

# CST8177 – Linux II

Review of Fundamentals (cont'd)

Todd Kelley  
kelleyt@algonquincollege.com

# Topics

- ▶ change your password on CLS if you haven't already
- ▶ the filesystem
- ▶ access permissions
- ▶ symbolic links
- ▶ hard links

# Variables

- ▶ Variables for general use (variables that are not environment variables) have lower case names
- ▶ Environment variables are indicated by their UPPER CASE names: SHELL, VISUAL, etc
- ▶ It's usually best to put variable expansions inside double quotes, to protect any special characters that might be inside the variable:

echo "\$somevar"

- if somevar contained the \* character, the double quotes stop the shell from globbing it

# Setting Variables

- ▶ set the variable `myvar` to have value `value`

```
myvar=value
```

- ▶ Note, to make this variable setting visible in sub processes we use `export`

```
export myvar=value
```

**or**

```
myvar=value
```

```
export myvar
```

# Variable setting for command

- set the `myvar` variable to have a null value, then run the `value` command with that variable setting in effect

```
myvar= command
```

- ▶ Notice that if you try mistakenly use this to try to set the value of `myvar` to `value`

```
myvar= value
```

in this case you are actually trying to run a command called `value`

# Variable setting for command (cont'd)

The usual way to use this mechanism is something like

```
VISUAL=nano vipd
```

- ▶ This means to set the value of the environment `VISUAL` variable to `nano`, and use that while the `vipd` command runs

# Setting Variables Mistakes

- set the `myvar` variable to have a null value, then run the `value` command with that variable setting in effect

```
myvar= value
```

- run the `myvar` command with one argument, namely `=value`

```
myvar =value
```

- ▶ run the `myvar` command with two arguments, namely `=` and `value`

```
myvar = value
```

# File Permissions

```
kgk — kelleyt@idallen-ubuntu: ~ — ssh — 80x24
kelleyt@idallen-ubuntu:~$ ls -ail
total 64
399303 drwxr-xr-x  4 kelleyt kelleyt  4096 Jan 13 11:44 .
131074 drwxr-xr-x 242 root     root     4096 Jan 10 16:18 ..
501292 drwx-----  3 kelleyt kelleyt  4096 Jan  7 11:37 Assignments
400793 -rw-----  1 kelleyt kelleyt  1810 Jan 13 12:06 .bash_history
399476 -rw-r--r--  1 kelleyt kelleyt   220 Jan  2 21:22 .bash_logout
503350 -rw-r--r--  1 kelleyt kelleyt  3625 Jan  7 11:34 .bashrc
500319 drwx-----  3 kelleyt kelleyt  4096 Jan 11 14:35 .cache
399306 -rw-r--r--  1 kelleyt kelleyt  8445 Jan  2 21:22 examples.desktop
503413 -rw-----  1 kelleyt kelleyt    51 Jan 11 15:53 .lesshst
397428 -rw-----  1 kelleyt kelleyt     7 Jan 11 15:42 .nano_history
394987 lrwxrwxrwx  1 kelleyt kelleyt    53 Jan  6 08:29 notes -> /home/idallen/
public_html/teaching/cst8207/12f/notes/
399921 -rw-r--r--  1 kelleyt kelleyt   675 Jan  2 21:22 .profile
397797 -rw-----  1 kelleyt kelleyt 10080 Jan 13 11:44 .viminfo
kelleyt@idallen-ubuntu:~$ pwd
/home/kelleyt
kelleyt@idallen-ubuntu:~$ █
```



# Typical directory and file

inode 399303  
drwxr-xr-x  
access time  
modification time  
change time  
...etc...

.	inode 399303
..	inode 131074
examples.desktop	inode 399306
Assignments	inode 501292
...etc...	...etc...

inode 399306  
-rw-r--r--  
access time  
modification time  
change time  
...etc...

data blocks for  
the file  
there is no  
filename here  
the filename(s)  
(at least one) are  
stored in  
directories

# File Permissions (cont'd)

inode 399303

drwxr-xr-x

access time

modification time

change time

...etc...

Need read (r) on directory to read this column

.	inode 399303
..	inode 131074
examples.desktop	inode 399306
Assignments	inode 501292
...etc...	...etc...

Need search (x) on directory to access this column

Need write (w) *and* search (x) on directory to change first column

# File Permissions (cont'd)

inode 399306  
-rw-r--r--  
access time  
modification time  
change time  
...etc...

Need search (x) on  
**directory this file is in**  
to access this info on  
the file's inode

data blocks for  
the file  
there is no  
filename here  
the filename(s)  
(at least one) are  
stored in  
directories

Need read (r) / write  
(w) / execute (x) on **file**  
to read / write / execute  
this file (contents)

# File Attributes

Field No.	Stat Name	Unix	Win98/NT	MacOS
1	st_dev	Device number of filesystem	Drive number	vRefNum
2	st_ino	Inode number	Always 0	fileID/dirID
3	st_mode	File mode	File mode	777 dirs/apps; 666 docs; 444 locked docs
4	st_nlink	Number of links to the file	Number of link (only on NTFS)	Always 1
5	st_uid	Owner ID	Always 0	Always 0
6	st_gid	Group ID	Always 0	Always 0
7	st_rdev	Device ID for special files	Drive No.	Always 0
8	st_size	File size in bytes	File size in bytes	Data fork file size in bytes
9	st_blksize	Preferred block size	Always 0	Preferred block size
10	st_blocks	Number of blocks allocated	Always 0	Number of blocks allocated
11	st_atime	Last access time since epoch	Last access time since epoch	Last access time -66 years
12	st_mtime	Last modify time since epoch	Last modify time since epoch	Last access time -66 years
13	st_ctime	Inode change time since epoch	File create time since epoch	File create time -66 years

# Extending Unix

- ▶ create a command with basic scripting
  - put “#!/bin/sh -u” at very beginning of file
  - put commands in file
  - make file executable
- ▶ put the file in a directory that is in \$PATH
- ▶ [http://teaching.idallen.ca/cst8207/13f/notes/400\\_search\\_path.html](http://teaching.idallen.ca/cst8207/13f/notes/400_search_path.html)
- ▶ Not a good idea to put “.” in PATH
- ▶ Security implications of putting “current directory” , “.” in PATH
- ▶ `PATH= . : $PATH`
- ▶ demonstration of how the bad guy can arrange for you to inadvertently run their malicious commands as you