Introduction to Linux and Cluster Computing Environments for Bioinformatics

Doug Crabill
Senior Academic IT Specialist
Department of Statistics
Purdue University
dgc@purdue.edu

What you will learn

- Linux Supercomputer overview
- Basics of a Linux shell, including moving/editing/creating/deleting files, how to launch/terminate programs, check progress
- Basic shell scripting, parallel execution
- Fundamentals of cluster supercomputer use
- Example of scaling things up

The rice.rcac.purdue.edu cluster

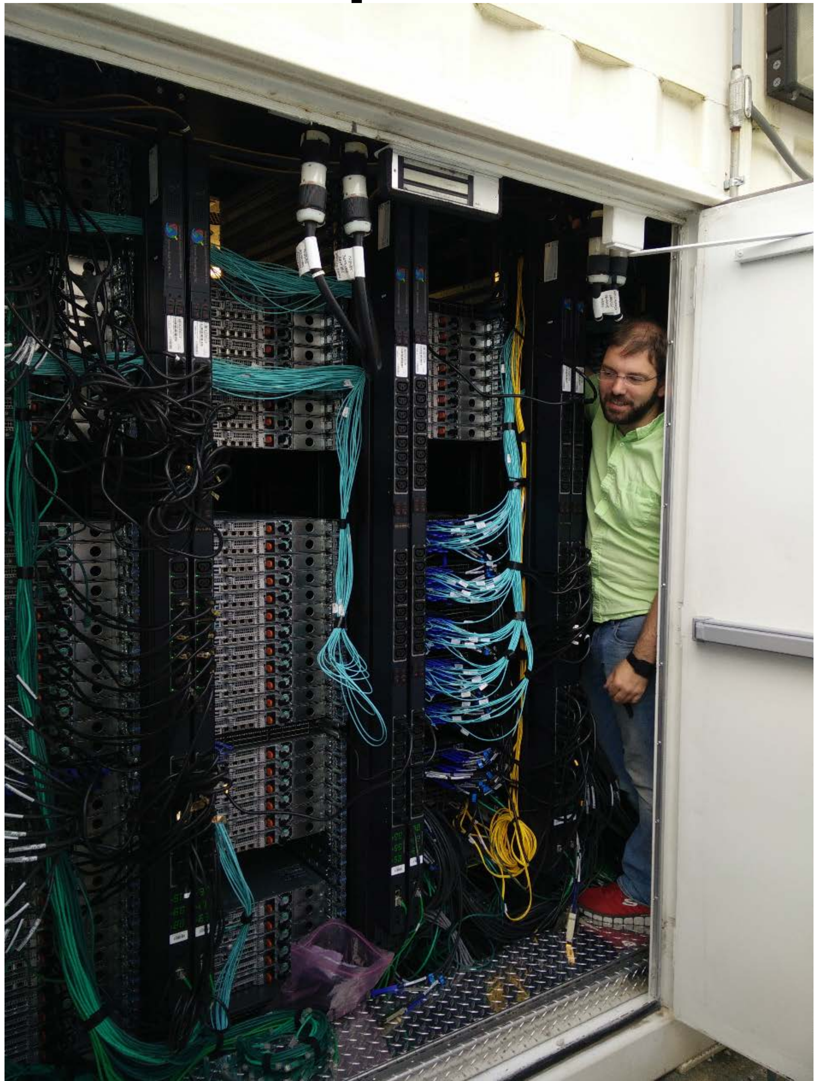
The rice.rcac.purdue.edu cluster

An individual node

The brown.rcac.purdue.edu cluster

The brown.rcac.purdue.edu cluster

The brown.rcac.purdue.edu cluster



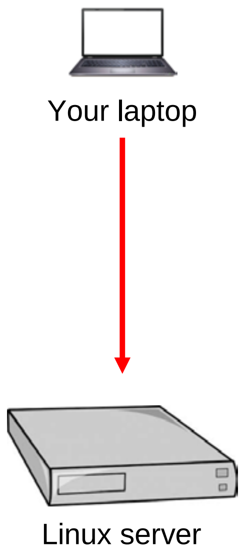
Brown supercomputer stats

- 550 Nodes, 13,200 total CPU cores
- Each nodes has 24 CPU cores, 96GB RAM
- 3.4 Petabytes of scratch space for this cluster alone
- 4.5 Petabytes of long term storage shared among all clusters
- Currently #302 on top500.org, Conte is #190.

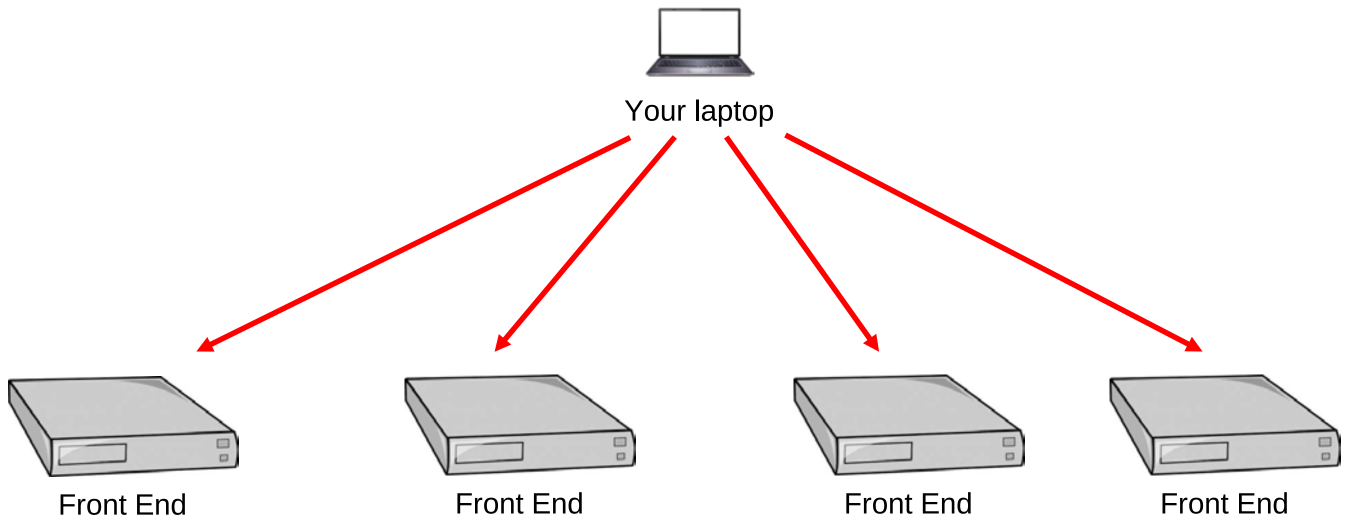
Anecdote time!

