Computer Systems – Activities 3

UD 10. LINUX

Computer Systems
CFGS DAW

Sergio García / Alfredo Oltra
sergio.garcia@ceedcv.es
alfredo.oltra@ceedcv.es
2022/2023

Versión:220602.1649

Licencia

Nomenclatura

A lo largo de este tema se utilizarán distintos símbolos para distinguir elementos importantes dentro del contenido. Estos símbolos son:

🎖 Actividad opcional. Normalmente hace referencia a un contenido que se ha comentado en la documentación por encima o que no se ha hecho, pero es interesante que le alumno investigue y practique. Son tipos de actividades que no entran para examen

👁 Atención. Hace referencia a un tipo de actividad donde los alumnos suelen cometer equivocaciones.

UD010. LINUX
Activities 3

Try to do those activities and **<u>discuss their results or doubts in the forum</u>**, specially difficult ones.

1.1 Activity 1

Do those exercises using *touch, cat, cd, ls, mkdir, cp, mv, rmdir, rm, grep*

1. Write a command to create a new file called *names.txt*.
2. Write a command to view the content of *names.txt*.
3. Write a command to view the content of your home directory in long format (permissions, size, date,...)
4. Write a command to view the content of your current directory in long format, showing hidden files/directories (permissions, size, date,...)
5. Write a command to list all files that end with *.png* and starts with *ga*.
6. Write a command to store the result of a *ls* command in a file called *myLS.txt*, deleting existing content.
7. Write a command to store the result of a *ls* command in a file called *myLS.txt*, adding the result to the end.
8. Write a command to create a directory called *Exercise1* in your home.
9. Write a command to move all files that starts with a from your home to directory *Exercise1.*
10. Write a command to change name of directory *Exercise1* to *Ex1*.
11. Write a command to show lines of */etc/passwd* that contains word *root*.
12. Delete all elements created.

1.2 Activity 2

We have obtained this result running *ls -l* command.

```
-rw----r--        1     pepe        pepe        409    Oct 11 12:52       doc1.txt
-rw-rw-rw-        1     pepe        pepe        230    Sep  7 08:39       doc2.txt
-rw--w--w-        1     pepe        pepe        332    Sep  7 08:39       doc3.txt

-rw-r-----        1     pepe        pepe        550    Sep  7 08:39       doc4.txt
-rw-rw-rw-        1     pepe        pepe        134    Sep  7 08:39       doc5.txt

drwxrwxrwt        5     root        root        1024   Nov 15 10:40       tmp
lrwxrwxrwx        1     alina       alina       21     Oct  1 09:46       curso -> ../docs
```

1. In symbolic mode: add execution permission to owner of *doc1.txt*.
2. In symbolic mode: delete write permission to group and others of *doc2.txt*.
3. In octal mode: add execution permission to group of *doc4.txt*.
4. In octal mode: delete write permission to group and read and write permissions for others of file *doc5.txt*.
5. Write a command to change owner to *Eulogio* and group to *Eulogio* of all files of the directory.

1.3 Activity 3

1. Create user *pepito* in command line.
2. Create group *tic* in command line.
3. Change primary group of *pepito* to *tic*.

1.4 Activity 4

🎖 Solves those exercises using grep. grep. Note: you can chain *grep* commands using | redirector.

1. Show all lines of file *list.xt* that contain *lib*.
2. Show how many lines contain *mp3* in *list.txt*.
3. Show files inside */etc* directory that contain *host* string inside.
4. Show all lines of file *list.xt* that not contains *a* (uppercase or lowercase).
5. Show all lines of file *list.xt* that not contains *a* (uppercase or lowercase) and contains *m* (lowercase).

> **Tip**: | is tool to create a redirection, that is, to use the output of a command as input of another command. For example: *cat file.txt | sort .* This command consists of two commands joined by |. The output of the *cat* command is passed as an entry of the *sort* command, so the final result you will see is the file *file.txt* sorted.

1.5 Activity 5

1. Create a folder called *shared* in your home where everybody has all permissions.
2. Create groups *office1* and *office2*
3. Create users *pedro* and *pablo*. Those users have to be members of group *office1*.
4. Create users *alba* and *nerea*. Those users have to be members of group *office2*.
5. As *pedro* create a file *topsecret.txt* that only *pedro* can read and write.
6. As *pedro* create a file *sales.txt* that owner and group *office1* can read and write. Check as *Pablo* if you can do those operations.
7. As *alba* create a file *employ.txt* that every user can read and group *office2* can read and write. Check if it is right with *pedro* and *nerea*.
8. Question: if an user has read permission to a file, but that file is inside a directory that our user doesn't have execution permission and our user have read permission. Could it read the file?
9. Question: if an user has read permission to a file, but that file is inside a directory that our user doesn't have read permission and our user have execution permission . Could it read the file?

1.6 Activity 6

1. Using *setUid* bit and supposing that temporally (something like 1 hour) you have access to a machine as root and in that machine you have an user called *alumno* without sudoer permissions.

   How can we use *setUid* bit to create a backdoor?

   CLUE: file */bin/sh* could be useful.
2. How can we detect that kind of backdoors on our system? What kind of measures can we take to be safe against this kind of attack?