CA170: Week 3

Intro to Unix

Files

```
ls                      List files
ls -a                   Show "hidden" files (begin with ".")
ls -l                   Detailed
ls -alR                 Recursive

cat (file)              Type file out in command-line window

more (file)             Type file, pause for each screenful
                        enter for new line, space for next page, q to
quit
```

Manipulating Files

- Human can use GUI file manager. Programs can do anything a human can do, with these commands.
- Human sometimes types these commands too.

```
cp                      Copy files
mv                      Move / Rename files
rm                      Remove files
mkdir                   Make directory
rmdir                   Remove directory
```

- Notes have tutorials

Detach Program (&)

```
(prog) &                Launch a process detached
                         from command-line (e.g. windowed)
(prog)                  Command-line frozen until prog exits.


eog file.jpg &          Launch image viewer Eye of GNOME on file
```

Web Browser

```
firefox &               Launch Web browser from command-line
firefox "URL" &

google-chrome &

konqueror "URL" &
```

- Firefox multiple windows is confusing on Gnome.
- To find other windows: "Activities"
- Or Alt-Tab and pause on Firefox and they pop up.

sort

```
sort        Sort alphabetically (pipe some stream into sort)
sort -n           Sort numerically
sort -r           Reverse sort

sort by 5th column:
sort -k 5

old syntax to sort 5th column:
sort +4
```

grep

```
grep      Search for a string in a file or files

grep (string) (file)
grep (string) *html

grep -i (string) (file)  Ignore case
grep -v (string) (file)  Return lines that do NOT match
```

find

```
find      Find files by type/date/name, in this dir or below

find -type d       Find all dirs
find -mtime -1     Find files modified today
```

du

- Links to more notes in notes

```
du                       Space used by me

du | sort -n           # why -n?

default is k

listing in M
du -BM

listing in G
du -BG

top-level only
--max-depth=1
```

Printing

```
lp (file)               Print
lpr (file)              Print (on some systems)
lp -Pl128 (file)        Print on printer l128 (L128)

lpq                     See print queue
lprm                    Remove job from queue
```

cal

```
cal                     Calendar

cal 8 1752              Calendar for Aug 1752
cal 9 1752              Calendar for Sept 1752
cal 10 1752             Calendar for Oct 1752
```

which

```
which (prog)
 what runs if "prog" is typed
 may return nothing if prog is an alias

which ls


type (prog)
 returns path of prog
 or else shows what prog is alias for

type h
type history


whereis (prog)          Where the binary, source, manual pages are for this prog
whereis perl
```

Misc

```
df -h                    Show space on all disks
df -k                    exact kilobytes


w                        Who is logged in
            (see this when you ssh student.computing.dcu.ie)

(command) ; (command)    Multiple commands on same line
```

wget

- Command line HTTP client

```
wget -q -O - URL                                    Download URL, quiet, output
to command-line

wget -q -O - URL > file.htm                         Output to file

wget -q -O file.htm  URL                            Output to file

wget -q -O - http://site/file.jpg  > file.jpg             Output JPEG to file
(output JPEG to command-line won't work)

wget -q -O file.jpg  http://site/file.jpg           Output JPEG to file
```

- Link to more in notes

Absolute and Relative Paths

- Relative path of a file

```
index.html
```

  - What file that refers to depends on what directory you are in now.
  - It looks for index.html in the current directory.

```
../index.html
```

  - is also relative path. It looks for index.html in the parent of the current directory.

Directory before	Command	Directory after
/users/gdf	cd users/ec2	/users/gdf/users/ec2
/users/gdf	cd ../ec2	/users/ec2
/users/gdf	cd ec2	/users/gdf/ec2

- Absolute path of a file

```
/dir/dir/dir/public_html/index.html
```

  - Gives the full "path" from the root down to the file.
  - Refers to the same file no matter what directory you are in.

Directory before	Command	Directory after
/users/gdf	cd /users/ec2	/users/ec2

Case Sensitivity

- Case matters in filenames in UNIX (this is why case often matters on Web).
- Question: Is case sensitivity a good thing? Or is it a flaw in UNIX?
- Advantages of case sensitivity:
  - More readable code. You know what to expect.
  - More variables. num and NUM and Num
  - Set up conventions, so that NUM probably refers to a compile-time-coded constant, num is a real-time-changing variable, etc.
  - Quicker/simpler searches on strings and changes of strings, since can just search for the literal string.
  - Better to be case-sensitive for passwords. - Larger space to pick from. Harder to guess. Good to be "unforgiving" for security.
- Not much return for such huge disadvantages:
  - Millions of programmer and user hours lost on case not right.
  - Millions of failed "404 Not Found" website hits because of wrong case in the URL.
- Solutions to "404 Not Found" because of case:
  - Set up program to handle 404. See My "404 Not Found" Handler
  - Detach website URLs from underlying (case sensitive) file system.
    - e.g. Content Management System.