- A colleague was working on a game theory problem…

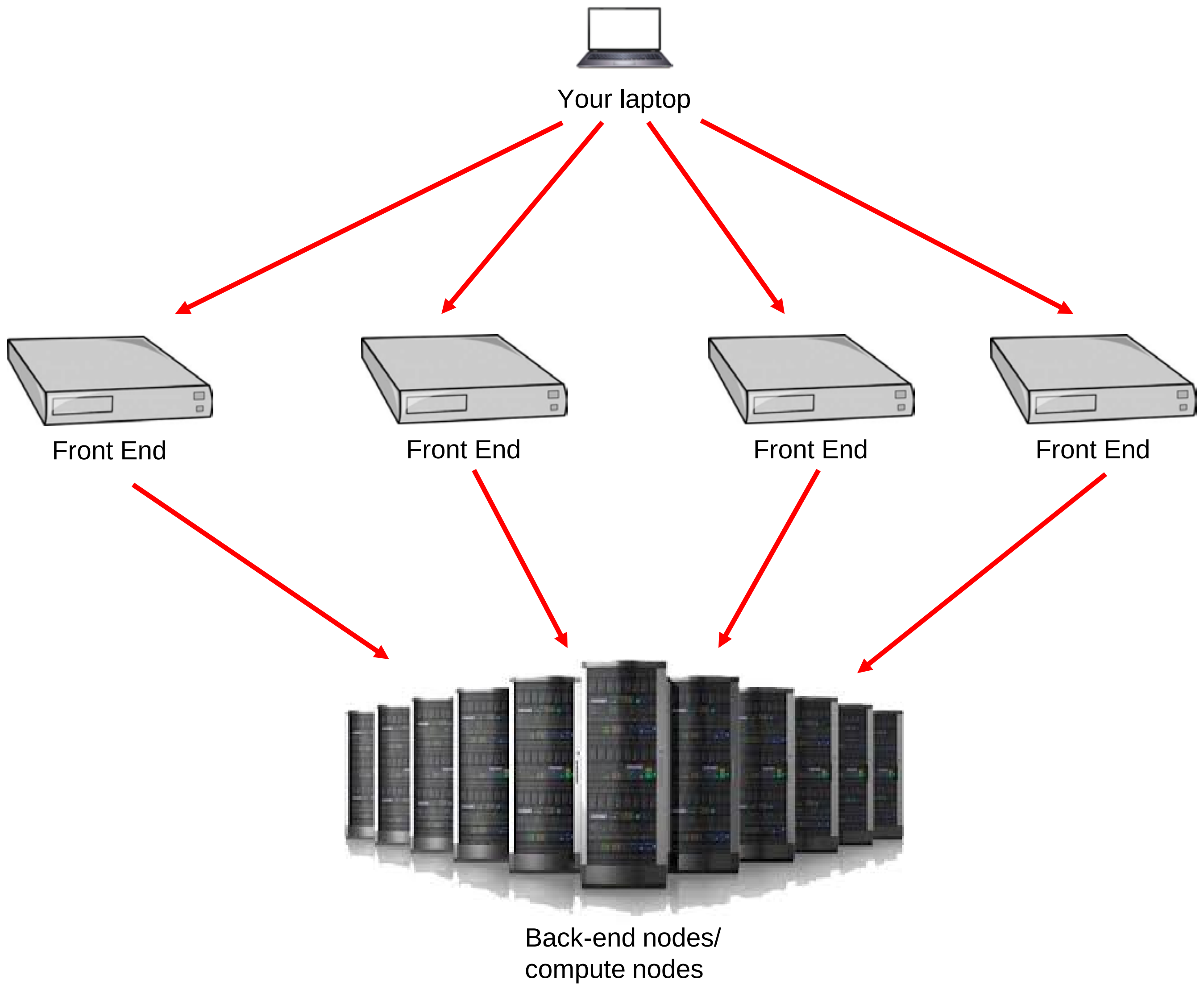
Linux and Cluster Computing Environments



