Intermediate O2

HMS Research Computing

We'll allow a few minutes for people to join the class.

Slides also at: github.com/hmsrc/user-training

IntermediateO2_Spring2020.pdf

Course Objectives

- Transferring data with rsync
- Linux tools
- Bash "for" loops
- Handling command output
- Customizing your O2 account environment
- SLURM deeper dive
- Cron
- Q&A

Housekeeping

- First Zoom version of this class!
  - It is not being recorded.
  - Audio/Video of attendees is disabled.

- Zoom Chat anytime during class: for technical problems or general questions.
  - If it's not a quick answer, we can try to discuss at the end of class.

Resources

- O2 docs:
  - https://wiki.rc.hms.harvard.edu/display/O2/
- Group Website
  - http://rc.hms.harvard.edu
  - *will be migrating to: https://it.hms.harvard.edu/rc
- Get Help:
  - rchelp@hms.harvard.edu
- HMS RC Office Hours: every Wed from 1-3pm via Zoom until we return to campus:
  - https://rc.hms.harvard.edu/office-hours

Stay informed about O2

O2 Status Page
- to view ongoing outages and scheduled maintenance
- https://wiki.rc.hms.harvard.edu/display/O2/O2+Cluster+Status

Twitter
https://twitter.com/hms_rc/

o2-announce email list
- Required for all O2 users
- Service outages, scheduled maintenance, other news

Message of the Day (MOTD)
- That message in the terminal you see when you login to O2.

O2 and the HMS VPN

You can login to O2 without using VPN

- Duo (2FA) authentication required for each login

VPN can not support large file transfers

- Very limited bandwidth. Even 100 MB can be problematic
- <u>Log out of VPN</u> when copying data between your desktop and O2, or any HMS filesystem (e.g. research.files.med.harvard.edu)

Please only use VPN when you need to!

https://it.hms.harvard.edu/our-services/network-and-servers/vpn

- HMS IT may kill any processes which are impacting VPN service.

Login to O2

If you don't have an O2 account, we can assign you a temporary one for the class.

O2 data transfer: which tool to use?

	Local	Remote	Not supported
Tools	**cp** **rsync**	**sftp** **scp** **rsync** **wget** **ftp** [more]	<u>**Inbound**</u> FTP and generally anything which not sending over SSH (port 22).

rsync: most common use

- Local on O2:

  $ rsync -av source/ destination/

  ```
  -a (-rlptgoD , recursive and preserves permissions)
  -v (verbose)
  ```

- Over a network to O2:

  $ rsync -av -e ssh source/ user@transfer.rc.hms.harvard.edu:destination/

  ```
  -z (data compression) option may be useful
  ```

- Dry run (-n): test your command without actually copying data

  ```
  $ rsync -n -av source/ destination/
  ```

Exercise: rsync

- Copy the class directory with rsync: *(dry run: -n)*

  $ rsync -n -av /n/groups/rc-training/o2_intermediate ~/

- For real:

  $ rsync -av /n/groups/rc-training/o2_intermediate ~/

  Note that adding a trailing slash on the source directory will have rsync only copy the files within, not the directory itself.

rsync: more options

- Synchronize directories (be careful !!)

  $ rsync -delete -av source/ destination/

  ◦ this **overwrites and deletes** files in the destination
    which don't match what is in the source.

- Set permissions

  $ rsync -chmod=ug+rw [..]

- Exclude patterns or a list of files from transfer:

  $ rsync -exclude '*.bam' [..]

  $ rsync -exclude-from 'exclude-list.txt' [..]

