



IMT Atlantique
Bretagne-Pays de la Loire
École Mines-Télécom

Integration week – TAF DCL

Integration week – Shell

Résumé

You can complete this lab session on any machine running Linux. You can use your own machine if you wish. Some questions require administrator access to your machine (to use the `apt install` command), which is not available on a school computer. Go at your own pace. Use the `man`, `help` commands when you have questions.

1 Création de répertoire et de fichier / Directory and file creation

1. Navigate to your home directory.

```
cd ~
```

2. Create the following folder hierarchy inside your home directory (you may use a single `mkdir` command) :

`~/Europe/France/Bretagne/Finistere`

```
mkdir -p ~/Europe/France/Bretagne/Finistere
```

3. Create a file `ville.txt` inside the `Finistere` directory, while remaining in your home directory `~`, i.e. without using `cd` to enter `Finistere` directory (use the `touch` command) :

```
touch ~/Europe/France/Bretagne/Finistere/ville.txt
```

2 Renommage, copie, suppression d'un fichier / Renaming, copying, and deleting files

1. Copy the `ville.txt` file to `Brest.txt` (use the `cp` command) :

```
cp ~/Europe/France/Bretagne/Finistere/ville.txt ~/Europe/France/Bretagne/Finistere/Brest.txt
```

2. Copy the **ville.txt** file to **Le Conquet.txt** (use the **cp** command and quotes where necessary) :

```
cp ~/Europe/France/Bretagne/Finistere/ville.txt "~/Europe/France/Bretagne/Finistere/Le Conquet.txt"
```

3. Copy the folder **Finistere** and name the copy **Morbihan** (use the **cp -r** command) :

```
cp -r ~/Europe/France/Bretagne/Finistere ~/Europe/France/Bretagne/Morbihan
```

4. Inside the **~/Europe/France/Bretagne/Morbihan/** folder, rename the file **Brest.txt** to **Quimper.txt** (use the **mv** command) :

```
mv ~/Europe/France/Bretagne/Finistere/ville.txt ~/Europe/France/Bretagne/Finistere/Quimper.txt
```

5. Remove the file **ville.txt** from the **~/Europe/France/Bretagne/Morbihan/** folder (use the **rm** command) :

```
rm "~/Europe/France/Bretagne/Finistere/Ville du Ponant.txt"
```

6. Navigate to the **Bretagne** folder, and remove the **Morbihan** folder located inside it (use the **rm -rf** command ; see the meaning of these options using **man rm**) :

```
rm ~/Europe/France/Bretagne/Finistere/ville.txt
```

3 Navigation dans le système de fichiers / File system navigation

1. Navigate to the `~/Europe/` directory (use the `cd` command).

```
cd ~/Europe
```

2. Move the `~/Europe/France/Bretagne` directory into the `~/Europe` directory.

```
mv France/Bretagne .
```

3. Navigate to the `~/Europe/France/` directory.

```
cd France
```

4. Move the `~/Europe/Bretagne` directory back into the `~/Europe/France` directory :

```
mv ../Bretagne .
```

5. Print your current working directory (use the `pwd` command) :

```
pwd
```

6. Navigate into the `~/Europe` directory using an **absolute path** (the absolute path starts at the root directory `/`) :

```
Sur un système Linux standard : cd /home/<yourlogin>  
À l'école : cd /homes/yourlogin  
Sur un MacOSX : cd /Users/<yourlogin>
```

7. Inside the **Europe** directory, create the directories **Germany**, **Italy** and **Spain** (all with a single `mkdir` command) :

```
mkdir Germany Italy Spain
```

8. Inside the **Europe** folder, navigate into its subfolder whose name contains exactly 6 letters (use the wildcard symbol `"?"`) :

```
cd ??????
```

You should have found yourself in the **France** folder. Why? (Only the France directory contains six letters.)

9. From the **France** folder, navigate one level above and then try to navigate into a directory whose name contains exactly 5 letters (use the wildcard symbol `"?"`) :

```
cd ../?????
```

This command failed, with the error : **bash: cd: too many arguments**

Why? (More than one directory has a name containing only five letters : Italy and Spain.)

10. Still from the **France** folder, navigate one level above and then try to navigate into a directory whose name contains exactly 5 letters and ends with the letter **n** (use the wildcard symbol `"?"`) :

```
cd ../????n
```

You should have found yourself in the **Spain** folder. Why? (Even though two directories

have names containing 5 letters (Spain, Italy), only one of them ends with the letter "n".)

11. From the **Spain** folder, navigate one level above and then try to navigate into a directory whose name ends with the letter **y** (use the wildcard symbol "*****") :

```
cd ../y
```

This command also failed, with the same error : **bash: cd: too many arguments**

Why? (More than one directory has a name containing ending in letter "y" : Germany and Italy.)

12. Still from the **Spain** folder, navigate one level above and then try to navigate into a directory whose name ends with the letter **n** :

```
cd ../n
```

You should have found yourself in the **Spain** folder. Why? (More than one directory has a name containing ending in letter "y" : Germany and Italy.)

