

LAB 2 – Permessi, ACL e password cracking

Sicurezza e Privacy

The learning objective of this lab is for students to get the basics of file security through file ownership, file permissions, access control and password cracking. After finishing the lab, students should be able to understand the basic of Linux access control mechanism and how use tools like John the Ripper to cracking passwords.

Students are invited to complete the following exercises by write the command for solve each question in a .txt file and submit it to SPLab@di.unimi.it, with subject: 2020Lab2NameSurnameUniversityid (ex. 2020Lab2MatteoZoia876271). Message not conforming to these rules will be automatically rejected.

The first five students who will respond correctly to the entire lab will receive a bonus of 0.5pt on the final grade, 0.25pt on permission and ACL part and 0.25pt on the password cracking part.

Permissions and ACLs

Exercise #1: Permissions

1. As normal user create a directory ~/permissions. Create a file owned by yourself in there. Copy a file owned by root from /etc/ to your permissions dir, who owns this file now?
2. As root, create a file in the users ~/permissions directory. As normal user, look at who owns this file created by root?
3. Change the ownership of all files in ~/permissions to yourself.
4. Make sure you have all rights to these files, and others can only read.
5. With *chmod*, is 770 the same as *rw-rw---* ?
6. With *chmod*, is 664 the same as *r-xr-xr--* ?
7. With *chmod*, is 400 the same as *r-----* ?
8. With *chmod*, is 734 the same as *rw-r-xr--* ?
9. Display the *umask* in octal and in symbolic form.
10. Set the *umask* to 077, but use the symbolic format to set it. Verify that this works.
11. Create a file as root, give only read to others. Can a normal user read this file? Test writing to this file with *vi/vim/nano/emacs* (your favorite editor).
12. Create a file as normal user, give only read to others. Can another normal user read this file? Test writing to this file with an editor.
13. Can root read this file? Can root write to this file with an editor?
14. Create a directory that belongs to a group, where every member of that group can read and write to files, and create files. Make sure that people can only delete their own files.

Exercise #2: Sticky bit, GID and UID

1. Set up a directory, owned by the group students.
2. Members of the students group should be able to create files in this directory.
3. All files created in this directory should be group-owned by the students group.
4. Users should be able to delete only their own user-owned files.

5. Verify the permissions on `/usr/bin/passwd`. Remove the *setuid*, then try changing your password as a normal user. Reset the permissions back and try again.

Exercise #3: Users and groups

1. Show all groups that your current user is in.
2. Add new group to your machine named *school*, create a new user called George and add it to the group *school*.
3. Create a file named `homework.txt` with your current user. Print the **acls** (with the binary `/usr/bin/getfacl`) associated with your actual user and the file.
4. Add user George with octal permission 7 to the acl of file `homework.txt` (use the binary `setfacl`). Then add the group *school* with octal permission 6 to the acl of the same file. Show the results with `getfacl`.
5. With the `setfacl` binary remove an acl entry for George from the `homework.txt` file. Can you still write the file?

Exercise #4: Links

1. Create two files named `poem.txt` and `tale.txt`, put some text in them.
2. Create a hard link to `poem.txt` named `h1poem.txt`.
3. Display the inode numbers of these three files, the hard links should have the same inode.
4. Use the `find` command to list the two hardlinked files
5. Everything about a file is in the inode, except two things : name them.
6. Create a symbolic link to `tale.txt` called `sltale.txt`.
7. Find all files with inode number 2. What does this information tell you ?

Password cracking

Exercise #5: Single cracking

John the Ripper is a password cracking software used for password strength testing. It combines a number of cracking mode into one tool, autodetects password hash types, and includes a customizable cracker.

The simplest way you can use for cracking password through John is "Single cracking" mode, it will use the **login names**, **"GECOS" fields**, **"Full Name" fields**, and **users' home directory names as candidate for passwords**, also with a large set of permutations rules applied. "Single cracking" is the mode you should always start cracking with.

Inside `simplecracking` directory there is a Unix password file, you need to **crack all the password in it**.

Exercise #6: Wordlist

As you know, not all users use passwords crackable via “single mode”, many of them use passwords that are not connected with user's personal information fields. In this case the “single crack” mode is not useful. Thought “wordlist” mode, John uses a dictionary of words and it **compares the hashes of the words present in the dictionary with the password hash**. We can use any desired wordlist, John also comes with a password.lst which contains most of the common passwords.

The students have to **crack all the passwords** inside the wordlistcracking directory **with wordlist mode** (you need to find a proper dictionary somewhere).

Exercise #7: Bruteforce

When a password is not common there are less chance to find it into a dictionary, in this case John's “incremental” mode implements a brute force attack that try all password hashes in a given wordspace (ex. lowercase). Recall that this attack can only be successfully completed if the password length is small; we can assume that cracking with this mode will never terminate because of the number of combinations being too large (actually, it will terminate if you set a low password length limit), and you'll have to interrupt it earlier. **The student has to crack all password in the file bruteforce directory using john incremental mode.**

Note: The password maybe lowercase a-z or 0-9 digits, not both in the same time. All passwords have a maximum length of 6 characters.

You need to crack at list 9 password in total to pass the entire password cracking exercise.