Introduction to Unix Operating Systems

Cedric Arisdakessian

2022-08-15

Day 1: The linux file system

Recommendations

- **Practice!** For simple tasks, try not to use your GUI on your personal laptop:
  - Moving a file/folder
  - Creating/removing a file/folder
  - Searching for files (`find`, `locate`)
- If you have the chance, you should check out the class **ICS-332** (Operating Systems) by Henri Casanova. You will learn a lot about what's happening under the hood.
- Don't be afraid. The worst you can do is delete personal files (but since you have a backup, no problems). By default, the OS won't let you do anything to:
  - Files that are crucial for the system to run
  - Files that you don't own
  - Unless your command starts with `sudo`
- If anything goes wrong, you can stop a command with `ctrl+c`

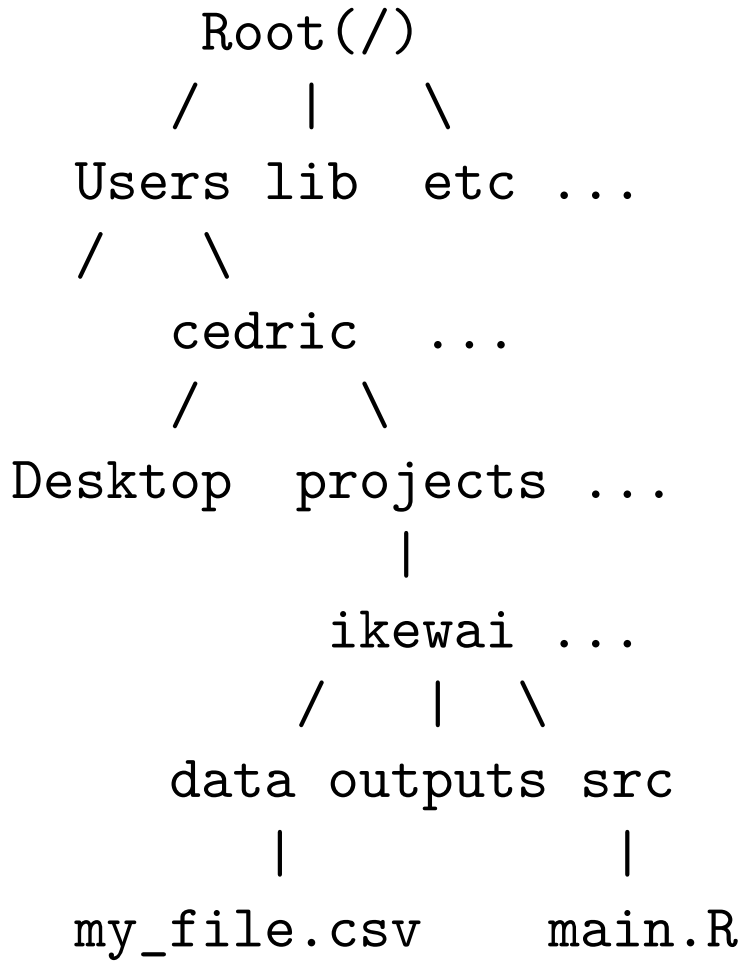
Terminology

- Terminal = Console = Application that let you interact and communicate with your computer
- Shell = Language used for communicating with the computer (bash, zsh, csh, etc.)

We use `bash` for the rest of this tutorial, but most commands would also work with `zsh`

Absolute vs Relative paths

- **Absolute** path: we start from the root ex: `/Users/cedric/projects/ikewai/data/my_file.csv`
- **Relative** path: we start from our current location ex: `../data/my_file.csv` (if my current folder is `src/`)



- Specific symbols
  - `.` refers to the **current** folder
  - `..` refers to the **parent** folder (therefore `../..` is the parent of the parent)
  - `~` refers to the **home** folder
- See https://github.com/labhuiofrank/tutorials/blob/main/pdf/project_guidelines.pdf

The file system

- Different file categories organized in different folders. for example:
  - Your **personal** folder, located in `/Users/{username}` for macOS and `/home/{username}` for linux. It's a safe zone, if you delete or change anything there, you can't break much.
  - System library (.so, .dylib): `/Library` (macOS), `/lib` (linux)
  - Binaries (no extension): `/bin`, `/sbin`, `/usr/bin`
  - Configuration files: `/etc`
  - ...
- Note: Hidden files start with "."

