Introduction to Linux, Bash, and the command line

Robert Fridman

June 10 2024

UNIVERSITY OF CALGARY
Research Computing Services

UNIVERSITY OF CALGARY

Components of Interactive Productivity

Linux
BASH
The Command Line

UNIVERSITY OF CALGARY

Introduction to Linux

- Linux is a multitasking open source operating system that is well suited to process large amounts of work.
- The interaction with Linux consists of a large amount of ***individual executable programs*** that accomplish different things.
- The user interface to Linux is done through a ***shell***, which is another executable that is part of Linux. The shell is the command interpreter.
- There are different shells in Linux. We will discuss the ***BASH*** shell.

Connecting to a Linux system

- Make sure you are connected to WIFI by checking that you have an internet connection.
- All examples will be run on the TALC cluster. This is a dedicated compute cluster for teaching and learning.
- You will need a way to **Login** to TALC from your laptop
  - **Windows: MobaXTerm** – a graphical SSH client, download the free version at https://mobaxterm.mobatek.net/
  - **MacOS:** Use the **Terminal** app to SSH directly to TALC
    - **% ssh –l firstname.lastname talc.uclagary.ca**
- Make the window as large as you can on your screen.

The first login

```
================================================================
=                TALC Terms of Use                             =
================================================================

The Teaching and Learning Cluster (TALC) is a computing resource
provided by Information Technologies Research Computing Services (RCS)
...
TALC is not suitable for Level 3 and Level 4 data (such as personally
identifiable medical information).

RCS reserves the right to examine files, programs and any other
material used on RCS systems at any time without warning.

Evidence of inappropriate use of TALC may result in immediate loss of
--More--
```

The Terms of Use

…
If you agree to these Terms of Use, please acknowledge so by entering "Y" below. If you do not agree, please enter "N" below and write to support@hpc.ucalgary.ca with your questions and concerns.

Do you agree to the TOU you just read? (y/n): **y**
TOU accepted
[fridman@talc ~]$ exit

Login again

- % ssh talc.uclagary.ca

```
================================================================================

     _________    _     _____     ______
    | _   _ |  / \    |_  _|   .' ___  |
    |_/ | | \_| / _ \     | |    / .'  \_|
      | |    / ___ \    | |   _ | |
     _| |_ _/ /   \ \_ _| |__/ |\ `.___.'\
     |_____||____| |____||________| `.____ .'.ucalgary.ca
         Teaching And Learning Cluster

Filesystem Limits:
  Home Directory   50 GB quota & 250,000 file limit

Please note:
 - WE DO NOT PERFORM BACKUPS

================================================================================
```

The Message of the Day (MOTD)

SYSTEM NOTICES:
****

2024/05/07
--- Power Interruption

TALC Experienced an brief power outage around 11AM May 7, 2024.
Most compute nodes have or are rebooting. Most jobs running at this time were lost. Administrators are actively working on restarting compute nodes. Sorry for the inconvenience.

==========

Welcome to Linux

```
Last login: Fri Jun  7 21:22:43 2024 from 10.58.160.14
Your account quotas as of Fri Jun  7 21:27:04 MDT 2024:
Filesystem      Available       Used / Total          Files Used / Total
----------      ---------    -------- / ----------     ---------- / ----------
Home Directory   495.9GB        4.0GB / 500.0GB (0%)      0.0 Mil / 1.5 Mil (0%)
/scratch         15.0TB         0.0TB / 15.0TB (0%)         0.0 K / 1000.0 K (0%)
/tmp (on talc)   99.9GB         0.0GB / 100.0GB (0%)        0.0 K / Unlimited

You may check your quota using the 'arc.quota' command.

[fridman@talc ~]$
```

UNIVERSITY OF CALGARY

The first command to know in Linux

- All Linux executables come with a description of what the command does and how to use it, called the *man page*.
- The executable to read the description of what a command does is called ***man***.
- [fridman@talc ~]$ ***man man***
- Use the space bar to scroll

UNIVERSITY OF CALGARY

MAN(1) General Commands Manual

NAME

man, apropos, whatis – display online manual documentation pages

SYNOPSIS

man [-adho] [-t | -w] [-M manpath] [-P pager] [-S mansect] [-m arch[:machine]] [-p [eprtv]] [mansect] page ...

man -f [-d] [-M manpath] [-P pager] [-S mansect] keyword ...
whatis [-d] [-s mansect] keyword ...

man -k [-d] [-M manpath] [-P pager] [-S mansect] keyword ...
apropos [-d] [-s mansect] keyword ...

DESCRIPTION

The man utility finds and displays online manual documentation pages. If mansect is provided, man restricts the search to the
specific section of the manual.