Linux and Cluster Computing Environments

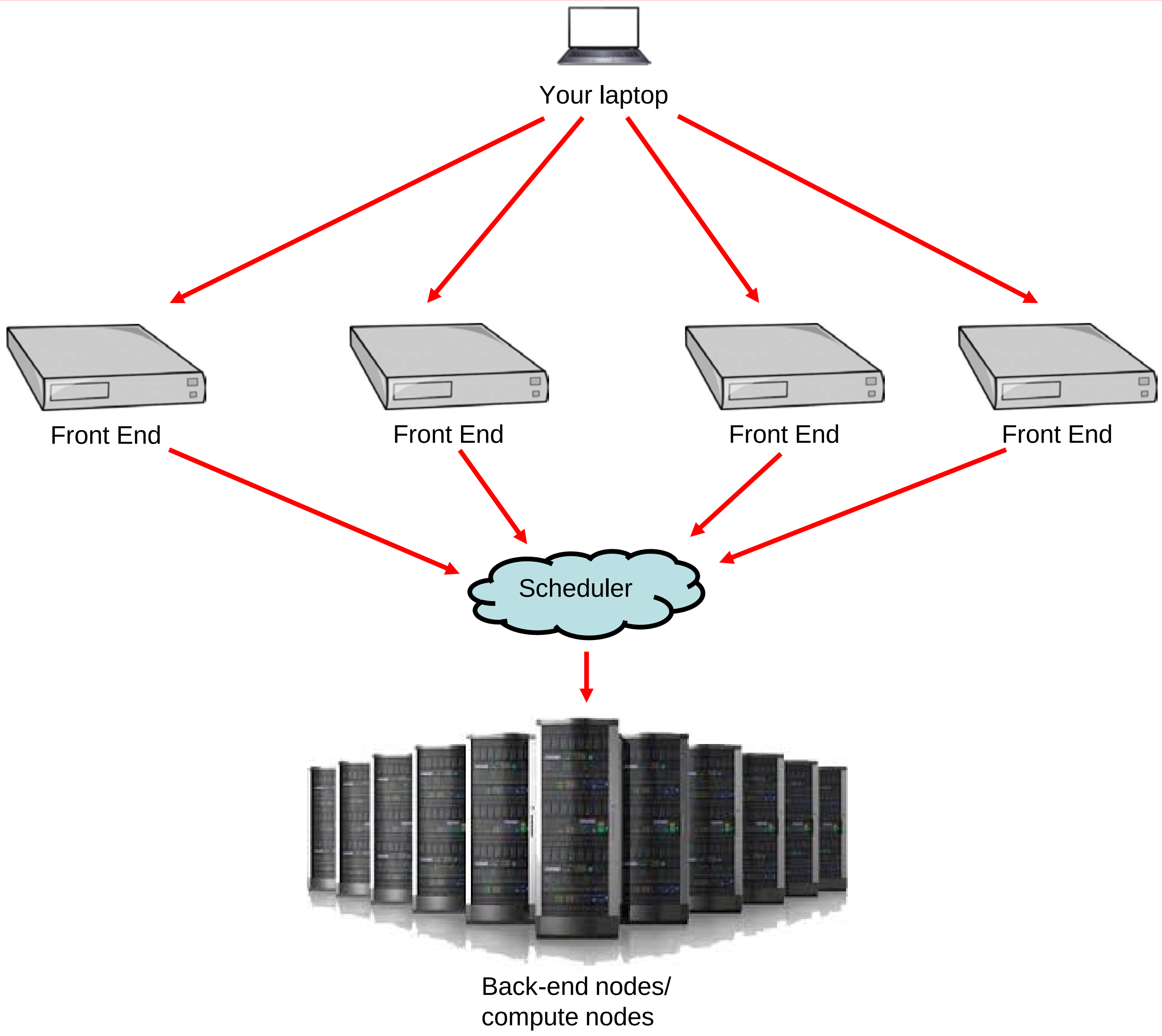


Linux and Cluster Computing Environments



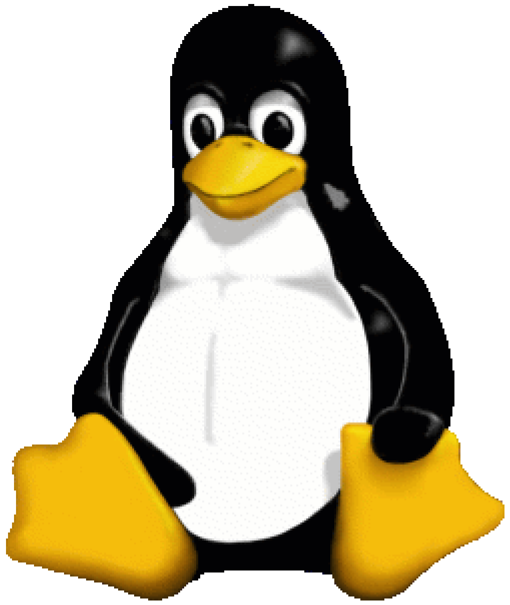
PURDUE UNIVERSITY

Linux and Cluster Computing Environments



Why Linux?

Why Linux?



Why Linux?

- Can be desktops, but tend to be larger servers in some remote, environmentally controlled data center (or pod!)
- Multiple CPU cores per server (~8-44)
- Large amounts of RAM (64GB – 1TB is common)
- Multiple users can use the same computer simultaneously

Why Linux? (cont.)

- Can interact with a graphical interface
- More common to interact with a text based interface
- Servers tend to stay up for a long time between reboots (months)
- Commonly launch programs and walk away for days, weeks, or months as they run
- Computations can scale up as servers added

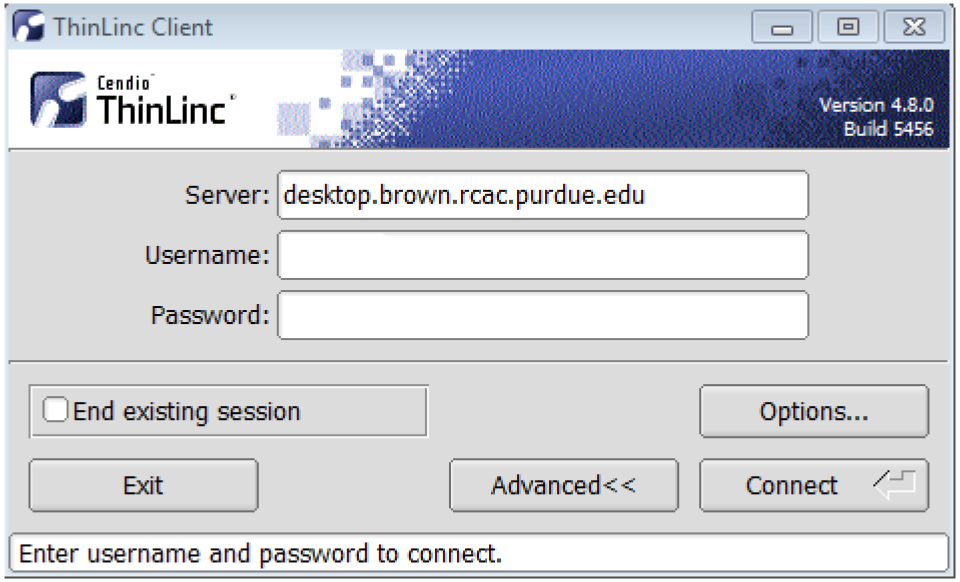
But where are the keyboards, mice, and monitors?

But where are the keyboards, mice, and monitors?

