Introduction to Linux Bash

Samuele Germiniani
samuele.germiniani@univr.it

UNIVERSITÀ
di VERONA
Department
of ENGINEERING FOR INNOVATION
MEDICINE

Bourne-Again Shell (Bash)

- Command-line shell and scripting language for Unix-like operating systems
- Provides a text-based interface for interacting with the operating system
- Users can enter commands in the form of text inputs, and Bash interprets these commands and communicates with the Linux kernel

Command syntax

- General syntax of a LINUX command

  command [-options] [argument list]

- Multiple commands can be given on the same line by separating them with ';' (they will be executed in sequence)

  $command_1$; $command_2$; … ; $command_N$

- Every Linux command has its own documentation

  man <command>

File system

File system

- View the contents of a directory

  ls [-options] <path_to_dir>

  Options

  -a also displays hidden files

  -l output in extended format

  -r order alphabetical reverse

  -F appends character to indicate the file type

  (/ for directories , * for executables, @ for links)

  -R also lists files in subdirectories

Path

Single Dot (.) resolves to the present directory

Double Dot (..) resolves to the parent directory of this work directory

Tilde (~) represents the home directory of the logged-in users

Wildcard (*) resolves to any sequence of characters

- Can be used as prefix (ex. *.c), or suffix (ex. fileName*), or both (ex. *name*)

Absolute path

ex. /dir1/dir2/...

Always starts with '/' of the file system. '/' is the directory root

Relative path

ex. dir1/dir2/...

dir1 is in the current directory

Viewing files

- cat file
  - Prints the contents of the supplied file on standard output

- head [-n] files
  - displays the first 10 lines
    -nk print the first *k* lines

- tail [-n] files
  - displays the last 10 lines
    -nk print the last k lines

- less [options] file
  - A text pager for viewing large files one screen at a time

File manipulation

- touch <file_path>
  - The touch command creates a file.
- cp [-r] <file_path> <destination_path>
  - Copy file to destination
  - Use -r to copy directories
- mv <file_path> <destination_path>
  - Move/rename file to dest
- rm [-rf] file/directory ...
  - Delete the provided file/directory. **f** force, **r** recursive

Directories

`cd <path>`

- change the directory to the indicated one
- goes to home (~) directory if now path is given

`pwd`

- show the absolute path of the current directory

`mkdir <dir_path>`

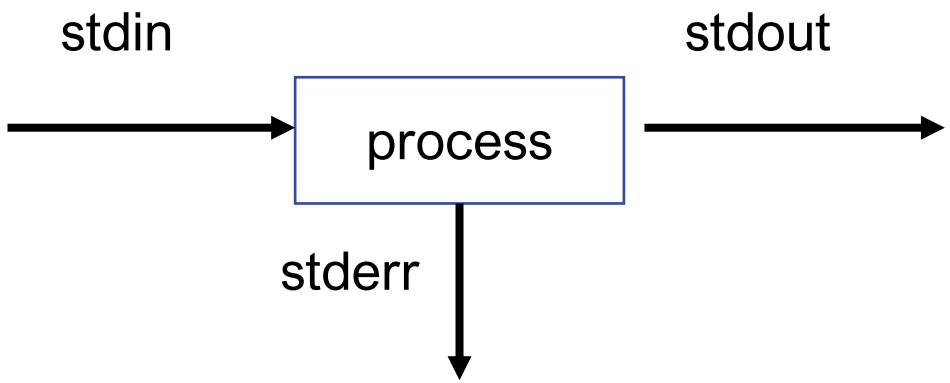
- create a new directory

`rmdir <dir_path>`

- delete the given directory (must be empty)

I/O redirection

- Each process has three associated channels



- Each channel can be redirected
  - on file
  - on another channel

I/O redirection

command < file

stdin read from file

command > file

stdout write to file

command >> file

stdout append to file

command 1> file_out 2> file_error

stdout to file_out and stderr to file_error

command1 | command2

pipe between command1 and command2. stdout of command1 is redirected to stdin for command2

Processes

The state of the processes

- The `ps command` provides a snapshot of current system processes

- Example outputs

```
PID  TTY   TIME     CMD
3490 pts/3 00:00:00 bash
3497 pts/3 00:00:00 ps
```

`PID` Process Identifiers
`TTY` terminal from which the process is run
`TIME` total execution time
`CMD` command executed corresponding

The state of the processes

- Main options
  - `a` : view all processes
  - `x` : all processes of the user
  - `r` : all processes in running state

- Process states
  - **R** running/in running queue
  - **T** stopped
  - **S** interruptible (awaiting an event to complete)
  - **D** uninterruptible (usually I/O)
  - **Z** zombies

Process management

- Processes normally run in **the foreground** and have three standard channels connected to the terminal: stdin, stdout, and stderr

- Processes run with **&** run in **the background** and have no stdin (e.g. ./a.out &)

