



# Introduzione a Linux

Lezione 10  
Esercizi riassuntivi

**Ruggero Donida Labati e Angelo Genovese**

**Laboratorio di Sistemi Operativi**

Università degli Studi di Milano  
Dipartimento di Informatica  
A.A. 2022/2023

RUGGERO DONIDA LABATI E ANGELO GENOVESE – INTRODUZIONE A LINUX – LEZIONE 10 – ESERCIZI RIASSUNTIVI

1

## Sommario

### 1. Esercitazioni online

- Linux introduction puzzle
- Linux scavenger hunt



RUGGERO DONIDA LABATI E ANGELO GENOVESE – INTRODUZIONE A LINUX – LEZIONE 10 – ESERCIZI RIASSUNTIVI

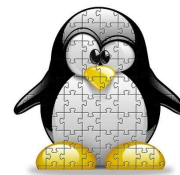
2

## 1. Esercitazioni online

- University of North Carolina  
Linux introduction puzzle/quiz/treasure hunt  
<http://its2.unc.edu/divisions/rc/training/scientific/>
- Chris Retford  
pushingice  
Scavenger hunt  
<https://github.com/pushingice/scavenger-hunt>

## Linux introduction puzzle (1/10)

- Linux Introduction Puzzle/Quiz/Treasure Hunt
  - [http://its2.unc.edu/divisions/rc/training/scientific/Introduction\\_to\\_Linux/linux.intro.puzzle.pdf](http://its2.unc.edu/divisions/rc/training/scientific/Introduction_to_Linux/linux.intro.puzzle.pdf)
  - Data file:
    - [https://homes.di.unimi.it/genovese/esercizi\\_linux/ncfmp.tgz](https://homes.di.unimi.it/genovese/esercizi_linux/ncfmp.tgz)



## Linux introduction puzzle (2/10)

- Get the file `ncfmp.tgz` (*data file*) and `untar` and `unzip` it in your directory space
- Under the top `ncfmp` directory, create a directory called `answers`. You will be copying files to this directory throughout this exercise.

## Linux introduction puzzle (3/10)

- For all the Linux commands discussed in this class there is a very useful command you can use to get extensive help on using any given command. We have emphasized using this command during this lecture. What is the name of this command? Find a directory in the `ncfmp` directory with the same name as this command. Go to that directory and copy the file you find there to your `ncfmp/answers` directory

**Linux introduction puzzle (4/10)**

- What file(s) do you find in the *ncfmp/HBL/runs* directory?
- Which one has been modified most recently? Go into that directory.
- One of the files there is *\*not\** a directory. Copy that file to your *ncfmp/answers* directory.

**Linux introduction puzzle (5/10)**

- What is the biggest file in the *ncfmp/dem* directory? Copy just the last line of that file to your answers directory into a file you create called *quiz.ans*. (Hint: you can do this on a single command line using the Linux command *tail* and a redirect).
- How do you list files such that file types are distinguished by the font color? (Hint: this is an option to a particular command).

## Linux introduction puzzle (6/10)

- Find the file under the *ncfmp* directory which has the same name as the option above. Move this file to your answers directory.
- Go to the *ncfmp/ww3/ftn* directory. List all the files that end with the *.f* extension. (Hint: see using wild cards). Now copy (just) these files to the answers directory.

## Linux introduction puzzle (7/10)

- Go to the *ncfmp/ww3* directory. To see all the files under this directory you could issue the command:
  - `find . -name "*"`
- If you wanted to count the files, you could pipe this output into the word count program, *wc*, like so:
  - `find . -name "*" | wc -l`

## Linux introduction puzzle (8/10)

- How many files do you count? Now remove the test directory and all of its contents (you can do this with one command). How many files are left?
- Find a file with this number under the top *ncfmp* directory and copy it to your answers directory. (Note: the file name is spelled out, for example 7 would be seven).

## Linux introduction puzzle (9/10)

- Finally, to put all this together and to see the answer to this puzzle, go to your answers directory. You should have 12 files there. To arrange these in the proper order and to get a nicely formatted output we will use a little Linux magic (in the tradition of Arthur C. Clarke, magic is defined as something more advanced than the level of this course :). Issue the following command and read the output:
  - `cat * | sort -n | tr -d '\n' | tr -d '[:digit:]' | tr '%' '\n' | tr -s ''`

## Linux introduction puzzle (10/10)

- This simply concatenates all the files together (*cat*) then sorts (*sort*) them using a numeric key that I put in the text, and then does a series of character translations (*tr*) to make the output pretty by 1) removing new lines between the files, 2) removing the numbers used for sorting, 3) inserting new lines wherever I put a '%' character and finally 4) squeezing any extra spaces down to a single space.