ThinLinc Linux graphical interface

- We will use ThinLinc to provide a graphical user interface on a Brown front-end
- From the front-end we'll connect to a Brown node, aka back-end node, aka compute node, where we will do the real computing
- The ThinLinc client is free (and better), but you can actually use a web browser instead

Logging in via ThinLinc Client



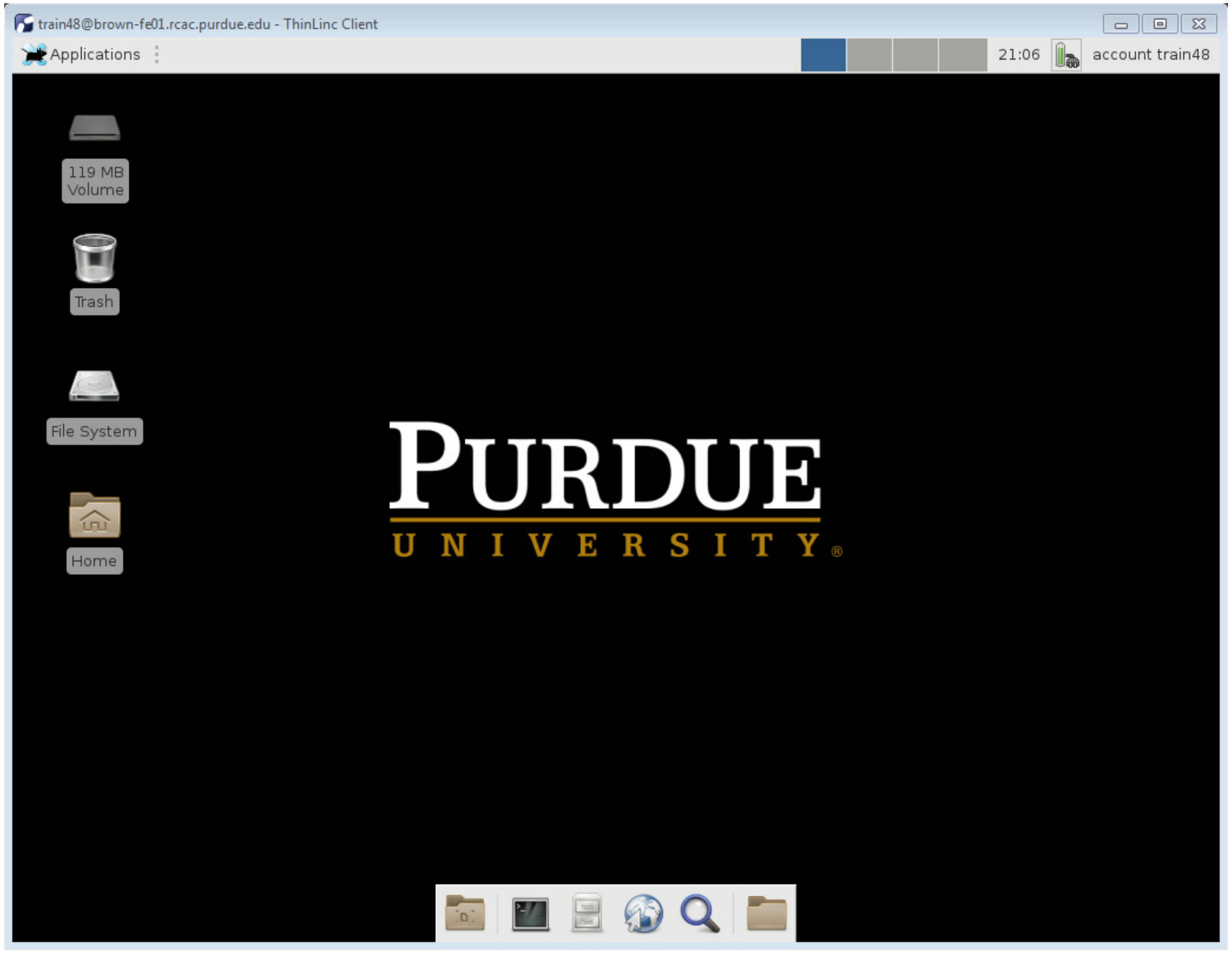
Server: desktop.brown.rcac.purdue.edu

Username:

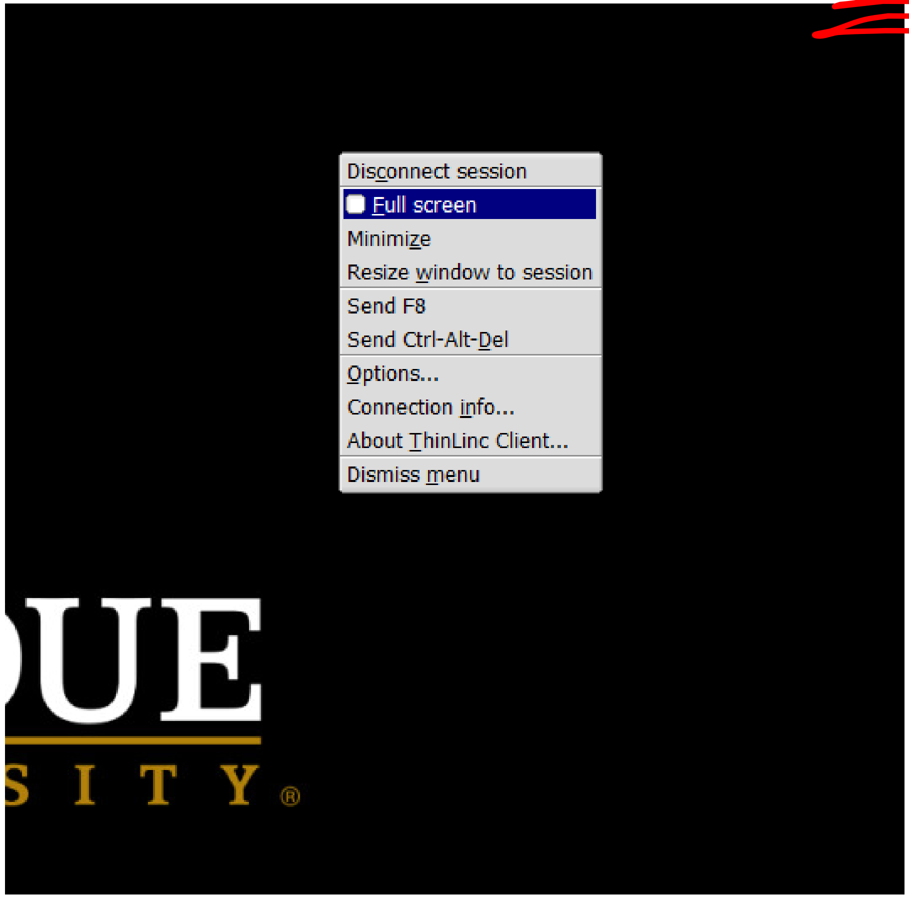
Password:

Enter username and password to connect.