UNIVERSITY OF CALGARY

Basic Linux Commands

- Show files
  - ls
  - ls –l
  - ls –a
  - ls –la
- Who is logged in to TALC
  - who
- Directories
  - mkdir *test test2*
  - rmdir test2
- Change directory
  - cd test

UNIVERSITY OF CALGARY

Command history

- All the commands that are executed get recorded.
  - Use the ***history*** command to list all the previous commands.
- To repeat the last command, use the UP-ARROW key on the keyboard.
- To edit the command line, use the LEFT-ARROW and RIGHT-ARROW keys to position the cursor.
- To move the cursor to the end of the line, use CONTROL-e
- To move the cursor to the beginning of the line, use CONTROL-a
- To search for a previous command, use the CONTROL-r key and type the starting characters, then hit enter to complete.

UNIVERSITY OF CALGARY

Power of Command Line

FIVE REASONS TO LOVE THE COMMAND LINE

The text interface is intimidating, but can save researchers from mundane computing tasks. Just be sure you know what you're doing. **By Jeffrey M. Perkel**

Jennifer Johnson's principal investigator had a simple request. Since 2018, their team had sequenced the DNA of some 1,300 silver fox (*Vulpes vulpes*) specimens, and the lab head wanted to know precisely how many bases it had collected, and how well those bases aligned to the reference genome.

available on Windows through such tools as the free 'Windows Subsystem for Linux' and MobaXterm, the command line (also called the shell) is a powerful text-based interface in which users issue terse instructions to create, find, sort and manipulate files, all without using the mouse. There are actually several

actions across multiple files. Johnson directed her terminal to scan her hard disk for sequencing data files, extract the needed information and compile them into a tidy spreadsheet. "That took me less than 10 minutes," she says; recomputing the data would have taken a full day.

https://www.nature.com/articles/d41586-021-00263-0

UNIVERSITY OF CALGARY

“With great power there must also come great responsibility"

-- Spiderman

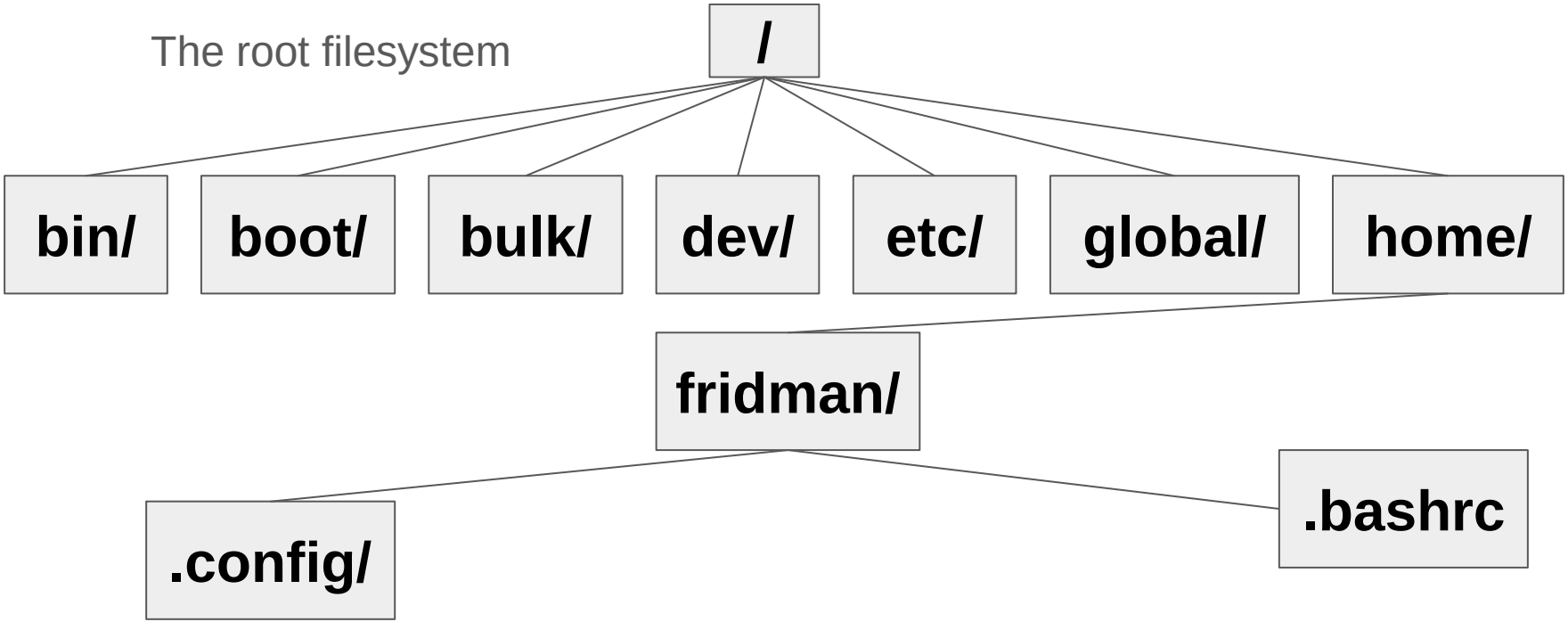
UNIVERSITY OF CALGARY

Why Command Line?