Command line shortcuts

- Tab   autocomplete filename / command
- Ctrl + c   kill command you are currently running
- Ctrl + a   move to the beginning of the line
- Ctrl + d   logout
- Ctrl + e   move to the end of the line
- Ctrl + k   erase line to the right
- Ctrl + l   clear the terminal
- Ctrl + u   erase line to the left
- Ctrl + w   erase word to the left
- [arrow keys]   move cursor, browse command history

head / tail / less / more / cat

- Commands to view text in a file or stream.
- Exercise: examine contents of a data file

```
$ cd ~/o2_intermediate/data
$ cat example.gtf
$ head example.gtf
$ head -20 example.gtf
$ tail example.gtf
$ tail -20 example.gtf
$ tail -f example.gtf      (CTRL-C to quit)
$ more example.gtf        ("q" or CTRL-C to quit, "return" or "space" to scroll)
$ less example.gtf        ("q" to quit, arrows and other keys for navigation)
```

ln

- A link is a special file type
    - ln with the -s option is the most common use: "symbolic"
    - Symbolic links work across filesystems

- Example / Exercise:
    ```
    $ mkdir work              (make a directory)
    $ ln -s work shortcut     (make a link called "shortcut")
    $ ls -l                   (lower-case "L" file type)
    ```

find

- find [path to search] [expression] [actions]

    - -name : the filename / pattern
    - -user   : user owner
    - -group : group owner
    - -type   : type of file (plain file, directory, pipe. etc)
    - -ctime : time of file creation
    - -atime : last access time of a file
    - -mtime : last modification time of a file
    - -exec [command] : runs a command against find's output

find: examples

- List all files matching the name *.bam

    $ find ./dir -name '*.bam'

- Make all files group-writable under a directory:

    $ find ./dir -type d -exec chmod -v g+rwxs {} \;

    $ find ./dir -type f -exec chmod -v g+rw {} \;

    $ find ./dir -exec chgrp -v labgroup {} \;

- Remove files not updated in the past 60 days:

    $ find ./dir -mtime +60d -exec rm -v {} \;

find: exercise

- Create symbolic links to all bam files located under a directory tree:

```
$ cd ~/o2_intermediate
$ find . -name '*.bam'
$ find . -name '*.bam' -exec ln -s {} \;
$ ls -l
```

- Don't delete these links — we'll use them later!

wc

- word count
  - -l   print number of lines
  - -w print number of words

- Example: (how many lines are in a file)

  $ cd ~/o2_intermediate/data
  $ wc -l example.gtf

du

- estimate file space usage
  - [default]   print summary size only (Kb)
  - -a   print usage of all files
  - -h   print human readable format (Kb/Mb/Gb/Tb)

- Example: (how many lines are in a file)

  $ cd ~/o2_intermediate/data
  $ du -h example.gtf
  $ du -a
  $ du -ah

Commands for Text Processing

RESEARCH
COMPUTING
https://rc.hms.harvard.edu/

HARVARD
MEDICAL SCHOOL

Information Technology   20

sort

- sort lines of text

    $ sort   file.txt

- a few common options:

    - -r   (reverse order)
    - -h   (human numeric sort: e.g. 2K, 1G, 500M)
    - -u   (remove duplicate lines)

Exercise: sort

```
$ cd ~/o2_intermediate
$ cat  sort.txt
$ sort  sort.txt
$ sort  -r  sort.txt
```

uniq

- report or omit repeated lines

  $ uniq   file.txt
  - with no options, uniq prints all lines but removes duplicate entries

- a few common options:

  - -i   (ignore case)
  - -c   (prefix lines by number of occurrences)
  - -d   (print only repeated lines)
  - -u   (print only unique lines)

Exercise: uniq

Try these commands:

$ cd ~/o2_intermediate
$ cat   uniq.txt
$ uniq   uniq.txt   (remove duplicate entries)
$ uniq -d uniq.txt   (show duplicates only)
$ uniq -u uniq.txt   (show unique entries only)
$ uniq -c uniq.txt   (unique entries with count)

RESEARCH
COMPUTING
https://rc.hms.harvard.edu/

HARVARD
MEDICAL SCHOOL

Information Technology   24

grep (global regular expression print)

- print lines matching a pattern

  - $ grep   pattern file.txt
  - $ grep '#pattern 2'   file.txt

- a few common options:

