Author: Xingpeng Li, Assistant Professor, University of Houston (UH). Link: https://rpglab.github.io/resources
Log: Initially documented by Xingpeng Li at Arizona State University, 2014 fall.
Updated on 08/23/2022: mainly removed some materials unrelated to the class ECE3340 Numerical Methods at UH, Fall 2022.
Updated on 03/22/2023: added software demo information.

Linux Commands

Documented by Xingpeng Li

Linux Operation Commands

// Note that some commands are for Fortran, and some commands are possibly outdated.

1. **Some basic commands:**

cp route/file1 route copy
mv route/file1 route move (rename)
mkdir folderName create a new folder
touch filename create a file
rm filename delete a file
rm –r folderName delete a folder
zip squash.zip file1 file2 file3
unzip filename.zip unzip a compressed file *.zip
tar -xvzf filename.tar.gz unzip a compressed file *.tar.gz
tar -zcvf archive.tar.gz directory/ "zip" a directory
tar –c –f filename.tar folder1 folder2 compressed folder1 folder2 as a *.tar file
tar –c –f filename.tar file1 file2 compressed file1 file2 as a *.tar file
qsub -I log in one node
qsub job.pbs send batch file to the cluster, queue for execution3
pwd show the path to current folder
du show all subfolder and files, provide statistic numbers
ls show the files and folders in the current folder. -l –t
cd .. go to the upper folder
cd go to the default foler. e.g. /home/xingpeng
. denotes current folder
clear clear the screen
who show who have recently logged on (however, graphical UI does not count).
who show user and IP address
last show who have ever logged on.
Last username ---- show the info that this user ever logged on
date - check the current date and time
id
exit - exit the current group if you are in a group
exit - exit the terminal
echo print something on the screen (e.g.1, echo xpli - you'll see "xpli" on the screen;
e.g.2, echo $PATH - you'll see the environment variable "PATH").
cat print content of a file to the command window.
env show all environment variables

logout log out and exit the terminal
ctrl + C --- stop current program/commands and come back to the command line ($ symbol).
find /path/ -type d –print - list all subdirectories recursively
find . -type d –print - list all subdirectories in current directory recursively
find /path/ -type d -ls - list all subdirectories recursively
ls -R - show all files/directory including subdirectory
find . -print - show all files/dir including subdirectory (diff format with that from "ls –R")
make - run program (need a file called 'makefile' ready)
make clean - in general, this command would remove *.o *.out *.mod. But it depends.
make -n show what make will execute but do not actually execute

2. **high-level basic commands:**

cp -R ~/files ~/files-backup - copy a directory with all subdirectories and files.
ls *.jpg list the names of all files with names ending in ".jpg"
ls a* list the names of all files with names beginning with "a"
ls *att* list the names of all files with names containing "att"
ls -a lists all files, including the ones whose filenames begin in a dot, which you do not always want to see.
rm –i filename confirmation would be asked before actually deleting anything
diff filename1 filename2 --- compares files, and shows where they differ
wc filename --- tells you how many lines, words, and characters there are in a file
cd ../adir/bdir goes up one level then over to the directory, "adir", and the sub-directory, "bdir".
cp -R Documents "Documents backup" - copy an entire directory named Documents"; name the copy "Documents backup". The quotes are needed because of the space in the directory name.
gzip filename --- compresses files
gunzip filename --- uncompresses files compressed by gzip
gzcat filename --- lets you look at a gzipped file without actually having to gunzip it (same as gunzip -c).
grep telnet /etc/inetd.config search the inetd.conf file, and print all lines that contain "telnet".
w --- tells you who's logged in, and what they're doing. Especially useful: the 'idle' part. This allows you to see whether they're actually sitting there typing away at their keyboards right at the moment.
who - tells you who's logged on, and where they're coming from.
finger username --- gives you lots of information about that user.
talk username --- lets you have a (typed) conversation with another user.
write username --- lets you exchange one-line messages with another user.
whoami --- returns your username

passwd --- lets you change your password
chmod options filename lets you change the read, write, and execute permissions on your files. E.g., chmod o+r filename will make the file readable for everyone, and chmod o-r filename will make it unreadable for others again.
date --- shows the current date and time.
cal --- shows a calendar of the current month
ln make links between files
ln f1 f2 create hard-link file f2which links file f1
ln -s f1 f3 create symbolic-link file f3 which links file f1
(if f1 is removed, f3 doesn't exist any more, however, f2 is not affected. "cat f3" would show "No such file or directory")
emacs filename --- is an editor that lets you create and edit a file
pico - edit the contents of a text file. As with more and less, this only really works with plain text files.
tail -1000 /var/log/system.log | more print the last 1000 entries from the main system log, using more to display them one screenful at a time.

3. System operation

top - List the top CPU-consuming processes running on the system, along with various other system load statistics. Note: it runs continuously, updating the stats repeatedly, until you quit it with "q".
top -us5 Display processes sorted by CPU useage, updating every 5 seconds
ps List processes belonging to the current user that are attached to a terminal
ps –x List processes belonging to the current user whether or not they're attached to a terminal
ps -ax List all running processes
ps -aux List all running processes, with additional information about their resource useage
kill - Kill (or send other signals to) a process
ifconfig - Configure network interfaces (e.g. ethernet ports, AirPort cards, etc).