## Scavenger hunt (1/33)

- Linux scavenger hunt
  - <https://github.com/pushingice/scavenger-hunt>
  - Readme
    - <https://github.com/pushingice/scavenger-hunt/blob/master/README.md>



## Scavenger hunt (2/33)

- If you are using a new Linux install or Live CD, you may need to install Git first (sudo apt-get install git on Ubuntu). Open a terminal and type:
  - git clone <https://github.com/pushingice/scavenger-hunt.git>
  - cd scavenger-hunt
- First, choose a secret number with at least 4 digits to share with your team, or keep to yourself if you are working alone. Don't forget it! The secret number makes your clues unique, so other teams can't look over your shoulder. Then type:
  - python generate\_clues.py [secret number]

## Scavenger hunt (3/33)

- Any time we enclose something in square brackets, you need to replace it with an actual value (called an argument). For example, to get started I might type:
  - python generate\_clues.py 42
- This will create a subdirectory called clues. Be sure to keep this file (called the README) open in a separate viewer.

## Scavenger hunt (4/33)

### CLUE 1: THE HUNT BEGINS

- man
  - The first command we are going to learn is man, which is short for manual. Typing man [command] will give you a help page (usually called a manpage) for most commands.
- ls
  - The next command we need to learn is ls (list). Type man ls and read the description. Press q to exit. Then type ls and you should see something like this:
    - APPENDIX.md clues generate\_clues.py LICENSE.md next\_clue.py README.md

## Scavenger hunt (5/33)

- If you ever get lost, just do
  - `cd ~/scavenger-hunt`
- If you `cd` to the clues directory and do an `ls`, you will notice that there are a lot of clue directories. Most of them contain fake clues. Throughout our hunt we will be looking for real clues. Using `cd`, navigate to `clues/12345` and type `ls`. You should see a single file named clue.



## Scavenger hunt (6/33)

- Finally we need to be able to look at our clues. First read the manpage for cat, then do:
  - `cat clue`
- This should list the clue in your terminal. From now on, everything we need will be contained in these clue files. It's a good idea to keep track of all the clue folders (like 12345) on a piece of paper. You can also do things like copy all the clue files to your home folder, or cut and paste the clue text into another file.

## Scavenger hunt (7/33)

### CLUE 2: THE LAY OF THE LAND

- What if we get lost and need to know where we are? Just type `pwd` (print working directory). This should print something like this:
  - `/home/user/scavenger-hunt/clues/12345`
- We are five folders deep, in a folder named `12345`.

## Scavenger hunt (8/33)

- To find the next clue, go to the */usr* directory and count the number of subdirectories. This is a hint to your next clue location.
- Go to the *scavenger-hunt* directory, and type
  - `python next_clue.py [secret number] [next clue number] [hint]`
- So, if there were 9 directories, we would type
  - `python next_clue.py 42 3 9`

## Scavenger hunt (9/33)

- If you get the hint wrong, an incorrect clue will be printed
- After you navigate to clue 3, do:
  - `less clue`



## Scavenger hunt (10/33)

### CLUE 3: HUMANS VS. MACHINES

- Your hint for clue 4 is the file `/etc/hostname`. This file contains a single word, which is the name of your computer. This name is your hint. Remember we can find the next hint by typing:
  - `python next_clue.py [secret number] [next clue number] [hint]`



## Scavenger hunt (11/33)

### CLUE 4: MOVING DAY

- We've been exploring the directories that already exist on the computer. But what if we want to make our own folders and files? The first thing we need to do is create a new directory.
  - `cd ~/scavenger-hunt`
  - `mkdir saved-clues`

## Scavenger hunt (12/33)

- What we're going to do is save off all the clues we find in a separate folder that we created with *mkdir* (make directory). Since the README is clue 1 we don't need to worry about it. If you've been writing down all the clue locations, this next part should be easy.
- Let's copy all of the clues we've found so far to our saved-clues folder:
  - `cp clues/02502/clue saved-clues/clue2`
  - `cp clues/12345/clue saved-clues/clue3`

## Scavenger hunt (13/33)

- Linux commands often have options that change how they behave. For instance, compare `ls -l` to ordinary `ls`. Here the `-l` is an option. You can group options together like this:
  - `ls -lahS`
- Now let's say we don't like the folder name saved-clues. We can just move (`mv`) it:
  - `mv saved-clues [pick a new name]`

## Scavenger hunt (14/33)

- Now do an `ls` to see the results of the move. Be careful with `mv`: you can easily overwrite an existing folder.
- Read the man page for `mv` and find an option to prevent overwriting. That option is your next hint.



## Scavenger hunt (15/33)

### CLUE 5: IS THERE AN ECHO IN HERE?

- Sometimes we want the computer to repeat back the results of some command. Try:
  - `echo hello`
- The most basic thing `echo` will do is repeat back whatever you type.

## Scavenger hunt (16/33)

- You can use this to create a small file, or start a new file:
  - `echo My bologna has a first name > oscar.txt`
- If you look in *oscar.txt* you will see exactly what you typed. The '`>`' symbol used here is called a redirect. It redirects whatever would normally be printed to the screen to a file. You can try it with other commands:
  - `ls > my_directory.txt`

## Scavenger hunt (17/33)

- You can also use `echo` to display what are called environment variables
  - `echo $PATH`
  - `echo $HOME`
- The `HOME` variable should make sense at this point.
- The `PATH` variable tells the computer where programs are. Each path that could contain a program is placed between colons. Your hint for the next clue is the first path listed in your `PATH`.