  - -i   (case-insensitive)
  - -v   (does not match the pattern)
  - -n   (precede matching line with a line number)

Exercise: grep

$ cd ~/o2_intermediate/data

$ grep stop_codon example.gtf

$ grep -v stop_codon example.gtf

$ grep -n stop_codon example.gtf

$ grep -i cds example.gtf

cut

- remove sections from each line in a file / stream
  - -d defines delimiter (default is a Tab)
  - -s prints only lines containing a delimiter
  - -f prints specified fields

- Examples:
  - $ cut -f 1 file.txt   (print 1st field only)
  - $ cut -f 1,3 file.txt   (print 1st & 3rd fields)
  - $ cut -s -d ":" -f 1 file.txt   (colon space delimiter)
  - $ O2squeue | cut -s -d " " -f 1 (list of O2 job IDs)

Exercise: cut

- remove sections from each line in a file / stream
- default delimiter is a Tab

```
$ cd ~/o2_intermediate/data
$ head example.tab
$ cut -f 1,2 example.tab | head
$ cut -f 3,4 example.tab | head
```

paste

- Write lines consisting of the sequentially corresponding lines from each FILE, separated by TABs, to standard output.

  - -d   defines delimiter (default is a Tab)

- Examples:
  $ paste file1.txt file2.txt
  $ paste file1.txt file2.txt > out.tsv   (tab separated file)
  $ paste -d , file1.txt file2.txt > out.csv   (comma separated file)

awk and sed

- ## awk
  - a special-purpose programming language for text processing
  - Does similar things as perl, but sometimes more quickly.
  - awk has been around "forever", so lots of tutorials online.

- ## sed
  - a stream editor that works on a per-line basis.
  - Example: global substitution of the string "Harvard" -> "HMS"

    ```
    $ sed 's/Harvard/HMS/g' doc.txt > doc_new.txt
    ```

Working with Command Output

Command output redirection:

- ## Redirect: >
  - sends output to a file, overwrites any existing file
    $ grep pattern file.txt > out.txt

- ## Append: >>
  - sends output to a file, appends to any existing file
    $ grep pattern file.txt >> out.txt

- ## Pipe: |
  - sends output to be input for another application
    $ cut -1 file.txt | sort | uniq -c

Exercise: handling command output

- Sort field entries from a data file (example.gtf)
- default delimiter is a Tab

```
$ cd ~/o2_intermediate/data
$ cut -f 4 example.gtf > out.txt
$ grep -i cds  example.gtf >> out.txt
$ cut -f 4 example.gtf > out.txt
$ cut -f 4 example.gtf  |  sort -n | uniq -c
$ grep  stop_codon  example.gtf  |  wc -l
```

Redirecting Standard Error (stderr)

- bash syntax:

  $ command 2>out.err   (send stderr to a file)

  $ command 2>&1   (send stderr to stdout)

  $ command > out.txt 2>&1   (send stderr and stdout to a file)

- Exercise:

  $ cd ~/o2_intermediate

  $ cat no.txt   (file does not exist — error)

  $ cat no.txt 2>out.err   (saves stderr to a file: out.err)

Customizing your O2 account

HARVARD
MEDICAL SCHOOL

Customizing your O2 account

- Aliases: create your own commands!

  $ alias ll='ls -la'

  $ alias h=history

- Change your default umask

  - Example: create group-writable files by default:

    $ umask 0002

- Set, environment variables like command path:

  $ export PATH=$PATH:/home/user/bin

Adding customizations on login

- ## ~/.bash_profile

  - executed on login
  - executed once before you get a prompt.

- ## ~/.bashrc

  - Supplemental config file, executed each time you run "bash"
  - On O2, gets run from ~/.bash_profile
  - Typically, this is where most customizations go:
  - aliases, modules, $PATH, other variables, etc.