The *Command Line Interface (CLI)* allows repetitive tasks to be applied to an arbitrary number of activities through a special syntax built into the command shell. This can be applied to:

1. Files
2. Processes
3. Workflows
4. Packaging workflows for reproducibility and sharing
5. Ad hoc, free format, exploratory inquiry, by using the built in features of the bash shell.

Files and Directories Continued



File and Directories: Orienteering

The way to specify a ***path*** to a file can be done in two ways:

- ***Absolute path*** – starts with a “slash”
  - ***/home/fridman***
- ***Relative path*** – does not have a leading “slash”
  - ***fridman/test***
  - Relative paths start in your current working directory.

UNIVERSITY OF CALGARY

Navigating in the file system

- The two most important commands to navigate the Linux file system are
  - ***CD*** – change directory
  - ***PWD*** – print the current directory
- The “cd” command can be used by itself or with an argument:
  - cd – changes directory back to your home directory
  - cd /usr/bin – changes directory to the specified path

Directories- Session 1

Run the following commands and see what the output is:

$ cd /tmp

$ pwd

$ ls -l

$ cd /usr/bin

$ ls –l

$ cd

UNIVERSITY OF CALGARY

Some simple *list* commands - Session 2

- Files
  - `ls`
  - `ls -a`, `ls -l`, `ls -al`
- Processes you have running
  - `ps`
  - `ps -a`, `ps -eaf`
- Users
  - `who`
  - `last`
- What's in a file?
  - `cat`
  - `more`/`less`
  - `grep`
  - `wc`
  - `uniq`
- Running a command you don't like? `CTRL-C`!

File Permissions

What is all of this?

```
$ ls -a

.  ..    .bash_history  .bash_logout  .bash_profile  .bashrc
.config

$ ls -al
total 88
drwx------   3 fridman root   4096 Feb 26 10:34 .
drwxr-xr-x 313 root  root  32768 Feb 25 14:32 ..
-rw-------   1 fridman fridman   237 Feb 26 10:44 .bash_history
-rw-r--r--   1 fridman fridman    18 Dec  2 20:23 .bash_logout
-rw-r--r--   1 fridman fridman   141 Dec  2 20:23 .bash_profile
-rw-r--r--   1 fridman fridman   312 Dec  2 20:23 .bashrc
drwx------   3 fridman fridman  4096 Jan  3 12:10 .config
```

Permissions continued

```
-rw-r--r--   1 fridman fridman   312 Dec  2 20:23 .bashrc
drwx------   3 fridman fridman  4096 Jan  3 12:10 .config
```

The first character describes the entry. A **"-"** means a file, a **"d"** means a directory (more on that later). After that we have 3 sets of three letters:

`r` - read permission
`w` - write permission
`x` - execute permission

A **"-"** in place of one of those letters means no permission

Each set applies to the owner, the group and the world respectively

http://permissions-calculator.org/symbolic/

UNIVERSITY OF CALGARY

Files and Directories

Files are simply that: A collection of bits that can be a document, a picture, some binary code, a script of some kind, source code, this presentation …

A directory is a collection of files and other directories.

They both have certain properties or attributes:

- Permissions
- Size
- Dates
- Name

Files and Directories: Executable Files

What is an executable file?

Remember file permissions? The "`x`" means executable.

But what is that? Just because the "`x`" is there doesn't mean it is executable.

Two types of executables:

- Scripts - the very first two characters are "`#!`" followed by the name of a program that will execute the following text as commands
- Binary executables - compiled programs in machine language ready to run

The command "`file`" will help you identify which is which

```
[fridman@talc ~]$ file /bin/*
```

Files and Directories: Naming

Names are CASE sensitive. **“Bob”** is NOT the same as **“bob”**

Do **NOT** use single characters for names of files or directories: a b c D E f

Try to make them descriptive, but not too descriptive:

- Good: Vacation_2010
- Bad: My Vacation at Uncle Bill’s in the Back Country of Western Wombat Australia 2010 from May 4th to June 5th with Auntie Louise and Cousin Fletcher
- Worse: xxxx

SPACES ARE EVIL in names. NEVER USE THEM. Why? Spaces are delimiters for the command line.

