A Objective

This lab exercise provides an introduction to access control for files and directories in Linux. Our setup will utilize users and groups of an organization called *Gotham City* as an example!

Note: You may already have a corresponding script or completed configuration handy from a previous lab.

B Platform and process

- You will need a Linux-system where you have superuser (root) privileges (with `sudo` oder `su`) – please use the prepared Kali VM running (the privileged `sudo`-user is called `junioradmin`)!
- Execute all commands in a terminal window using the standard shell and don't forget to elevate your privileges first using e.g. `sudo su`!

→ Be sure to provide answers to all questions and to document **each and every** shell command line in your lab notes (lab report)!

C Refresh: Creating users and groups

(1) Create the following users, groups and directories (and replace *NN* by the last two digits of your login-number!):

Username	Name	Login Group	Home Directory	Password
bossNN	Your Name	hereosNN	/home/hereos_NN/*boss_NN*	I!love!linux!
batman	Bruce Wayne	hereosNN	/home/hereos_NN/batman	darkknight
robin	Dick Grayson	friends	/home/friends/robin	nightwing
catwoman	Selina Kyle	friends	/home/friends/catwoman	meowmeow
gordon	James Gordon	police	/home/police/gordon	law-n-order
joker	The Joker	villains	/home/villians/joker	whysoserious
penguin	Oswald Cobblepot	villians	/home/villains/penguin	fish!

Use a simple shell script that has one command per line by using the command `useradd`[1] (you can use the script you developed in lab "Linux User Management" and adapt it according to the table above.)

Hint: If this is a repetition, there may be a shell script *attached* or provided online!

(2) Comment each line of the shell script and insert the shell script here *after* finishing this exercise (or at home).

D File Permissions – Commands

(3) By either checking your notes, re-reading the provided hand-out, testing in the shell or having a look at the *man pages* (you can also use the web for that, e.g. a query for `man chmod` will likely help), lookup any needed information on the following commands:

[1] Note the mnemonic trick for the important options of `useradd`: **G**etting **s**omething **d**one **m**akes **c**harts **g**reen ;-)

`ls`, `chmod`, `chown`, `chgrp`

And then provide answers to the following questions *(shell commands only!)*:

(4) How can you have the permissions of a file or directory be displayed?

(5) How can you set or change both the `owner` and `group` of a file using a *single* command line?

Hint: Check the manual pages of the corresponding command in order to answer the next two questions! You may do this *after* finishing this exercise or at home.

(6) How can you change the owner of a directory, all files in that directory, all sub-folders and all files beneath the directory using a `single` command line?

(7) What does the parameter `-c` imply when used with `chown` respectively with `chgrp`?

(8) How does `chmod` work on symbolic links *(does it change the permissions of the link itself or the permissions of the link target?)*?

(9) What is the meaning of the *sticky bit* for *directories*?

→ You should verify your answers by actually *testing* the command lines on some dummy files in your Linux-VM!

E File Permissions – Lab I

(10) Explain the permissions for the files `/etc/passwd`, `/etc/shadow`, `/bin/passwd` (or `/usr/bin/passwd`, if `/bin/passwd` does not exist), `/var/log` and the directory `/home` (who is permitted to do what?)? Provide the equivalent octal notation for each file!

Hint: Please provide the actual (real) user or group names, don't use terms like "owner". Example: "junioradmin may read …", **NOT** "the owner may read …".

(11) Review the permissions of the **home directories** created earlier:

a. Who is the owner *(e.g. of* `/home/friends/robin`*)*? What is the name of the corresponding group?

b. What rights have been granted – "who is allowed to do what" in these directories?

Important: (Again) Pls. provide names, e.g. "Robin may…" and not "the owner may…".

(12) Change the group of the folders above the home directories as follows *(be sure to note down the command lines you used!)*:

Directory	Group
`/home/friends`	`friends`
`/home/villains`	`villains`
`/home/police`	`police`
`/home/heroes`*`NN`*	`heroes`*`NN`*

F File Permissions – Lab II

(Be sure to take note of the command lines you used!)

(13) We want to create shared folders for the work documents of the individual groups of our fictive organization:

a. Create a directory `/data25/friends` (Owner should be `root`), where all members of `friends` can have read and write permissions (so they can work on their documents), but everybody else (except `root`) *has no rights* whatsoever (all entitled users should of course be able to read the contents of the directory and be permitted to enter the directory).

b. Create the directories `/data25/villains` and `/data25/police`, that feature equivalent rights for the corresponding groups (read/write only for the group, all rights for `root`, group members can use the directory normally, all others have *no rights*)

c. `batman` should be able to use and write to *all* directories *(owner must still be `root` – how do you do that?)*!

(14) *IMPORTANT:* Now comes the critical part of your security setup → *TEST* your security:

For each of the following operations, consider if the type of access attempted should be possible in theory (if correct permission settings should permit it) and take a note. Then execute the corresponding command and check if the operation succeeds or not, documenting the results as you go! So the task is to write a *test log* stating – in a structured manner – if the operation should succeed, if it worked out or not and if this result is ok (as desired)!

a. As user `robin`, create a the file `robin`***`NN`*** in `/data25/friends`. The file shall contain the text "*Robin was here!*".

b. As user `gordon`, create a file in `/data25/police`.

c. Create a new directory called `stuff` in `/data25/villains` as user `joker`.

d. As user `joker`, create a file in `/data25/police`.

e. Working as user `gordon`, list all files in `/data25/friends`.

f. Have `penguin` delete a file in `/data25/friends`.

g. Logged-in as `batman`, list all files in `/data25/friends`, `/data25/police` and `/data25/villains`.

h. Create a file in `/data25/villains` as user `batman`.

i. Can `batman` append the text "*Batman was here as well!*" to file `robin`***`NN`*** in `/data25/friends`? Why/Why not?

j. Now try to delete the file `robin`***`NN`*** in `/data25/friends` as user `batman`! Does `batman` need to have write permissions on the file? Why/why not?

k. With `root`-permissions: Create a directory `/data25/gotham`, that is owned by `batman`, who may create, delete and rename files in this folder. All other users should only have *read* access (that is, be permitted to `cd` to this directory and list the contents)!

Hint: Again, you can easily switch users on the command line by entering `su - someuser` (check with `whoami`), executing the command(s) and then leaving the user shell with `exit`. Or just keep a terminal open for each user.

Recommendation: Create a *table* to document your tests! *Example:*

Command	Should succeed?	Did Succeed?	Ok?
a. su - robin; > /data25/friends	Yes	No	No
…	…	…	…

(Obviously, in above example, if real result is not equal to expected result, something's wrong and you need to check and correct your configuration).

Have fun!