4. On MAC
   No insert key --- just type key i, like type enter insert key on Windows, then, "vi - mode".

5. How to search a file
   find . -name '*foo*' Search the current directory and all subdirectories for files whose names contain "foo".
   find -x / -name foo Search just the boot volume (the -x prevents it from crossing to other volumes) for files named exactly "foo", and print their paths.
   find / -name foo Search the entire file structure (including all mounted volumes) for files named exactly "foo", and print their paths. Note: under some conditions (especially involving

complex NetInfo networks) this may search infinitely amongst various network volumes. Use the -x option to avoid this (see next example).
find . -mtime -2 Search the current directory and all subdirectories for files modified within the last 2 days. (Note: this may also find some files with confused modification dates -- i.e. files that're marked as having been modified in the future. -mtime -0 would find ONLY the files with future modification dates.)

6. **Output info into a file.**
   AProgramExeCommand > tmp.txt all the info that is supposed to be printed on the screen now would be outputted to a file named "tmp.txt", if tmp.txt existed before, all the contents it has would be first deleted before any data move in.
   AProgramExeCommand >> tmp.txt all the info that is supposed to be printed on the screen now would be outputted to a file named "tmp.txt", all new data would append if tmp.txt existe before.

7. **How to check cpu info?**
   less /proc/cpuinfo - show cpu info
   cat /proc/cpuinfo - show cpu info
   grep "physical id" /proc/cpuinfo | sort -u | wc -l # Count the number of "physical processor(s)"
   grep "cpu cores" /proc/cpuinfo |sort -u |cut -d":" -f2 # Count the number of "physical cores per CPU"
   grep -c "processor" /proc/cpuinfo # Count the number of "logical cores " (including multi-threading cores)
   cat /proc/cpuinfo | egrep "core id|physical id" | tr -d "\n" | sed s/physical/\\nphysical/g | grep -v ^$ | sort | uniq | wc -l # the number of physical cores on a system
   cat /proc/cpuinfo | grep "physical id" | sort | uniq | wc -l # number of physical CPU sockets

8. **How to check Linux version you have?**
   uname -or
   uname -a
   lsb_release -irc
   cat /etc/lsb-release or cat /etc/debian_version or cat /etc/fedora-release or cat /etc/gentoo-release or cat /etc/*{release,version}

9. **Sudo commands:**
   sudo ls - show you all files and folders contained in your current folder.
   sudo rm file - delete file

10. **vi commands:**
    vi file open a file through vi
    vim file open a file through vim
    :q exit vi

:wq exit vi and save any change made to file
:q! exit vi and ignore any change made to file
:w save changes made to file
:d delete current line
i in vi mode, I is insert on iMAC
shift + g go to the end of the file
Press g twice to go to the beginning of the file
:22 go to 22th line.
22G Move to nth line of the file
G Move to the last line of the file
0 Move to the begining of the line
$ Move to the end of the line
/string Search forward for string
?string Search back for string
n Search for next instance of string
N Search for previous instance of string

11. nano commands:

nano is like notepad in windows
ctrl + x exit nano

Cluster Remote Access (ASU SAGUARO Cluster as an example):

1. Use putty or MAC-terminal or other similar software to connect the ASU SUGUARO CLUSTER
   Command: ssh yourname@saguaro.fulton.asu.edu
   logout - for exist the cluster
2. Before you use the cluster, you should claim a core to run your code, otherwise, you can't even compile your code.
   Use Command to get a core assigned for you: module load intel
3. How to upload file?
   Command: scp LocalFileName yourname@saguaro.fulton.asu.edu:
   Note scp does not work in putty on windows.
4. How to download file?
   Command: scp yourname@saguaro.fulton.asu.edu:Route/FileName .
   The dot in the end of the command means the current folder, Route means the files' route that you want to download. If it is the home route, then just ignore it.
   You should use this command in your local PC instead of the cluster.
5. Use SSH software to transfer files between the remote cluster server and ur local machine. It's very convenient. Using SSH can open a command window like putty.

Note SSH is for windows machines.

6. In Makefile, use \ to connect a following line, remember no space after the back slash \
   e.g.
   ifort line1code \
   line2code these two lines mean: ifort line1code line2code
   in Makefile, it would eat the first dollar ($)
7. qsub –I login a node
   qsub jobName.pbs submit your work to the cluster, in this batch file, you could define how many nodes you need, the wall-clock time and so on. Use qstat to check your job's status: Q: queue, R: running, C: complete, E: error
8. if you try to use MPI, you can use the following commands:
   load module intel-mpi/4.0 (use 4.1 instead of 4.0, you can get the latest version of the Intel compilers)
   make (if you already write a Makefile)
   qsub –I
   module load intel-mpi/4.0
   export FOR_COARRAY_NUM_IMAGES=4 (set the images number at least it could work for coarray parallel computing)
   ./executefilename < data.dat (run your program)

FORTRAN

1. Fortran read data by rows (line after line), but store them by column
2. ifort hello.f90 –o hello ! create 'hello', execute file
   ./hello ! run hello
   NOTE you can use X-code on your Mac
3. ifort –c file1.f90
   ifort –c file2.f90
   ifort –c file3.f90
   test program
4. for Gfortran: gfortran –o HelloWorld HelloWorld.f90

Software Demo:

https://www.youtube.com/watch?v=qmZpTsIqImo

(i) **Putty**, at t=10mins, (ii) **WinSCP**, at t=17:29mins, and (iii) **Cygwin**, at t=21:40mins.

Disclaimer:

The author doesn't make any warranty for the accuracy, completeness, or usefulness of any information disclosed and the author assumes no liability or responsibility for any errors or omissions for the information (data/code/results etc) disclosed.