Marc Coma races his KTM in the 10th stage of the 2012 Argentina-Chile-Peru Dakar Rally between Iquique and Arica, Chile on January 11, 2012. (Jerome Prevost/ Associatied Press/Pool)

Viewing and Manipulating Files and Directories

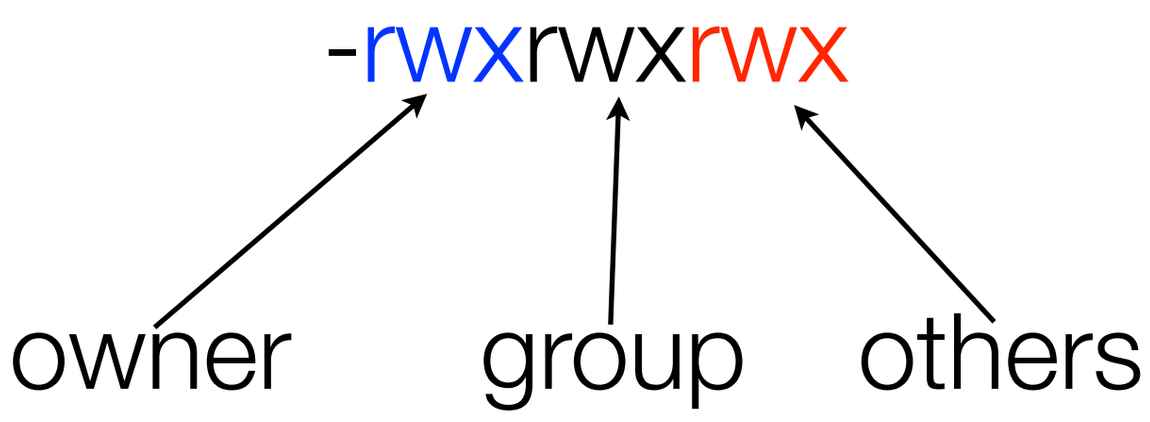
Lecture #3 - COMP 364
January 8, 2010, updated 2012
Derek Ruths

Last class

- More commands:
    - Print the current directory? Getting a manual page for a command?
- Files and directories
    - Difference between an absolute path and a relative path?
- Today: finish permissions, files and directories
- Next class: Regular Expressions

Permissions (on UNIX)

- The three main actions a user may perform on a file/directory: *read (r), write/modify (w), execute (x)*
- The file system enforces permissions on every file and directory: permissions indicate whether a user may perform each of these actions
- A separate rule exists for the owner of the object (u), the group owning the object (g), and everybody else (o).
- *Can only change permissions if you are the owner of the file!*



chmod: changing permissions

- *chmod <a/u/g/o><+/-><r/w/x> <file/directory name>*

- Adding a permission:
  - *chmod u+w foo.txt*
  - *chmod u+wx bar*
- Removing a permission:
  - *chmod o-r foo.txt*
  - *chmod o-rwx bar*

Viewing file contents

- cat <filename> - print all content out
- head <filename> - print out the first 10 lines
    - head -n # <filename> - print out the first # lines
- tail <filename> - print out the last 10 lines
    - tail -n # <filename> - print out the last # lines
- more <filename> - a better version of *cat*
- less <filename> - a better version of *more*

Manipulating files and directories

- **4 main activities:**
    - Creating
    - Deleting
    - Moving
    - Copying

Creating empty files and directories

- Creating an empty file:
    - *touch <filename>*
- Creating an empty directory:
    - *mkdir <dirname>*
        - *mkdir hs_genes*
        - *mkdir ~/project1*
        - *mkdir -p ~/project1/doc/final*

rm: deleting a file

- *rm <file to remove>*
  - *rm manuscript.txt*
  - *rm ~/test.py*
  - *rm test**

* and ?: the wildcards

- Wildcards are powerful ways of referring to several objects at once without having to name each one.
- * = zero or more of any characters
- ? = exactly one character (it can be any character)
- Examples:
  - *test?.py*: **test1.py, test2.py, test3.py, test4.py,** test40.py, test400.py
  - *test4*.py*: test1.py, test2.py, test3.py, **test4.py, test40.py, test400.py**
  - *foo**: **foo, foo.txt, foobar,** fo1, fo2, **foobar.txt**

More wildcard examples

- What string (containing at least one wildcard) would you use for the bolded characters in each set?
  - **hs001, hs002, hs003, hs004, hs009** ec001
  - **hs001, hs002, hs003, hs004, hs009,** hs010, hs011
  - **hs01.txt**, hs01.csv, **hs10.txt**, hs10.csv
  - **hs1g1.fasta hs2g1.fasta hs3g20.fasta hs4g2.fasta.old** h10g3.fasta

Wildcard strings are expanded

- In a directory that contains:
    - foo.txt, foo.csv, bar.1, bar.2
- *ls -l foo* ➠ ls -l foo.txt foo.csv*
- *rm foo* ➠ rm foo.txt foo.csv*
- What does this command do? *rm **
- What does this command do? *chmod a+r bar.?*

Deleting a directory

- rmdir: (“Remove Directory”) remove an empty directory
    - rmdir <directory name>
- rm -r: (“recursive delete”) remove a directory and everything in it (BE CAREFUL!)
    - rm -r <directory name>

mv: moving files and directories

- *mv <source> <destination>*
  - *mv foo1 foo1.old* (renaming)
  - *mv foo1 ..* (moving)
  - What will this do? *mv foo* ..*

cp: copying file and directories

- cp <source> <destination>
  - *cp foo.txt ..*
  - *cp foo.txt ~/backup/foo.txt = cp foo.txt ~/backup*
- cp -r: recursive copy
  - cp -r ~/projects /usr/backup

ln: creating symbolic links

- Like the “shortcut” feature in Windows and the “alias” feature in Mac

- Create a “link” object that refers to another file/directory (called the source). Most actions performed on this object are performed on the source.

- ln -s <source> - creates a link in the current directory that is identically named to the source

Pipelines

- The | operator will take the output of a command and send it to another command
    - curl http://en.wikipedia.org/wiki/Pipeline_(Unix) | head -n 10
    - cat /usr/share/dict/words | less
    - cat *.fasta | grep AAA
    - tail and head can be combined together with a pipeline!