Connected!!!


train48@brown-fe01.rcac.purdue.edu - ThinLinc Client
Applications
21:06
account train48
119 MB Volume
Trash
File System
Home
PURDUE
UNIVERSITY

Toggle full screen on ThinLinc client by pressing the F8 key



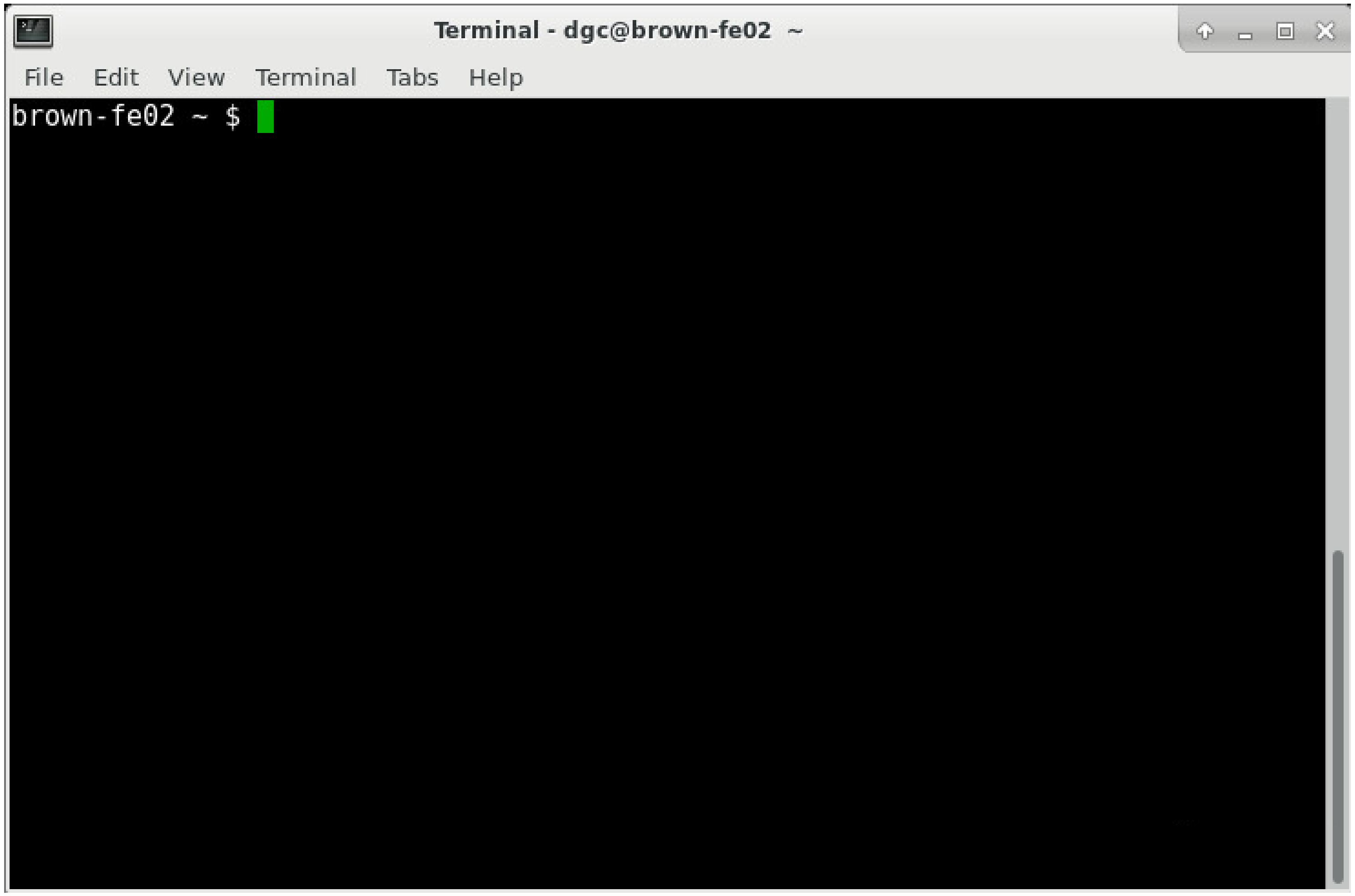
ThinLinc sessions can persist!

- Programs/windows that are open and running can persist after closing the ThinLinc Client
- Smile patiently while I demonstrate persistence
- If you explicitly click Applications->Log Out you will be logged completely out and application state will not persist

What is a “shell”?

- A text-based user interface used to launch programs. The shell we use is called “bash”
- Used to launch programs, pass arguments to programs, specify input/output files
- Terminal is one way of accessing a shell
- Launch via Applications->Terminal Emulator or Applications->System->Xfce Terminal (my preferred method)

A Terminal


Terminal - dgc@brown-fe02 ~
File Edit View Terminal Tabs Help

```
brown-fe02 ~ $
```

Multiple Terminal windows

- You can have many Terminal windows open at once
- To open an additional Terminal window on the same server as an existing Terminal, type:

```
    xfce4-terminal &
```

- If you omit the &, the first Terminal cannot be used again until the second is closed
- Type `exit` to log out of a shell