## Scavenger hunt (18/33)

### CLUE 6: WHICH 'WHICH' IS WHICH?

- In the previous clue we learned about *PATH*. This tells the computer where to find binary programs we can run. But, how do we find where a specific program is? The answer is which:
  - `which mv`
- This should print `/bin/mv`. This tells us that the `mv` command is installed in the `/bin/` directory. which itself is a program so you could try:
  - `which which`

## Scavenger hunt (19/33)

- *python* is a program we have been using to generate and test our clues. Your next hint is the location of the *python* program.



## Scavenger hunt (20/33)

### CLUE 7: MAKE ME A SANDWICH

- Linux has the concept of a root user, which is similar to the administrator user on Windows. This user is sometimes called the super user. If you want to do something as the super user, but stay logged in as yourself, there is a command for that: *sudo*. It stands for “super-user do”.

## Scavenger hunt (21/33)

- Sometimes you need a new program. To install software on some versions of Linux (Ubuntu and Debian), you use the command *apt-get*. On other versions (Fedora, CentOS) you use the command *yum*. Let's install a text editing program called *vim*.
  - `sudo apt-get install vim`
- Now we have the ability to edit files. Try:
  - `vim README.md`
- from the scavenger-hunt directory. Some of the commands for vim are a little strange. For now, just type `:q!` to quit.

## Scavenger hunt (22/33)

- The hint is the name of the first folder listed in `/sys/kernel/debug`.
- Depending on the system you are using, you may not have permission to use `sudo`. In this case you can use the hint *denied*.



## Scavenger hunt (23/33)

### CLUE 8: COUNTING WORDS

- Word count (`wc`) is a useful program. You can use it to tell how many lines, words, and/or characters are in a file:
  - `wc README.md`
- This will print the number of lines, words, and characters, in that order. If you just want one of those, you can use `-l`, `-w`, or `-c`.

## Scavenger hunt (24/33)

- Check to see if you have the file `/usr/share/dict/words` installed. If not, run this:
  - `cat /usr/share/dict/words`
  - `sudo apt-get install ispell`
- Now there is a file that acts as a dictionary for spell-checking: `/usr/share/dict/words`. Your next clue is the number of words in the dictionary.

## Scavenger hunt (25/33)

### CLUE 9: SEARCHING HIGH AND LOW

- Searching files is another useful trick. Try this:
  - `grep secret README.md`
- This will print out every line that contains the word "secret".

## Scavenger hunt (26/33)

- Regular expressions can be more powerful and/or complicated. For example:
  - `grep m.n README.md`
- will find any line where the letters *m* and *n* exist with a single character between them. Check the man page for many interesting features of *grep*.

## Scavenger hunt (27/33)

- The next hint is the word that occurs after “tactful” in */usr/share/dict/words*. There's a specific option for *grep* that will make this easy.



## Scavenger hunt (28/33)

### CLUE 10: PIPES

- Many commands will print their output. This is called “standard output” or stdout. We saw earlier how you can redirect standard output to a file (`>`). There is also standard input (`<`). For example, `cat < README.md` is the same as `cat README.md`. But standard input and output can be chained together using pipe (`|`). For example, you can count the number of files and folders in a directory like this:
  - `ls | wc -w`

## Scavenger hunt (29/33)

- This works by taking the output of `ls` and using it as the input to `wc`. Another example:
  - `grep ^sand /usr/share/dict/words | wc -l`
- will print the number of words that start with “sand”. The carat `^` symbol is a regular expression that means “starts with”. You can also use `$` for “ends with”.

## Scavenger hunt (30/33)

- Sometimes you need to sort data alphabetically. You'll notice that the dictionary file is already sorted. You can create your own unsorted copy like this:
  - `sort -R /usr/share/dict/words > random_words`
- Now you can sort *random\_words* to get back to alphabetical order, or `sort -r random_words` for reverse alphabetical order.

## Scavenger hunt (31/33)

- Use the command `ls -la /usr` to get a big list of files. The 5th column in that list is the size of the file in bytes. Find the sort command to print the list of files with the largest file first, and then the rest in descending order. Your hint is the options you had to use. You'll need to use double quotes for your hint. For example, if your command was "`sort -a -b -c`", your hint would be:
  - `python next_clue.py [secret] 11 "-a -b -c"`



## Scavenger hunt (32/33)

### CLUE 11: THE FINAL FRONTIER

- Using everything you've learned so far, and the fact that the real clues are different from the fake clues, find the final clue!



## Scavenger hunt (33/33)

### CLUE 12: SUCCESS!

- You Found All the Clues
- This is just the start of learning Linux. Explore, google things, break things. You can also start learning shell scripting (executing multiple commands), or even Python programming. Just type:
  - python
- and you'll get an interactive programming environment.

## In sintesi

### 1. Esercitazioni online

- Linux introduction puzzle
- Linux scavenger hunt