4 Recherche de fichier / File search

1. Inside the **~/Europe/** directory, find the file whose name starts with **Brest** (use the **find** command ; see the manual for appropriate options).

Use the following syntax : **find <location> -type f -name <filename>**.

Example : **find ~/ -type f -name myFile.txt**

```
find ~/Europe -type f -name "Brest"
```

2. Inside the **~/Europe/** directory, find the file whose name starts with **Brest** and which has a size bigger than 1 megabyte (use the **find** command) :

```
find ~/Europe -type f -name "Brest" -size +1M
```

You should have obtained an empty result. Why?

(Existing files with a matching name have a file size less than 1 megabyte)

3. Inside the **~/Europe/** directory, find the directory with the name **Finistere** (use the **find** command) :

```
find ~/Europe -type d -name Finistere
```

If the command returns some "Permission denied" errors, remove the error messages by adding

-print 2>/dev/null to the end of your command.

```
find ~/Europe -type d -name "Finistere" -print 2>/dev/null
```

4. Suppose we want to navigate directly to the result of a **find** command. If your previous command found a single folder named **Finistere**, you can use the path resulting from this command in another command. Use the following syntax to do it : **cd \$(YourFindCommandHere)**. The **\$()** syntax forces the shell to first interpret the content between the parentheses.

Tip : You can use the **↑** arrow on your keyboard to retype your previous command.

```
cd $(find ~/Europe -type d -name Finistere)
```

5. Locate the **Finistere** folder using the **locate** command :

```
locate Finistere
```

6. Locate the **Finistere** folder using the **locate** command, specifying that it should be located inside the **~/Europe** folder (use wildcards) :

```
locate ~/Europe/*Finistere
```

5 Lecture et edition d'un fichier / Reading and editing file contents

1. Navigate into the **Finistere** directory. Edit the content of **Brest.txt**, by replacing its content with the phrase "**Brest is a beautiful city.** ". Use the command **echo** and the **>** redirection symbol :

```
cd ~/Europe/France/Bretagne/Finistere; echo "Brest is a beautiful city. " > Brest.txt
```

2. Read the content of the file **Brest.txt** using the **more** command :

```
more Brest.txt
```

3. Replace the content of the file **Brest.txt** with the phrase "**Brest is a modern city.** ". Use the command **echo** and the **>** redirection symbol :

```
echo "Brest is a modern city. " > Brest.txt
```

4. Read again the content of the file **Brest.txt**, this time using the **less** command (press **Q** key afterwards to return to the terminal) :

```
less Brest.txt
```

5. Add to the content of the file **Brest.txt** the phrase "**It is located in Finistere.** ". Use the command **echo** and the **»** redirection and concatenation symbol :

```
echo "It is located in Finistere. " » Brest.txt
```

6. Use a text editor (nano, gedit, vim, etc.) to modify the content of the file **Brest.txt**. Add the phrase "**Brest lies on the atlantic coast.** ", then save and quit the text editor to return back to the terminal. Use the following syntax to open a file with your text editor (**\$<TextEditorName> <FileName>**) :

```
nano Brest.txt
```

7. Read the content of the file **Brest.txt** using the **cat** command :

```
cat Brest.txt
```

6 Archivage / Archiving and Compression

1. Navigate to the **~/Europe/** directory. Create a (non-compressed) archive file containing the directories **France** and **Germany** using the **tar** command, and name it **Europe.tar** (see

the manual for appropriate options) :

```
tar -cvf Europe.tar France Germany
```

Tar files are not compressed! They occupy the same space on your filesystem as their contents.

2. List the contents of the **Europe.tar** archive file that we just created (use the **tar** command; see the manual for appropriate options) :

```
tar -tf Europe.tar
```

3. Add the **Italy** and **Spain** directories to your archive.

```
tar -rvf Europe.tar Italy Spain
```

4. Navigate to the **~/Europe/** directory. This time create a **compressed** archive file containing the directories **France** and **Germany** using the **tar -czvf** command, and name it **FraGer.tar.gz** (see the manual for appropriate options) :

```
tar -czvf FraGer.tar.gz France Germany
```

Use the **ls** command to verify that the archive was created. As the name suggests, a compressed archive usually occupies at most the same size as its contents.

5. List the contents of the **FraGer.tar.gz** archive file that we just created (use the **tar -tf** command; see the manual for appropriate options) :

```
tar -tf FraGer.tar.gz
```

6. Create a directory called **tmp** and extract the contents of the **FraGer.tar.gz** archive into it (use the **tar -xvf** command; see the manual for appropriate options) :

```
mkdir tmp; tar -xvf FraGer.tar.gz -C ./tmp; ls -l ./tmp
```

Check that the tmp directory really contains the contents extracted from the archive.

7 Gestion des droits sur un fichier / File permissions

1. See the file permissions for the **Europe.tar** archive that you created (use the **ls -l** command).

```
ls -l Europe.tar
```

2. Change the permissions to the **Europe.tar** archive allowing only the file author to read and write it (use the **chmod** command; use the octal mode) :

```
chmod 600 Europe.tar
```

Verify if you succeeded using the **ls** command.