Files and Directories: Odd Things

Special Characters:

```
/'"`{}()[]?\|><~^*&;
```

All of those have special meanings to bash. We'll talk about some of them. NEVER try to use them as part of a file name.

Never use a **"-"** at the beginning of the file name, the **"-"** can confuse commands that use options.

WARNING: If you copy a text file between Windows and Linux, you'll be surprised by the fact that the end of line is different. It breaks lots of things.

File System Operations

Our friend ls:

`ls -al`, `ls -aF`, `ls -al /usr/bin`, `ls -al .`, `ls -al ..`, `ls -al ../..`

`touch`

Now: moving, copying, making directories

- `*`, `?`
- Illustrate what happens when using '`*`' globbing, use echo
- Single and double quotes

Be careful of "`rm *`" in a dir.

BASH quotes

The BASH shell has three types of quotes

1. Single quotes: ‘ ’ The text in quotes does not get modified. The text is treated as literal text.
2. Double quotes: “ ” The text in quotes gets expanded if there are shell variables present in the text ($variable).
3. Back quotes: ` ` The command in the string gets executed and the output of the command is returned.

UNIVERSITY OF CALGARY

BASH variables

- Variables in BASH look different depending on how they are used.
- An assignment just uses the name of the variable
  - filename=‘myprogram.sh’
- The use of the variable requires the $ sign
  - echo “$filename”
  - echo ‘$filename’ does not print the value of the variable but the literal string.
  - echo ${filename}.backup

UNIVERSITY OF CALGARY

```
$ filename=‘myprogram.sh’

$ echo ‘$filename’

$ echo “$filename”

$ echo ${filename}

$ echo ${filename}.backup
```

Linux editors

- Linux has a number of editors. Note, editors are not word processors. Editors are designed for text input and colour coding of programing languages.
- Examples of word editors are:
  - Emacs
  - Vim
  - Nano
- We will work with nano

The nano editor

```
[fridman@talc ~]$ nano test.sh

#!/bin/bash
whoami


[fridman@talc ~]$ chmod u+x test.sh

[fridman@talc ~]$ ./test.sh
```

UNIVERSITY OF CALGARY

Interesting things

More commands.

1. Some simple looping
2. Generate files with slightly different names
3. Generate automatic file names

Repetitive Tasks - loops

```
for VARIABLE in LIST ;  do
        Commands
        Commands
        Commands
        ….
done
```

UNIVERSITY OF CALGARY

```
$ nano newfiles.sh

#!/bin/bash
for i in {1..10}; do
    name="datafile-`date +%Y%M%d%H%m%S`"
    touch $name
    sleep 2
done

$ chmod u+x newfiles.sh
$ ./newfiles.sh
```

UNIVERSITY OF CALGARY

Repetitive Tasks - loops

```
while LIST; do
        Command
        Command
        ….
done
```

```
$ nano make_backups.sh

#!/bin/bash
ls | while read file; do
  cp $file ${file}.BAKUP
done

$ chmod u+x make_backups.sh
$ ./make_backups.sh
```

More Interesting Things!

Practise repetitive tasks

- *Who*
- *Who* |
- Get a count of how many files we have using *wc*
- *Grep* for something in a bunch of files, then summarize result with *count/uniq*

UNIVERSITY OF CALGARY

Putting it together: Scripts

Scripts allow us to:

- Automate complex tasks: *bundle logic into a self contained script*
- Create your own tools/utilities: *data cleaning, integrity checking*
- Customize administrative tasks: *backups, house cleaning, reports*
- Create applications - *reproducibility, package encapsulation*

Repetitive Tasks - conditionals

```
if [[ condition ]]; then
        Command
        Command
        …
fi
```

$nano my_procs.sh

```
#!/bin/bash
for user in `who| cut -f 1 -d ' '`; do
    if [[ $user == 'root' ]]; then
       continue
    fi
    echo ""
    echo "--- $user ---"
    ps -f -u $user
done
```

UNIVERSITY OF CALGARY

$ chmod u+x my_procs.sh

$./my_procs.sh

Who and What is Research Computing

- Research Computing Services (RCS) is a group that plans, manages, and supports high performance computing (HPC) systems used by researchers
- Our focus is to support the increasing demand for scientific computation through a range of specialized services that help researchers solve highly complex real-world problems or run large scale computationally intensive workloads on our high-end HPC resources.

UNIVERSITY OF CALGARY

Questions!

Reach us at : support@hpc.ucalgary.ca