UNIT 10.LINUX

Activities 3 - Solutions

Computer Systems
CFGS DAW

Alfredo Oltra / Sergio Garcia / Vicent Bosch

alfredo.oltra@ceedcv.es

2020/2021

Versión:210217.1125

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. Solutions

Try to do these activities and **discuss the results or doubts in the forum**, specially the difficult ones.

1.1 Activity 1

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

1. Write a command to create a new file called *names.txt*.

```
user@hostname:~$ touch names.txt
```

2. Write a command to view the content of *names.txt*.

```
user@hostname:~$ cat names.txt
```

3. Write a command to view the content of your home directory in long format (permissions, size, date,…)

```
user@hostname:~$ ls -li
```

```
user@hostname:~$ ls -l
```

4. Write a command to view the content of your current directory in long format, showing hidden files/directories (permissions, size, date,…)

```
user@hostname:~$ ls -lia
```

5. Write a command to list all files that end with *.png* and starts with *ga*.

(Recommended: using *touch,* create empy files to verify the results)

```
user@hostname:~$ ls ga*.png
```

6. Write a command to store the result of a *ls* command in a file called *myLS.txt*, deleting existing content.

```
user@hostname:~$ ls -li > myLS.txt
```

7. Write a command to store the result of a *ls* command in a file called *myLS.txt*, adding the result to the end.

```
user@hostname:~$ ls -la >> myLS.txt
```

8. Write a command to create a directory called *Exercise1* in your home.

```
user@hostname:~$ mkdir Exercise1
```

9. Write a command to move all files that starts with a from your home to directory *Exercise1.*

```
user@hostname:~$ mv a* Exercise1/
```

10. Write a command to change name of directory *Exercise1* to *Ex1*.

```
user@hostname:~$ mv Exercise1 Ex1
```

11. Write a command to show lines of */etc/passwd* that contains word *root*.

```
user@hostname:~$ grep root /etc/passwd
```

12. Delete all elements created.

```
user@hostname:~$ rm -r Ex1
user@hostname:~$ rm myLS.txt
```

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*.

```
pepe@hostname:~$ chmod u+x doc1.txt
```

2. In symbolic mode: delete write permission to group and others of *doc2.txt*.

```
pepe@hostname:~$ chmod go-w doc2.txt
```

3. In octal mode: add execution permission to group of *doc4.txt*.

```
pepe@hostname:~$ chmod 650 doc4.txt
```

4. In octal mode: delete write permission to group, and read and write permissions for others of file *doc5.txt*.

   ```
   pepe@hostname:~$ chmod 640 doc5.txt
   ```

5. Write a command to change owner to *Eulogio* and group to *Eulogio* of all files of the directory.

   ```
   root@hostname:/home/pepe# chown eulogio:eulogio *.*
   ```

1.3 Activity 3

1. Create user *pepito* in command line.

   ```
   use the script adduser
   ```

2. Create group *tic* in command line.

   ```
   groupadd tic
   ```

3. Change primary group of *pepito* to *tic*.

   ```
   usermod -g tic pepito
   ```

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*.

   Solution: `grep "lib" list.xt`

2. Show how many lines contain *mp3* in *list.txt*.

   Solution: `grep mp3 list.txt | wc -l`

3. Show files inside */etc* directory that contain *host* string inside.

   Solution: `grep -r host /etc`

4. Show all lines of file *list.xt* that not contains *a* (uppercase or lowercase).

   Solution: `grep -vi *a* list.txt`

5. Show all lines of file *list.xt* that not contains *a* (uppercase or lowercase) and contains *m* (lowercase).

   Solution: `grep -vi *a* list.txt | grep I *m*`

1.5 Activity 5

1. Create a folder called *shared* in your home where everybody has all permissions.

```
user@hostname:~$ mkdir shared
user@hostname:~$ chmod 777 shared
```

2. Create groups *office1* and *office2*

```
user@hostname:~$ sudo groupadd office1
user@hostname:~$ sudo groupadd office2
```

3. Create users *pedro* and *pablo*. Those users have to be members of group *office1*.

```
use the script adduser to create both users.
user@hostname:~$ sudo usermod -aG office1 pedro
user@hostname:~$ sudo usermod -aG office1 pablo
```

4. Create users *alba* and *nerea*. Those users have to be members of group *office2*.

```
use the script adduser to create both users.
user@hostname:~$ sudo usermod -aG office2 alba
user@hostname:~$ sudo usermod -aG office2 nerea
```

5. As *pedro* create a file *topsecret.txt* that only *pedro* can read and write.

```
use su (switch user) to change user.
pedro@hostname:/home/user/shared$ touch topsecret.txt
pedro@hostname:/home/user/shared$ chmod 600 topsecret.txt
```

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.

```
pedro@hostname:/home/user/shared$ touch sales.txt
pedro@hostname:/home/user/shared$ chmod 660 sales.txt
pedro@hostname:/home/user/shared$ su pablo
```

```
pablo@hostname:/home/user/shared$ echo writing >> sales.txt
pablo@hostname:/home/user/shared$ cat sales.txt
```

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*.

```
alba@hostname:/home/user/shared$ touch employ.txt
alba@hostname:/home/user/shared$ chmod 464 employ.txt
```

8. Question: if a user has read permission to a file, but that file is inside a directory where our user doesn't have execution permissions, and our user has read permissions. Could it read the file?

No.

Check it:

a) Create a file in a directory with user A.

b) Change to parent directory.

c) Change to user OWNER of the directory.

d) Remove execution permissions for others (user A)

e) Change to user A.

f) From parent directory try to access using cd

g) From parent directory try to read the content of the file using cat directory/file

9. Question: if a user has read permissions 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?

Yes.

Check it:

a) Create a file in a directory with user A.

b) Change to parent directory.

c) Change to user OWNER of the directory.

d) Remove read permissions for others (user A)

e) Change to user A.

f) Move to the directory.

g) Try to read the content of the file using cat 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 a user called *alumno* without sudoer permissions.

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

   CLUE: file */bin/sh* could be useful.

   Solution

   AS root:

   cd $HOME

   cp /bin/sh ./

   chown root ./sh

   chmod 4777 ./sh

   Now we have created the backdoor

AS myuser:

Simply run ./sh and you will be root (you can check it with id command)

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?

Solution

With:

find / -path /proc -prune -o -type f -perm +4000 -ls > listado.txt

We can obtain all the files with setUID bit active. If the list changes, maybe a new setUID file has been created.

Also we can use software for “system integrity” **http://www.ossec.net/**