Stopping & Pausing processed

- Ctrl + c sends SIGINT to the foreground job
  - Terminates the process

- Ctrl+z sends the SIGTSTP signal to the foreground job
  - Stops the process

- Ctrl + d logs out of the interface

Process Management - Commands

- `jobs [-l]`
  lists background or suspended processes.
  Example output: [1]+ Stopped sleep 2

- `bg [%job]`
  resumes specified processes in the background

- `fg [%job]`
  resumes the processes indicated in the foreground

- `kill [-signal] PID`
  sends a signal to the indicated process (most common SIGKILL, SIGTERM)

htop

- htop is a command-line system monitoring utility for Unix-like operating systems, including Linux.

```
  0[||                                                    1.3%]   Tasks: 111, 205 thr; 1 running
  1[|                                                     0.6%]   Load average: 0.00 0.00 0.00
  2[|||                                                   3.8%]   Uptime: 03:46:29
  3[                                                      0.0%]
Mem[||||||||||||||||||||||||||||||||||||||||||||||||||| 460M/3.70G]
Swp[                                                     0K/1024M]

  PID USER      PRI  NI  VIRT   RES   SHR S CPU% MEM%▽  TIME+  Command
 1282 gdm        20   0 3210M  132M 85456 S  0.0  3.5  0:12.64 /usr/bin/gnome-shell
 1285 gdm        20   0 3210M  132M 85456 S  0.0  3.5  0:00.79 /usr/bin/gnome-shell
 1287 gdm        20   0 3210M  132M 85456 S  0.0  3.5  0:00.60 /usr/bin/gnome-shell
 1288 gdm        20   0 3210M  132M 85456 S  0.0  3.5  0:00.01 /usr/bin/gnome-shell
 1290 gdm        20   0 3210M  132M 85456 S  0.0  3.5  0:00.11 /usr/bin/gnome-shell
 1291 gdm        20   0 3210M  132M 85456 S  0.0  3.5  0:00.11 /usr/bin/gnome-shell
 1292 gdm        20   0 3210M  132M 85456 S  0.0  3.5  0:00.18 /usr/bin/gnome-shell
 1293 gdm        20   0 3210M  132M 85456 S  0.0  3.5  0:00.11 /usr/bin/gnome-shell
 1190 root       20   0  262M 73384 46764 S  0.0  1.9  0:01.98 /usr/lib/xorg/Xorg vt1 -displayfd 3 -auth /run/user/127/gdm/X
 1261 root       20   0  262M 73384 46764 S  0.0  1.9  0:00.00 /usr/lib/xorg/Xorg vt1 -displayfd 3 -auth /run/user/127/gdm/X
14159 sam        20   0  563M 37616 30400 S  0.0  1.0  0:00.19 /usr/libexec/goa-daemon
14162 sam        20   0  563M 37616 30400 S  0.0  1.0  0:00.00 /usr/libexec/goa-daemon
14164 sam        20   0  563M 37616 30400 S  0.0  1.0  0:00.02 /usr/libexec/goa-daemon
14165 sam        20   0  563M 37616 30400 S  0.0  1.0  0:00.00 /usr/libexec/goa-daemon
```

How to install: **`sudo apt install htop`**

Environment variables

- Bash has a set of environment variables. Each variable respects the pattern: variable=value

The main environmental variables

- `PWD : current path on the filesystem`
- `SHELL: path to Bash`
- `USER : user username`
- `HOME : path to the user's`

`Add/Edit a variable`

```
export <varName>=<value>
```

Environment variables

How to read the environment variables:

- printenv variable
  - prints the value of the provided environment variable
- env
  - print all environment variables
- echo $variable
  - prints the value of the provided environment variable

Bash editing

Keystroke action	Description
ctrl-A	Move the cursor to the beginning of the line
ctrl-E	Move the cursor to the end of the line
ctrl-W	Erase the preceding word
ctrl-U	Erase from cursor to beginning of line
ctrl-K	Erase from cursor to end of line
ctrl-Y	Paste erased text (for example, from ctrl-U)
ctrl-R	Perform a reverse search for previous commands

Nice to have

Add the following commands to your .inputrc (in the home directory ~)

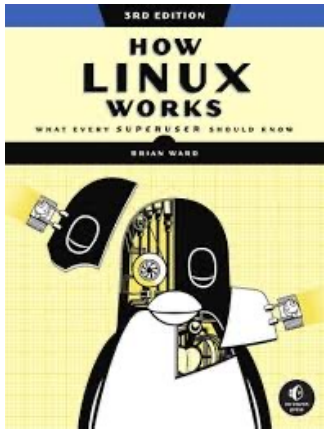
set completion-ignore-case On

Enables case-insensitive tab-completion

set show-all-if-ambiguous

Displays all possible completions when ambiguity arises

Suggested reading



How Linux Works, 3rd Edition
by Brian Ward
Released April 2021
Publisher(s): No Starch Press
ISBN: 9781718500402

Exercises

- Solve the exercises in following repo.
  - Use git to clone the repository

https://github.com/SamueleGerminiani/bash_tutorial