Sample ~/.bashrc file

```
$ cat ~/.bashrc
#
alias ll 'ls -la'
alias h history
#
module load gcc/6.2.0
module load R/3.5.1
#
export PATH=$PATH:/home/user/bin
export DUO_PASSCODE=push
```

RESEARCH
COMPUTING
https://rc.hms.harvard.edu/

HARVARD
MEDICAL SCHOOL

Information Technology   38

Exercise: edit your .bashrc file

$ nano ~/.bashrc

(Add some things you would like to set automatically on login)

$ source ~/.bashrc

(to manually run it without having to re-login)

Try it out! (Run an alias command, etc)

bash "for" loops

Automate commands with a "for" loop

- Repeat commands against an designated list
  - this syntax is for bash, but other shells (tcsh) are different

- Examples

  `$ for i in 1 2 3 ; do mkdir $i ; done`

  `$ for i in `cat list` ; do cp $i  ~/work ; done`

  - more complex loops can be put in bash scripts
  - also useful for submitting batches of jobs to O2!

"for" loop in a shell script

```
#!/bin/bash

list=/home/user/files.txt

for i in `cat $list`
   do
        [command 1]
        [command 2]
   done
```

Research
Computing
https://rc.hms.harvard.edu/

HARVARD
MEDICAL SCHOOL

Information Technology   42

more about Slurm…

Job Monitoring

```
$ O2squeue
$ squeue -u your_user
$ squeue -u your_user -t PENDING
$ squeue -u your_user -t RUNNING
$ squeue -u your_user -p short
$ scontrol show jobid <jobid>
  (for more details)

$ O2sacct
$ sacct -j <jobid>
```

Jobs with command line arguments

- Run the following:

      $ cd ~/o2_intermediate
      $ sbatch arguments.sbatch hello
      $ O2squeue   (to view job status)

- The output file will contain the argument "hello"

- This technique gets more useful when submitting from a script and the arguments vary over iterations.

Jobs with command line arguments

```
#!/bin/bash

#SBATCH -p short    #partition

#SBATCH -t 0-01:00 #time days-hr:min

#SBATCH -o %j.out  #out file

#SBATCH -e %j.err  #error file


echo $1
```

A better example (bamsort.sbatch)

```bash
#!/bin/bash
#SBATCH -p short    #partition
#SBATCH -t 0-01:00 #time days-hr:min
#SBATCH -o %j.out  #out file
#SBATCH -e %j.err  #error file

## Update path below and uncomment for your account:
## dir=/home/rc_training000/o2_intermediate/data

module load gcc/6.2.0
module load samtools/1.9

samtools sort $1 > $dir/"${1%.*}".sorted.bam
#where $1 is a bam file
```

Using sbatch with a bash "for" loop

- To submit a bunch of separate jobs systematically:

    $ for i in [input] ; do [sbatch command] ; done

- Exercise (remember those symbolic links?):

    $ cd ~/o2_intermediate
    $ for i in *.bam ; do sbatch bamsort.sbatch $i ; done

Canceling one or more job

The [-u] option is always required.

```
$ scancel -u your_user
$ scancel -u your_user -v[vv]
$ scancel -u your_user -p short
$ scancel -u your_user -t PENDING
$ scancel -u your_user -t RUNNING
$ scancel -u your_user -t SUSPENDED
$ scancel -u your_user JOBID1 JOBID2 [..]
```

Cron

Process automation: cron

- Task Scheduler for Linux

- O2 has a centralized cron server where jobs get executed.

- Examples:

    - Automate a nightly rsync process
    - Run a weekly analysis report
    - Purge old files on a schedule

Cron: Editing a Crontab

- Create/Edit a crontab <u>from a login server</u> using: **crontab -e**

- Format of a cron job process:

  [Minute] [Hour] [Date] [Month] [Day of the Week] Command
  Asterisk (*) = "every"

- Example: have a job run at 2:00am every Monday:

```
0 2 * * 1 sbatch /home/user/rsync.sbatch
```

Thank you!

- The Harvard Training Portal will be emailing you a short survey about the class. Please complete it so we can learn what works, what needs improvement, and what you'd like to see offered in the future!