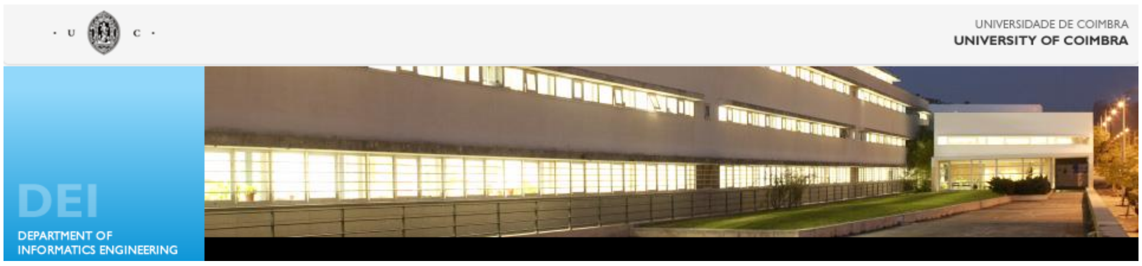

Operating Systems [2020-2021]

Assignment 02 – Introduction to Linux

Introduction

Since its introduction in 1991, Linux popularity and domain of application have been growing. Distributed under an open-source license, and backed by a strong community, it now runs in multiple devices, from mobile phones to supercomputers.

Linux has a number of different ‘flavors’ called distributions that contain a different set of applications suited to fit to different users (e.g., Ubuntu, Debian, Fedora, CentOS). While all distributions are different, they all run with the same Linux kernel.

Objectives

Students concluding this work successfully should be able to:

- Use common commands to operate Linux
- Use filters and pipes
- Change file permissions
- Use basic shell scripting

Support Material

- Linux online manual
- Bash Reference Manual (https://www.gnu.org/software/bash/manual/bash.pdf)

Basic shell commands

Getting help

• **man** [section] *keyword*	# online manual
	# 'b': previous page; 'space': next page; 'q' quit
	# '/' to search within the man pages
• **apropos** *keyword*	# search the manual page names and descriptions
	# to find the right man page

Working with directories and files

• **pwd**	# print working directory
• **ls** [-a] [*wildcard*]	# list directory contents
• **cd** *directory*	# change directory
• **mkdir** *directory*	# make a new directory
• **rmdir** *directory*	# remove a directory
• **cp** *file1 file2*	# copy file(s)
• **mv** *file1 file2*	# move file(s)
• **rm** *file*	# remove file(s)
• **ln** {-s} *file linkname*	# make a symbolic link to a file/directory
• **cat** *file*	# concatenate and print files to *stdout*
• **grep** [-n] *"string" file*	# print lines from a file that match a pattern
• **more** *file*	# print files to *stdout* page by page
• **diff** *file1 file2*	# compare two files
• **sort** *file*	# sort a file
• **tail** *file*	# shows the end of a file
• **cut** {-c *list*}{-f *list* -d }	# cut parts from a file
• **wc** {-lwc} *file*	# line, word, character count
• **file** *file*	# determine the file type
• **head** *file*	# shows the start of a file
• **chmod** {ugoa}{+-}{rwx} *file*	# change the permissions mode of a file

Working with processes

• **ps** [-a]	# process status
• **kill** -{SIGNAL} *pid*	# sends a signal to a process
	# (*SIGNAL*=-9 terminates a process)

Working with users

• **who** or **w**	# display who is logged in the system
• **whoami**	# who am I
• **passwd**	# change user password

Editors

• **nano** *file*	# file editor
• **vi** *file*	# file editor
• **emacs** *file*	# file editor

Other

date	# print date & time
cal *month year*	# calendar
find *dir* {-name *file* [-print]}	# find a *file* (or files) starting at directory *dir*
sleep *seconds*	# suspend execution for n seconds
lpr -P*printer file*	# send a job to the printer
echo *'string'*	# echo arguments to the standard output
clear	# clear the terminal screen
csh *script*	# C-shell command interpreter
uname	# print system information
du	# estimate file space usage
df	# report file system disk space usage