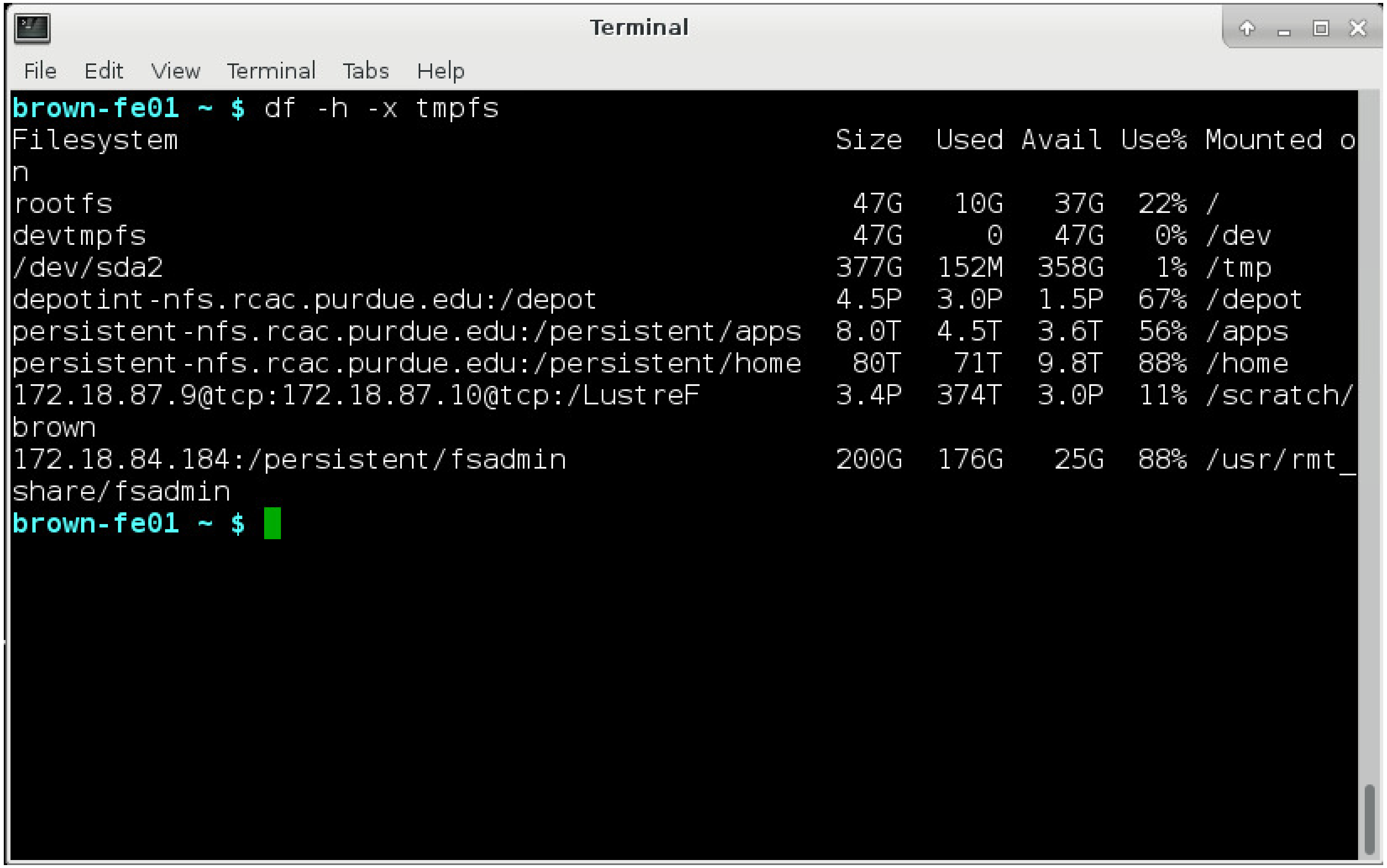
Using copy/paste

- Using the Windows shortcuts Control-C and Control-V will generally not work, because those keys mean other things under Linux
- Either select the text and select Edit/Copy and then Edit/Paste
- Or select the text which implicitly copies it, and press down on the mouse wheel to paste (don’t roll it, press down like it’s a button)

Filesystems

- Filesystems on Linux similar to network drives on Windows, but without drive letters
- Example directories on different filesystems: /home/dgc, /depot/nihomics, /scratch/brown/dgc
- Hierarchical. Directory names separated by “/”, not by “\” as with Windows. Avoid spaces in filenames and directory names.

Filesystems on Brown



```
brown-fe01 ~ $ df -h -x tmpfs
Filesystem                                        Size  Used Avail Use% Mounted o
n
rootfs                                             47G   10G   37G  22% /
devtmpfs                                           47G     0   47G   0% /dev
/dev/sda2                                         377G  152M  358G   1% /tmp
depotint-nfs.rcac.purdue.edu:/depot               4.5P  3.0P  1.5P  67% /depot
persistent-nfs.rcac.purdue.edu:/persistent/apps   8.0T  4.5T  3.6T  56% /apps
persistent-nfs.rcac.purdue.edu:/persistent/home    80T   71T  9.8T  88% /home
172.18.87.9@tcp:172.18.87.10@tcp:/LustreF         3.4P  374T  3.0P  11% /scratch/
brown
172.18.84.184:/persistent/fsadmin                 200G  176G   25G  88% /usr/rmt_
share/fsadmin
brown-fe01 ~ $
```

Shell features

- Shell environment variables used to control settings for how certain things work
- Thousands of potential commands can be executed
- Commands available varies from one Linux computer to the next, depending on what has been installed, and the value of your PATH environment variable

Shell features (cont.)

- Filename completion (using “Tab” key)
- Command completion (using “Tab” key)
- Command line editing using arrow keys (up-arrow key to go to the previous command)

Let’s get dirty!

