*command*] ·**Show full path to the specified command**
who [*opts*] [*args*] ·**Show who is logged in**

Bash Shell

> ·**Send STDOUT to a file. overwrite/create a file**
Ex: ls -l > list-of-files.txt
>> ·**Send STDOUT to a file, appending to te end of the file**
Ex: ps aux > pslog.txt
| ·**Send the STDOUT from a command to the STDIN of another**
Ex: cat listofnames | sort
2> ·**Send STDERR to a file, overwriting the filename**
Ex: startx 2> X-errorlog

Command pipelines in action

(records all running apache processes and kills last 10 in process table)

```
$ ps auxw | grep [a]pache | tee apache-allprocs.txt | grep ^apache |
   awk {'print $2'} | tail -n 10 | xargs kill
```

alias ·**Create a command alias in the shell**
Ex: alias ls='ls -la --color=auto'
cd [-] [*directory*] ·**Change the current working directory**
- :Change to the previous directory you were in
clear ·**Clear the terminal display** (also can use Ctrl-L)
env [*opts*] [*command*] ·**Run command in modified environment**
export [*opts*] [*variable*] ·**Export an environment variable**
Ex: export TERM=vt100
for ·**Execute sequence of commands for a list of items**
Ex: for i in *.mp3 ; do mpg123 $i ; done
history ·**Show the command history up til now**
nice [*opts*] [*command*] ·**Set the OS process priority**
Ex: nice 19 gzip access_log (lowest priority on Linux)
Ex: nice -20 kswapd (real time priority on Linux)
pwd ·**Print out the current working directory**
range **[*opts*] </*numpattern*/>** ·**Print a range of numbers for use in loop**
Ex: for i in `range /1..20/` ; do mkdir $i ; done
renice [*opts*] <*arg*> ·**Change priority of a running process**
-p <*PID*>: Specify a process id (<*PID*>) to "renice"
Ex: *bob 6319 ? S 0:20 gzip bigfile.txt* (output line from running ps auxw)
then run: 'renice 19 -p 6319' (which changes the priority)
reset ·**Initializes the terminal as if you just logged in**
set ·**Set a shell option or variable (run 'help set')**
sleep ·**Pause for specified period before continuing**
Ex: ps aux ; sleep 3600 ; ps aux
umask ·**Set the default file permissions**
Ex: umask 022 (files will be created 644 by default)
while ·**Loop that runs commands while a condition is true**
Ex: while (true) ; do ps auxw ; sleep 1m ; done > pslog
xargs [*opts*] [*command*] ·**Execute a command for each arg**
-n *number* :How many arguments to give each command run
-p :Prompt the user before each command is run