Jobs

Ctrl-c	# cancel a foreground process
Ctrl-z	# suspend a foreground process
command **&**	# put a process in background
jobs	# list the background processes
fg	# resume to foreground a stopped process
fg *%job_number*	# bring a background job into the foreground
bg	# put in background a stopped process

Redirection (> output < input)

command **>** *file*	# redirect *stdout*
command **>>** *file*	# redirect *stdout* but append
command **>&** *file*	# redirect *stdout* and *stderr*
(*command* > *file1*) > &*file2*	# redirect *stdout* to file1 and *stderr* to file2
command **<** *file*	# redirect *stdin*
command1 **|** *command2*	# pipe between two commands

Exercises

Note: Only some of the exercises provided in this assignment will be done during the practical classes. The extra exercises should be done by the student as homework and any questions about them should be clarified with the teacher.

1. Linux commands

a. Open a terminal window
b. Go to your to your “Desktop” directory
c. Create a new directory named “A2”
d. Change the name of the directory from “A2” to “Assignment”
e. Enter the new directory (use autocomplete if possible)
f. Execute the following commands, by the order indicated, in the Linux prompt. Analyze the results.
   - `date|cut -c4-10`
   - `who|grep tty|wc -l > file1.txt`
   - `cat file1.txt`
   - `ls -la |tee file2.txt`
   - `cat file2.txt`
   - `echo “End of file”>>file2.txt`
   - `cat file2.txt`

g. Find all processes running in your machine and owned by your user. Put the result in “file3”.
h. Change the permission in “file3” to protect it from being written. After changing the permission try to change its contents.
i. Create a directory named “txt” under “Assignment”
j. Copy all files with extension “.txt” to the new directory
k. Remove the directory “txt” and all its files
l. Find all the files inside directory “/usr/src” and its subdirectories that have “.c” extension
m. Locate command *gcc* using command *which*
n. Go to your user directory
o. Using the editor ‘nano’ create a file named ‘test’ with the following lines:

```
ls
echo ---
date
echo ---
whoami
```

p. Save the file and exit nano
q. Execute the file ‘test’. If it does not run, check PATH and permissions.
r. Remove “Assignment” directory
s. Check your PATH by displaying its value.
t. Temporarily change your terminal prompt. It should look like: “MyMachine>”

2. Jobs

Execute the following code and analyze the results.

```
$ sleep 5; ls
$ sleep 5; ls&
$ (sleep 40;ls)&
$ jobs
$ sleep 160
^Z
$ jobs
$ fg
^Z
$ jobs
$ bg %1
$ ps -l
$ kill %1
```

Note: ^Z = Press keys “Ctrl” + “Z”

3. Shell programming

3.1 - Create a bash shell script named *create_header.sh* that receives by parameter the name of a file, the name of a project and your name, and automatically creates a file with the name provided with the extension “.c” containing in the header your name and the name of the project.

E.g.

```
$ ./create_header.sh file1 MyFirstCProject John
```

```
// John
// Project: MyFirstCProject
// ----------------------------
```

File *file1.c*

3.2 – Create a script named *mem_free.sh* to monitor the memory in your computer. Once executed, the script should write to the screen and append to a specific log file (`mem_free.log`), each 5sec, the system free memory together with the date and hour of data collection. Free memory can be collected in file `‘/proc/meminfo’`.

E.g.:

```
MemFree: 46900 kB : 14/09/2017 16:06:30
MemFree: 46000 kB : 14/09/2017 16:06:35
```

Hint

Analyze the following command:

```
$ echo `grep MemFree /proc/meminfo` : `date +%d/%m/%Y" "%H:%M:%S`
```

3.3 – Create a script named *list_dirs.sh* that lists all the directories in your current directory, ordered by name and showing their number.

E.g.:

```
$ ./list_dirs.sh
Directory 1: .
Directory 2: ..
Directory 3: SO
Directory 4: IRC
Total directories = 4
```