Listing files in Terminal

- Type `ls` to list files in the current directory
- Type `ls -l` to list files with more detail
- Type `ll` to list files with even more detail

Navigating directories in Terminal

- Type pwd to see full path to current directory
- Type `cd` *`dirname`* to change directories
- Type `cd ..` to go to the parent directory, or `cd ../..` to go to the grandparent, etc.
- Type `cd ~` to go to your home directory
- `cd /depot/nihomics/data`
- Absolute paths start with `/`, relative paths are relative to the current directory

Special directories

- `/home/`*`USERNAME`* – Your home directory, where source code, programs, and final results go
- `/scratch/brown/`*`USERNAME`* – Enormous scratch directory. Can place original data sets and intermediate results there
- Type *`myquota`* to see used disk space and limits

Editing, copying, moving files

Editing, copying, moving files

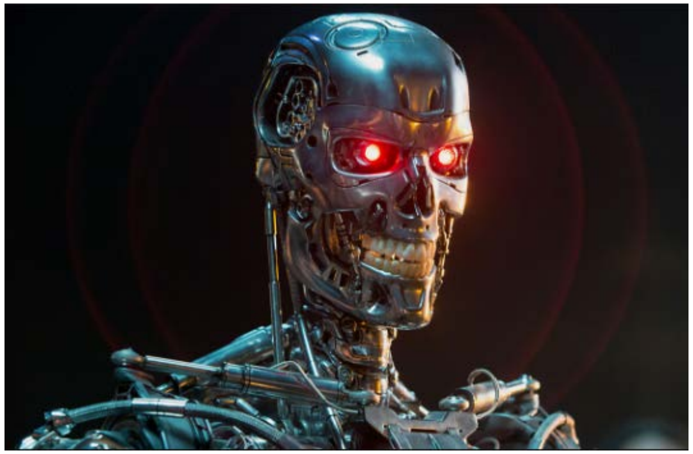
- `gedit` *`filename`* – Edits *`filename`*
- `mv` *`oldname newname`* – Moves a file or directory, possibly to a new directory, possibly renaming the file or directory in the process
- `cp` *`oldname newname`* – Copies files
- `cp -r` *`olddir newdir`* – Copies `olddir` and all files and subdirectories within to `newdir`

Create/Remove directories, files

- `rm` *`filename`* – removes *`filename`*
- *`mkdir dirname`* – creates *`dirname`*
- *`rmdir dirname`* – removes *`dirname`*, but only if *`dirname`* is empty
- Let’s practice, and use filename completion and command line editing while we are at it!

Terminating a program

- If you are running a program in a terminal window that you would like to terminate, press Control-C
- This won’t work if you started that program it with an &



See what programs are running

- `ps xuww` - Show what programs we are running now
- PID column shows the Process ID of each program
- Can use `top` to see most CPU intensive programs currently running by everyone on this server. Press q or just control-c to exit `top`

Terminate or `kill` or program

- Must first know the process id number (PID) using either `ps xuww` or `top`
- `kill NNNNN` Will kill most programs
- `kill –HUP NNNNN` Use if the previous doesn’t work
- `kill -9 NNNNN` Use if the previous doesn’t work

Let’s practice starting/killing progs

- On a Brown node, type `busy 1000 &`
- Type it again a few times (use the up-arrow!)
- Type `top` to see the PIDs of all the jobs running, press q to quit
- Kill all of the busy jobs by typing the PIDs *like*: `kill 24933 24937 24939 24944`
- Type `top` again to confirm they are gone

Redirecting input/ouput

- Some programs write output to the Terminal/shell screen
- We can save it using output redirection
- `qstat -a > out1` Saves results of the command `qstat -a` to the file `out1`
- `head < out1` See the first 10 lines of `out1`
- `head < out1 > out2` Save to `out2`