Text Filtering / Mutative

average **[*opts*] [*filearg*s]** ·**Print the average of all numbers encountered**
awk [*opts*] [*exp*] ·**pattern scanning and processing language**
-F<*fs*> :Set the field separator to <*fs*>
Ex: cat access_log | awk {'print $1'} (prints hostnames)
Do a 'man awk' for more information and examples
comm [*opts*] [*file1*] [*file2*] ·**Compare two sorted files**
-1 :Suppress lines unique to left file
-2 :Suppress lines unique to right file
-3 :Supress lines unique to both files
csplit [*opts*] [*file*] [*pattern*] ·**Split a file on context**
-f *prefix* :Use *prefix* instead of xx in output filenames
-n <*digits*> :Use <*digits*> number of digits instead of 2
-z :Remove empty output files
Ex: csplit mailspoolfile "/^From /" {*}
cut [*opts*] [*filepattern*] ·**Remove sections from each line**
-c *range* :Output only the characters in *range*
Ex: cut -c 1-80 *file* (truncate lines at 80 characters)
diff [*opts*] [*file1*] [*file2*] ·**Differentiate two files**
Ex: diff program-old.c program.c > program.patch
echo [*opts*] [*string*] ·**Print a line of text**
-e :Enable interpretation of backslashed sequences
-n :Don't automatically insert a newline character
fold [*opts*] [*files*] ·**Wrap each line to a specified width**
-s :Break at spaces instead of in the middle of a word.
-w <*WIDTH*> :Use <*WIDTH*> columns rather than 80
grep [*opts*] [*pattern*] [*file*] ·**Print lines matching pattern**
-B <*num*> :Print <*num*> lines of leading context on matches
-C <*num*> :Print <*num*> lines of trailing context on matches
-E :Interpret *pattern* as an extended regular expression
-i :Do case insensitve matching
-l :Just print the files that match the *pattern*
-r :Read all files under each directory recursively
-v :Print the lines that don't match *pattern*
head [*opts*] [*file*] ·**Print the first part of a file**
-n *num* :Print the first *num* lines instead of the first 10
numsum **[*opts*] [*filepattern*]** ·**Print the sum of a group of numbers**
Ex: cat numbers.txt | numsum (Add up all numbers in a file)
numgrep **</*numpattern*/> [*filepattern*]** ·**Print lines matching numpattern**
Ex: cat numbers.txt | numgrep /2..100/ (Print numbers from 2 to 100)
nl [*opts*] [*file*] ·**Number the lines of a file**
paste [*opts*] [*files*] ·**Merge lines of files horizontally**
patch [*opts*] [*patchfile*] ·**Patch a file using a diff file**
sed [*expression*] [*file*] ·**Stream editor**
Ex: cat file | sed 's/*frompattern*/*topattern*/' > output
sort [*opts*] [*file*] ·**Sort lines of text files**
-n :Compare according to string numerical value
-r :Reverse the result of comparisons
split [*opts*] [*file*] ·**Split a file into pieces**
-l <*num*> :Put <*num*> lines per output file
tail [*opts*] [*file*] ·**Print the last lines of a file**
-f :Output appended data as the file grows
-n <*num*> :Print last <*num*> lines of instead of the last 10
tee [*opts*] [*file*] ·**Send current output stream to file**
-a :Append to the given file instead of overwriting
tr [*opts*] <*set1*> [*set2*] ·**Translate char. from set1 to set2**
Ex: cat index.html | tr A-Z a-z > index-new.html
uniq [*opts*] [*input*] [*output*] ·**Remove duplicate lines**
-c :Prefix lines with number of occurances
-d :Only print duplicated lines
-u :Only print unique lines
-w <*n*> :Check no more than <*n*> characters in lines
wc [*opts*] [*file*] ·**Print the number of lines in files, etc.**
-m :Print the character count
-l :Print the line count
-w :Print the word count
-L :Print the length of the longest line

Admin

adduser [*opts*] <*username*> ·**Add a user to the local system**
-d <*dir*> : Set the home directory for the user to *dir*
-g <*group*> : Set the primary group for the user to *group*
-G <*group,group,…*> : Set additional groups for the user
-s <*shell*> : Set the default shell for the user to *shell*
crontab [*opts*] ·**Edit user crontab for periodic execution**
-e : Edit a crontab
-u <user> : Specify <user> for crontab operation
edquota [*opts*] <*user*> ·**Edit a user's or group's quota**
-g : Edit the group quota instead of user quota
fsck [opts] [filesystem] ·**Check and repair a filesystem**
-y : Answer yes to any questions. (Use with caution!)
kill [-*signal*] <*pid*> ·**Terminate a process/Send it a signal**
-HUP,-1 : Signal usually makes process to reread config
-9 :Send a SIGKILL, process must die
-l :Print a list of signal names and numbers
killall [-*signal*] [*name*] ·**Kill processes by name**
-e :Require an exact name of a process
-i :Interactively ask for confirmation before killing
Ex: killall -9 sendmail
ldd [*opts*] [*program*] ·**Show a programs library dependencies**
ldconfig ·**Configure dynamic linker run time bindings**
(run this program after changing /etc/ld.so.conf)
makewhatis ·**Create the whatis db for searching man pages**
mount [*opts*] <path/*device*> [*mountpoint*] ·**Mount a filesystem**
-o <*opts*> : Specify options for mounting. Listed below
loop - Mount a disk file such as a CD-ROM image or floppy image
remount - Remount the filesystem with new options
ro, rw - Mount filesystem in readonly or read-write mode
user - Allow normal users to mount this filesystem
-r :Mount the filesystem read-only. Same as '-o ro'
-t <*vfstype*> :Specify the type of filesystem to mount
ext2, ext3 - Native Linux partition types.
reiserfs - Advanced Linux filesystem
xfs, jfs - Other advanced Linux filesystems
vfat - Windows 9x 32-bit partition type
msdos - Old DOS/Windows partition type
iso9660 - CD-ROM filesystem
nfs - Network remote filesystem
passwd [*opts*] [*username*] ·**Change user's system password**
-l :Lock the password for the account
-u :Unlock the password for the account
-S :Show the status of the password for the account
su [-] [*username*] ·**Switch users or login as the superuser**
- :Make shell a login shell
-c <*command*> : Run <*command*> as *username*
umount [*opts*] [*path/device*] ·**Unmount a mounted filesystem**
-f :Force unmounting (in case of unreachable NFS system)
-l :Complete the unmount once filesystem is no longer busy