Filenames and Special Characters

- Long file names and multiple periods OK.
- e.g. product.4652.suppliers.us.html
- UNIX had long filenames from the start, unlike DOS/Windows with the 8.3 format (still part of legacy command-line interaction on Windows).
- openSUSE uses ext3 file system (and here)
- Comparison of file systems
- Limits
    - ext3 allows 255 char file names
    - Linux has a maximum path name limit of 4096 chars.
    - Part of the source code on my install:

      `/usr/include/linux/limits.h`
    - shows the following:
    - `#define NAME_MAX 255 /* # chars in a file name */`
    - `#define PATH_MAX 4096 /* # chars in a path name including nul */`
- Avoid these
    - If the command-line is used to address files, it is best to avoid many special characters in filenames.
    - Avoid these chars in filenames, because they have meaning to the Unix command-line and utilities:

```
space (separate arguments)
# comment
< redirection
> redirection
` result of a program
| pipe
& detach process
; separate multiple commands on the same line

* wildcard
? wildcard
^ start of line
$ end of line / variable value
[ pattern matching
] pattern matching
\ "quoted" character
/ should be in pathname, not filename
' string delimiter
" string delimiter
! shell history
```

    - On many UNIX/Linux distributions (e.g. openSUSE) you can actually put these chars in filenames. But the file may then be hard to work with at the command-line, and naive scripts may crash.
    - If you just point-and-click your UNIX files (which is allowed too) then you can use many of these characters.

- Stick to these
  - If you're going to use the command-line, best to just use these chars in filenames:

```
0-9
 a-z
 A-Z

Use these inside filename only, not at start or end:
 .
 -
 _
```

  - Filename (notes on wiki)
  - File name characters (notes in wiki)
  - UNIX / Win are quite restrictive - lot of bad chars.
  - Mac can use almost any char.
  - Though could then get problems on Mac command line.
  - Can get problem on UNIX when you download or copy in files from another OS with odd chars in file names.
  - Filenames with spaces are creeping into the UNIX world, but old scripts may fail with them.
  - Find any file or directory below current dir with spaces in its pathname:

```
$ find . | grep ' '
```

Notes on File Protection

File Protection

- "ls -l" shows something like:

```
-rwxr-xr-- 1 userid  groupid  153 Nov  6 2008 filename

-     file (d for directory, l for link/shortcut)
rwx   User (u) can read,write,execute.
r-x   Other members of group (g) can read,execute only.
r--   Other people (o) can read only.


set via the "chmod" command.
see "man chmod"


    user      group      other
 [ ][ ][ ]  [ ][ ][ ]  [ ][ ][ ]

r - read
w - write
x - execute
```

- e.g. user can do everything, group/others can do nothing:

```
chmod u+rwx,go-rwx file
```

- result:

```
-rwx------ 1 userid groupid 153 Nov 6 2008 filename
```

- There is also a number that corresponds to each permission setting:
  - chmod converter (and search for more) - more in notes
- Default permissions for new files: umask

User bits

- Note if turned off, user has power to turn them on any time.
- So these can only be for some kind of temporary self-check.

[r][w][-]	Don't execute by accident. Because UNIX will try to execute any text file as shell script if name is typed. e.g. text files, web pages
[r][-][x]	write-protect for safety annoying?
[r][-][-]	both of above
[r][w][x]	normal

group/other bits

[r][w][x] [r][w][-]	Shared writable file
[r][-][x] [r][-][-]	Shared read-only file
[-][-][-]	Normal - Hidden from others

Minimum Needed for Web Files

- (Web server is "other".)

```
Web pages need r:
 -rwx---r--
PHP scripts only need r, not x:
 -rwx---r--
```

Notes on Directory Protections

```
    user       group       other
 [ ][ ][ ]   [ ][ ][ ]   [ ][ ][ ]

r - read (can do ls)
w - write
x - search (can access files given their name)
```

User bits

- Note if turned off, user has power to turn them on any time.

[r][-][x]	write-protect for safety annoying?
[r][w][x]	normal

group/other bits

[r][w][x]	shared writable directory can create/delete files
[r][-][x]	shared read-only directory can do ls
[-][-][x]	shared read-only dir can't do ls can access file if know its name can't explore without filenames Example: "share" in my home dir. You just need to know this dir exists. Example: web dir Can only browse named files. The names are *in the links*. The site advertises a starting point (a web page from which all other web pages can be found by following links).
[-][-][-]	normal - hidden

-

Raw listing of files on web servers

- It used to be that we could demo the difference between r and x for web directories.
- In my web dir:

```
drwx---r-x    readabledir
drwx-----x    executabledir
```

- readabledir/file.html - link in notes
- executabledir/file.html - link in notes
- executabledir - index.html does not exist, so it just returns error. - link in notes
- readabledir - index.html does not exist, but dir is readable, so what it used to do is return a raw listing of files. - link in notes

Raw listing of files is now turned off

- The above (raw listing of files) does not work any more on student.computing.dcu.ie.
- On Apache, the behaviour of listing directory contents or not can be controlled with `Options +Indexes` (or `Options -Indexes`) in .htaccess files.
- It is now turned off.

Minimum needed for web directories

- (Web server is "other".)

```
drwx-----x
```