New2Linux:
New to the Command Line

June 3, 2021

John Nash
for
Virtual Meeting of Linux-Ottawa

Motivations

- Unix systems traditionally were “terminal” based
- Now plenty of GUI options, but the Terminal or Command Line still EXTREMELY useful
- Often more efficient
- Lets us create small scripts to handle repetitive tasks, or to customize tools for **ourselves**
- Initially may seem daunting. Rapidly gets easier with a bit of “try it and see”

Recent reminder of this

- Google is funding Arkajyoti Bhattacharjee for a project to improve nonlinear least squares function of the R base package
- AB uses Windows 10, where building R from source is “challenging”, but in Linux it is just

  ./configure; make; sudo make install
- Use VirtualBox VM of Linux Mint, with shared directory. Build accomplished easily, but AB needed to learn some command line skills.

“Open a Terminal”

- Many different programs are terminal emulators – they mimic 1970s screen and keyboard devices that were connected to local or remote computers.
  - Xterm, Konsole, mate-terminal, gnome-terminal, etc. -- tmux (for pro users)
- Once opened, there is a ***prompt.*** The prompt can generally be customised.

  john@M21:~/current$
- User can type commands that are executed by the ***shell***. This is a program, of which there are many
  - Examples: bash = Bourne-Again shell (often default shell),
  - Also sh (original and still available for compatibility), csh, tcsh, ksh
    - All slightly different in some of the more advanced details

“Open a terminal” 2

- In Linux Mint, “Terminal” (mate-terminal) is in System Tools portion of main menu.
  - Right-Click / Add to panel -- **recommended**
- Keyboard shortcut: Ctrl-Alt-T often works;
  - Superkey+T in Bunsenlabs Linux
- To close terminal, issue “exit” command
  - Or click close button on the terminal window

Why use Terminal?

- Particularly useful when we need access to administrative controls
  - Disk setup (partitioning), formatting, cleaning/checking
  - Copying some system files which are not “owned” by us
  - Changing permissions and ownership of files
- For reasons in the motivations
  - Scripts
  - Sometimes quicker/more flexible than using GUI

Some common commands

Learning about commands: man, info e.g., man rsync

- But Google may be more useful for beginners. Use man for syntax etc.

Copy, move, rename: cp, mv (mv can rename, but program “rename” can be installed and I find it easier to use; WARNING: regex used.

List files: ls (options) path

- Option “-al” gives lots of information; lots of others.

Update files: rsync (missing from Windows)

User access: adduser, passwd

Control / Information on processes: ps, kill, top, free, df, du

Make / Change directory: mkdir cd

Print a (text) file: cat (in fact can con**cat**enate text files)

```
cat a.txt b.txt c.txt  > x.txt # x.txt combines the three others
```

Power and Danger: sudo

- Runs command as admin (super-user / root)

  sudo chown john:john afile.type

    changes owner:group of afile.type to john

  sudo chmod o+x another.t2

    changes owner permissions on another.t2 to allow execution (part of Linux security model)

  sudo apt install packagename # to install programs

- Become “root” with user or root path

  sudo su sudo -i su

  - NOT a good idea generally, but sometimes needed

Running special programs

- If program is on the PATH, then type its name

  echo $PATH # to learn what the PATH is set to.

- If program is in current directory

  ./nameprogram

  Note that ./ points to current directory, ../ is 1 up

  - NOT familiar to Win-users! And use / not \

Conveniences

- Command completion: Using the Tab key will try to complete partially typed command or filename
    - REALLY worth trying and playing with
- The ~ symbol represents the users “home” directory.
    - I am user “john”, so cd ~

      will change directory to */home/john*
- Commands can be **piped** (daisy-chained so output of one feeds the next) with |
- Files can be read or written with < and >
- Pipe to “less” so output can be reviewed; exit with ‘q’

Annoyances

- Copy and Paste in Linux ***generally*** allows the keyboard shortcuts Ctrl-C and Ctrl-V (!! Other conventions.)
- BUT Terminal ***generally*** allows Ctrl-Shift-C and Ctrl-Shift-V
  - So you highlight material in a text editor or browser window and copy with Ctrl-C, but paste into the Terminal with Ctrl-Shift-V and vice-versa. This lets you move strings from and to a terminal window.
- Spaces and some characters “-”, “<”, “>”, “|” have meaning for some commands. CAUTION advised with spaces and special characters in filenames.
  - Surround filenames with quotes if there are spaces

Capture output - tee

- The command “ls Adirectory” will list files in the directory
  - Capture as list1.txt with “ls Adirectory >list1.txt”
  - Or “ls Adirectory | tee list2.txt”
  - Both files go in the **current** directory in focus of the terminal. Example:

    ./NSVD <hilbk2c1.in 2>&1 | tee ./NSVDhilbk2c1.out

    This runs program NSVD using data in hilpk2c1.in and copies errors to the standard output and puts all this output to the file NSVDhilbk2c1.out.

Some reference tutorials

https://linuxstudio.org/how-to-learn-and-master-the-linux-terminal/#What_is_Linux_Terminal

https://www.digitalocean.com/community/tutorials/an-introduction-to-the-linux-terminal

https://www.howtogeek.com/140679/beginner-geek-how-to-start-using-the-linux-terminal/

https://www.geeksforgeeks.org/tee-command-linux-example/

https://linuxize.com/post/linux-tee-command/