Wrangling Data in Linux

Peter Ruprecht
peter.ruprecht@colorado.edu

www.rc.colorado.edu

Slides:
https://github.com/ResearchComputing/RMACC/tree/master/2017/

Outline

- Quick Linux review
- Filesystem layout
- Pattern matching (regular expressions)
- Finding text in files
- Stream editing and column operations
- Sorting
- Finding files in the filesystem
- *How full is a disk?*
- *Permissions*
- *Links*

Pipes and redirection

- Input and output redirection
  - Send output from a command to a new file with >
  - Append output to an existing file with >>
  - Use a file as input to a command with <

- Pipes: | sends output of one command to another command

```
ps -ef | grep ruprech
```

File- and directory-related commands

pwd – prints full path to current directory
cd – changes directory; can use full or relative path as target
mkdir – creates a subdirectory in the current directory
rmdir – removes an empty directory
rm – removes a file (`rm -r` removes a directory and all of its contents)
cp – copies a file
mv – moves (or renames) a file
ls – lists the contents of a directory (`ls -l` gives detailed listing)
chmod/chown – change permissions or ownership
df – displays filesystems and their sizes
du – shows disk usage (`du -sk` shows size of a directory and all of its contents in KB)
tar – agglomerates multiple files into a single file (like “zip”)
gzip/gunzip – compresses or uncompresses files

File-viewing commands

less – displays a file one screen at a time

cat – prints entire file to the screen

head – prints the first few lines of a file

tail – prints the last few lines of a file (with –f shows in real time the end of a file that may be changing)

diff – shows differences between two files

grep – prints lines containing a string or other regular expression

tee – prints the output of a command and also copies the output to a file

sort – sorts lines in a file

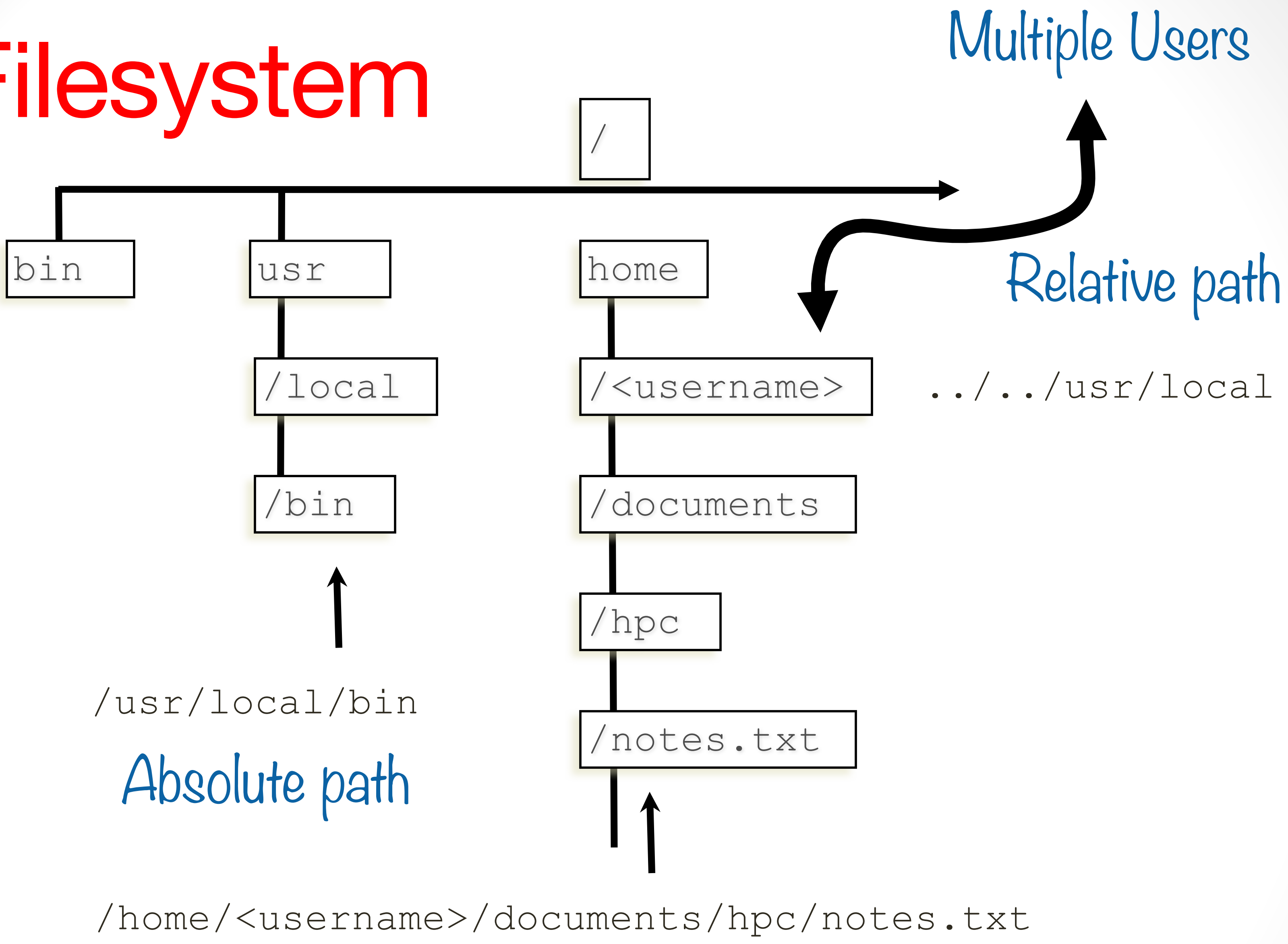
find – searches for files that meet specified criteria

wc – count words, lines, or characters in a file

Some shorthand

`.`	(the current directory)
`..`	(the directory one level above)
`~`	(home directory)
`-`	(previous directory, when used with `cd`)