Day 2: Bash syntax

Variables

- **Dollar notation** (with/without braces), **no spaces** around =
- BONUS: Capture the output of a command in a variable with `$()` or backticks
- BONUS: String substitution

```
x = 1 # wrong
x=1 # correct
echo $x ${x} # outputs "1 1"
echo $y # Return nothing (no errors)

output=$(ls .) # preferred
output=`ls .` # works too

name=Cedrik
echo ${name/k/c} # substitute k with c
```

Comparison operators

- **Square brackets** around comparison `[ expression ]`
- **String** comparisons: `==`/`=`, `!=`
- **Numeric** comparisons: `-eq`, `-ne`, `-le`, `-lt`, `-ge`, `-gt`

Conditions

Syntax: - classic: `if [ ... ]; then ...; elif [...] do ...; else ... ; done` - compact: `&&` and `||` notations

```
name=Cedric

if [ "$name" == Cedric ]; then
  echo "Hi ${name}"
else
  echo "Hi stranger"
fi

[ "$name" == cedric ] \
  && echo "Hi ${name}" \
  || echo "Hi stranger"
```

Loops

Syntax: for ...; do ...; done

```
for file in $(ls my_dir); do
  mv $file ${file/.tsv/.csv}
done
```

Streams

- pipe: |
- redirection: >, >>
- input stream: < (and '-')
- stdout and stderr

```
# 1) print rows where second column > 2
# 2) count
awk -F, '$2 > 1' metadata.csv | wc -l

# 1) get the 5th column of my csv file
# 2) sort the values
# 3) Compute the frequencies of consecutive values
# 4) Redirect stdout to file
cut -d, -f5 data.csv | sort | uniq -c > freqs.txt
```

Day 2: Useful linux commands

Basic commands

- `ls`
- `cd` (meaning of ~, ., .., ../..) + `cd` with no argument
- `pwd`
- `mv` (careful)
- `mkdir` (-p) / `touch`
- `rm` (+ `rmdir`)
- `echo`
- `sudo`
- `open` for GUI

Explore unknown commands:

- `man` + / notation for searching
- `-h`, `--help`, or no args

Manipulate files

- `head`/`tail`
- `less`/`more`
- `cat`
- `column`, bonus: `visidata`
- `grep`
- `cut`
- `uniq`
- `wc`

Find files in computer

- `find <folder> -name <pattern> -exec <cmd> \;`

```
find . -name "*.csv" \;
find Desktop -name "*.txt" -exec wc -l {} \;
```

- `locate <filename>` (if initialized)

Remote machines

- hostname
- ping
- ssh:

```
# ssh <username>@<ip or domain>
ssh cedric@142.250.69.196
ssh cedric@kewalo
```

- scp + rsync

```
# upload file to remote location
# scp <file> <username>@<ip or domain>:<path>
scp test.txt cedric@kewalo:~
# download file to my computer
scp cedric@kewalo:~/Documents/data.csv .
```

Checking if file is not corrupted (interrupted transfer)

On remote machine

```
md5sum sample1.fastq > md5sum.txt
cat md5sum.txt
# 46c5daef3cac500540c55a51f0809b14  sample1.fastq
```

On local machine (upload md5sum.txt to machine where the file is)

```
md5sum -c md5sum.txt
# returns: "sample1.fastq: OK"
```

Checking many files

```
md5sum *.fastq > md5sum.txt
```

On local machine (upload md5sum.txt to machine where the file is)

```
md5sum -c md5sum.txt
```

Install new tools

- macOS: `brew`, `port`
- Ubuntu: `apt-get install`
- Other linux distributions: `yum`, `pacman`, ...

More tools:

- `tmux`
- `enhancd`
- `autocompletion`
- `grep`, `egrep`
- `sed`
- `awk`

Day 3: Customization and history search

Customize your environment

- Command aliases
- .bashrc, .bash_profile (.zshrc, .zsh_profile, . . .)
- PATH and LDPATH environment variables
- And more (e.g. .ssh/config)

History search

- CTRL+R
- `history | grep`
- `!` symbol

Terminal editors

- `nano`, `pico`
- `emacs`
- `vi`/`vim`