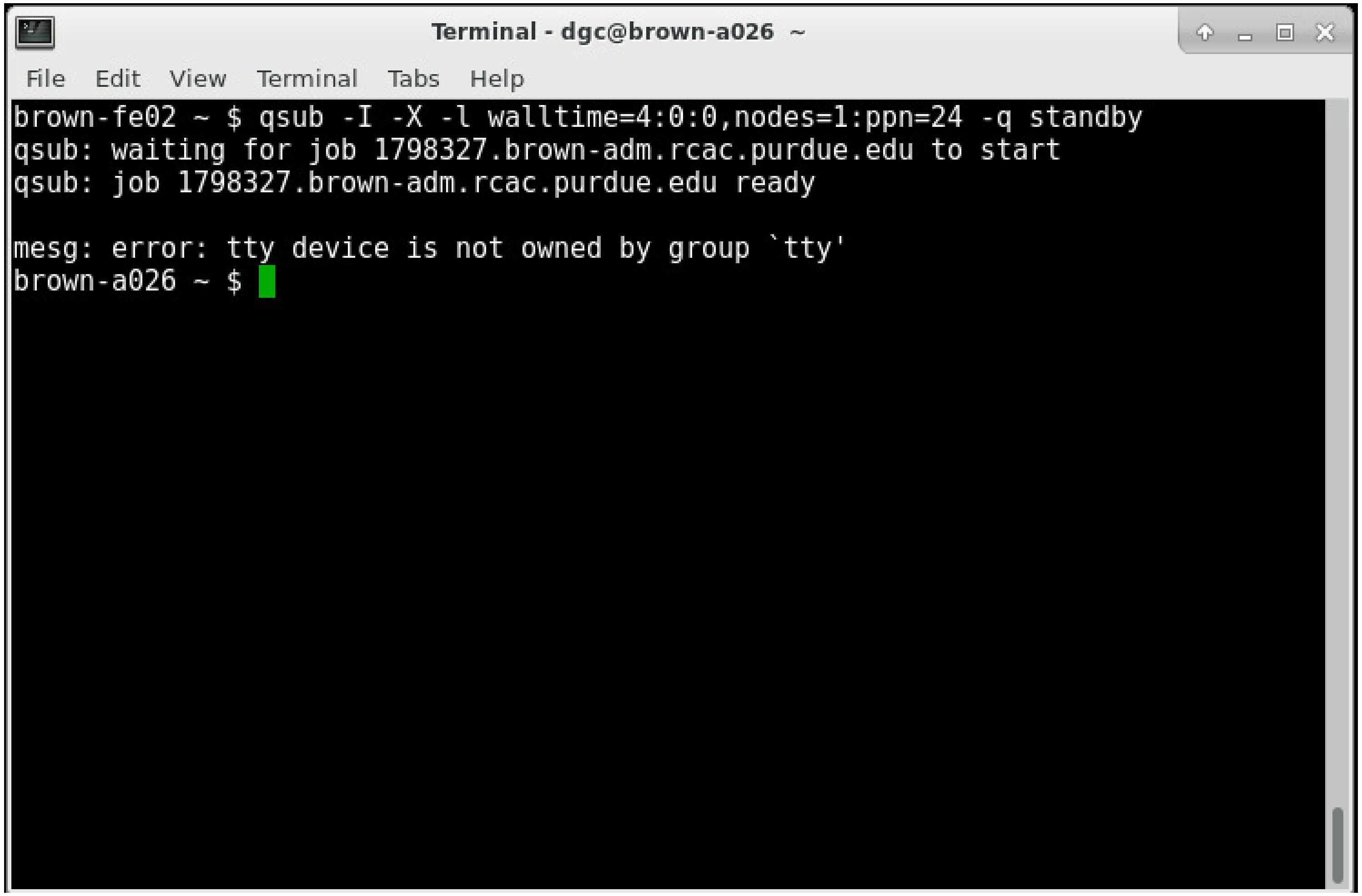
Redirecting input/ouput

- Can only save the text output that would have normally appeared on the screen. If a program wouldn’t normally generate any text output, nothing will be saved
- `Terminal > out3` (Nothing is saved!)

Interactive shell on back-end node

- So far we've been working only on a Brown front-end node. We really want a back-end.
- `qsub -I -X -l walltime=4:0:0,nodes=1:ppn=24 -q standby` (one long typed line)
- Now we have a whole single node to ourselves for interactive use – for 4 hours

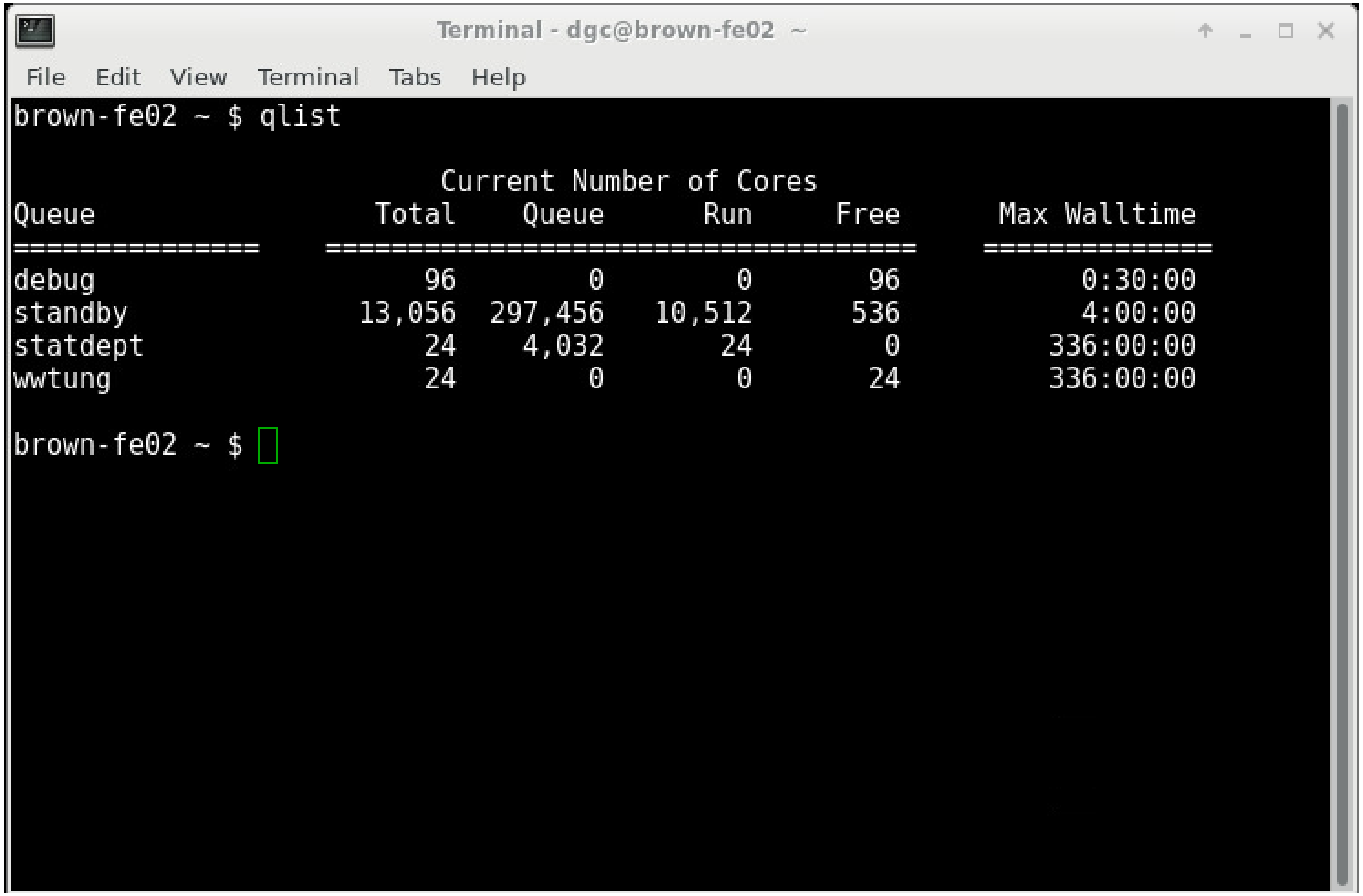
Interactive shell on back-end node


Terminal - dgc@brown-a026 ~
File Edit View Terminal Tabs Help

```
brown-fe02 ~ $ qsub -I -X -l walltime=4:0:0,nodes=1:ppn=24 -q standby
qsub: waiting for job 1798327.brown-adm.rcac.purdue.edu to start
qsub: job 1798327.brown-adm.rcac.purdue.edu ready

mesg: error: tty device is not owned by group `tty'
brown-a026 ~ $
```

Using qlist


Terminal - dgc@brown-fe02 ~
File Edit View Terminal Tabs Help

```
brown-fe02 ~ $ qlist

                          Current Number of Cores
Queue                 Total     Queue      Run      Free      Max Walltime
===============     ====================================    ==============
debug                    96         0        0        96           0:30:00
standby              13,056   297,456   10,512       536           4:00:00
statdept                 24     4,032       24         0         336:00:00
wwtung                   24         0        0        24         336:00:00

brown-fe02 ~ $
```

This talk continues at a later date