3. Change the permissions to the **Europe.tar** archive allowing no permissions to anyone (use the **chmod** command) :

```
chmod 000 Europe.tar
```

4. List the contents of the **Europe.tar** archive using **tar -tvf** command.

```
tar -tvf Europe.tar
```

What do you observe? Is it normal?

The command fails with the error "**tar: Europe.tar: Cannot open: Permission denied**" because we have no reading permission for this file.

5. Inside the **Europe** directory, create a new file called **newFile** (use the **touch** command). See what permissions are set for this file using the **ls -l** command.

```
cd /Europe; touch newFile; ls -l newFile;
```

Why are these specific permissions set for this file? These default permissions were defined by the **umask** command.

6. See the list of default file permissions (use the **umask -S** command) :

```
umask -S
```

8 Remote connection to a computer / Connexion à un autre ordinateur via le réseau

1. Identify the name of a neighbouring computer (use the **uname** command) :

```
uname
```

2. Use the **ssh** command to connect remotely to that computer using your username :

```
ssh username@computername
```

Verify that you are connected to that computer using the **uname** command.

3. Log-out from that remote computer using the **exit** command. Verify that you are back on your computer.

```
exit
```

4. Connect again to the remote machine, this time using the **-X** option, that will allow you to retrieve the graphic output of applications that you launch.

```
ssh -X username@computername:location
```

5. Launch the **xclock** command on the remote machine, and see if you retrieve the output on your computer.

```
xclock
```

Once you succeed, log out from the remote computer using the **exit** command.

6. You may have noticed that you have to enter your password every time you connect to a remote computer. You can automatize this step by using a pair of private/public keys. Generate a pair of keys on your computer using the **ssh-keygen** command :

```
ssh-keygen
```

Check the content of the **/.ssh** directory to see what was generated.

7. Copy your public key to the remote computer using the **ssh-copy-id** command :

```
ssh-copy-id -i ~/.ssh/myKey username@computer
```

Reconnect to the remote computer to see if you are still being asked for a password, then disconnect.

- Copy a file to or from a remote computer using the **scp** command, using the following syntax : **scp <copyFrom> <copyTo>**. A remote location can be accessed using the syntax : **username@computerName:pathToFile**. For a location on the local machine, simply indicate the path to file.

```
touch testFile; scp testFile username@computer:~/
```

9 Installation de nouveaux logiciels et des librairies / Installation of software and libraries

- If you have administrator rights on your computer, you can use the **apt install** command to install new software. Try to install the Nano text editor (you may need administrator rights for this; use **sudo** command for that) :

```
sudo apt install nano
```

- To install a Python library, you can use the **pip** package manager. Try install the **numpy** library :

```
pip install numpy
```

- To see which executable of Python you are using, use the command **which** :

```
which python
```

10 Environment

- Identify the name of the shell that you are using (use the **echo** command) :

```
echo $SHELL
```

This will output the content of the **\$SHELL** variable of your environment.

- Display the variables present inside your shell environment (use the **env** command) :

```
env
```

- Using the **echo** command, display the content of the following variables : **\$TERM**, **\$MAIL**, **\$PATH**, **\$LOGNAME**, **\$SHELL**.

```
echo $TERM; echo $MAIL; echo $PATH; echo $LOGNAME; echo $SHELL;
```

- Define a new variable called **VAR1**, and set its content to **abc** (use the **export** command) :

```
export VAR1=abc
```

- Check if your variable was added to the environment using the **env** command (you can also use **grep** command to filter the result of **env**) :

```
env | grep VAR1
```

6. Open a new terminal window, and check if your variable exists in the environment of that terminal as well.

```
env | grep VAR1
```

Normally, the **VAR1** variable is not present in this new terminal. Why?

7. Display the tree of processes by using the **ps tree -h** command. Understand the implication of this tree structure : the variables declared on a branch will not be available on neighbouring branches.

```
ps tree -h
```

8. To declare a variable in every new terminal that you open, you can add the declaration of that variable to your `~/.bashrc` file, by adding the following line to its content : **export VAR1=<value>**. The `.bashrc` file is executed before each bash shell that you open, so a variable declared in this file will be available in that shell.

```
echo "VAR1=abc" >> ~/.bashrc
```

11 Processes

1. Use the **ps** command) to print the list of processes being executed on your computer (see the manual for identifying appropriate options) :

```
ps -aux
```

Notice that each process has a process identifier (a **PID**).

2. Launch an application such as **xclock** in background mode (use **&** symbol) :

```
xclock &
```

3. Identify the PID of this **xclock** process (use **ps** and filter the result using **grep**) :

```
ps -aux | grep xclock
```

4. Stop this process using the command **kill** and the PID of the **xclock** process :

```
kill -9 <pid>
```

5. You can see which processes consume most of your processing capacity using the command **top** :

```
top
```

6. You can send a process to execute in the background using the **CTRL+Z** key combination. You can bring it back to foreground using the **fg** command. Try it using the **xclock** command.

```
fg
```