Filesystem


Multiple Users
/
bin
usr
home
/local
/bin
/<username>
/documents
/hpc
/notes.txt
Relative path
../../usr/local
/usr/local/bin
Absolute path
/home/<username>/documents/hpc/notes.txt

Shell Wildcards and Special Characters

- `*` - matches zero or more characters
- `?` - matches a single character

- # - comment; rest of the line is ignored
- \ - escape; don’t interpret the next character

Regular expressions

`string`	match string exactly
`.`	Match single character `'19.3'` (matches 1903, 1913, 19A3)
`*`	Match zero or more of preceding character `'bugs*'` (matches bug, bugs, bugsss)

Regular expressions, contd

^	Match beginning of line ‘^data’ (line starts with data)
$	Match end of line ‘^…$’ (line with exactly 3 chars)
[ ]	Match from set ‘Jun[0-9]*_201[01]’ (Jun followed by any number of integers followed by _2010 or _2011)

Stream editing (with sed)

```
sed ‘s/Kr/krypton/g’ < input.txt > output.txt
```
(global find-and-replace of Kr with krypton)

```
cat input.txt | sed ‘/^$/d’ > output.txt
```
(remove all completely empty lines)

```
cat input.txt | sed ‘/^[[:space:]]*$/d’ >
      output.txt
```
(remove all lines containing only white space)

```
sed -e ‘s/^/   /’ input.txt > output.txt
```
(add 3 spaces to beginning of each line)

Column operations (with `awk`)

```
awk '{print $3}'
```

(print 3rd field or column)

```
awk -F: '{print $1,$3}'
```

(print 1st and 3rd fields; fields delimited by :)

```
awk '{print $NF}'
```

(print last column; NF means number of fields)

```
awk '{print NF}'
```

(print number of fields)

More with awk

```
awk '{total = total + $1}END{print total}'
```

(sums the first column)

```
grep '^[0-9]' data.txt | \
   awk '{print $2, 3.14*$1}'
```

(for lines beginning with a number, print the 2nd column followed by the 1st column times pi)

Sorting

```
sort file.txt
```

(sort file in ASCII order)

```
sort -n -r file.txt
```

(sort file in numerical order and print in reverse order)

```
sort -n -k 3
```

(sort file in numerical order by 3rd field)

```
sort -n -t, -k 3
```

(as above but fields are comma-separated)

Finding files (with `find`)

```
find /somedir -name "*.pdf"
```

(find files ending in .pdf in /somedir (and subdirs))

```
find ~ -mtime +3
```

(find files in homedir modified over 3 days ago)

```
find . -name "*.csv" -a -mtime -3
```

(find .csv files modified less than 3 days ago)

```
find . -perm 644 -exec chmod g+w {} \;
```

(find files with `rw-r--r--` ; change to `rw-rw-r--`)

File editing

- **nano** – simple and intuitive to get started with; not powerful; keyboard driven
- **vi/vim** – universal; keyboard driven; powerful but some learning curve required
- **emacs** – keyboard or GUI versions; helpful extensions for programmers; well-documented
- **OpenOffice / LibreOffice** – for WYSIWYG

http://xkcd.com/378/

Data transfer

- Globus Online
  - Large file transfers with “drag and drop” interface to move data between Globus or Gridftp endpoints

- Utilities
  - scp, sftp, rsync
  - Work best with smaller files or smaller numbers of files
- GUIs
  - putty, cyberduck, fugu, etc

Links

- Hard
  - Another name for an existing file
  - Adds additional name to file inode
  - Cannot cross filesystems
  - `ln original_file link_name`
- Symbolic
  - A special kind of file that is a pointer (“shortcut”) to another file or directory
  - Can cross filesystems
  - `ln -s target_name link_name`
  - `ln -s /scratch/summit/ruprech scratch`

How full is a disk?

- `df` displays filesystem information
  - Check if your disk is filling
  - Find where a filesystem is physically located
  - The "-h" flag gives "human readable" units

- `du` shows disk usage
  - `du -sk * | sort -n` is useful for finding large directories

Modes (aka permissions)

- 3 classes of users:
  - User (u) *aka "owner"*
  - Group (g)
  - Other (o)
- 3 types of permissions:
  - Read (r)
  - Write (w)
  - Execute (x)

Modes (continued)

- `chmod` changes modes:

To add write and execute permission for your group:

```
chmod g+wx filename
```

To set only read and execute for your group and others:

```
chmod go=rx filename
```

Thank you!

Slides and materials available at:

https://github.com/ResearchComputing/RMACC/tree/master/2017/