Linux Access Control Lists

Imagine a system with the following

users: *student00*, *student01*, *student02*, *student03*, *student04*, *student05* and *student06*. In that system, users *student00* and *student01*  are members of a group called *sysop*. The user *student00* creates a new file called *script00.sh*. For this new file, the owner (*student00*) has read, write and execute permissions, the group *sysop* has read and execution permissions, and the rest of the users only have the read permission. Now, we want to give to *student05* the following permissions: read and write (but not execute permission).

With traditional Linux permission we cannot give this particular set of permissions to *student05* because neither as a member of others nor as a member of *sysop* that user would have the desired permissions. Therefore, we need a much more sophisticated system for controlling the permissions for files and directories, Access Control Lists (ACLs), supported by both Windows and Linux.

For Linux, **ACL (Access Control Lists)** provide a finer-grained control over which users can access specific directories and files than do traditional Linux permissions. Using ACLs, you can specify the ways in which each of several users and groups can access a directory or file.

Displaying access permissions

The getfacl command displays the file name, owner, group and the existing ACL for a file.

student@isc-v2:~$ getfacl my-script.sh

# file: my-script.sh

# owner: student

# group: student

user::rw-

group::rw-

other::r--

#### Setting ACLs of files

The setfacl command sets ACLs of files and directories. The -moption adds or modifies one or more rules in a file or folder's ACL.

setfacl -m ugo:user\_or\_group\_name:permissions file\_or\_folder\_name

Examples:

setfacl -m u:student04:7 script00.sh => Adds (or modifies) a rule to the ACL for the script00.sh file that gives student04 read, write and execute permissions to that file.

setfacl -m u:student04:rw- script00.sh => Adds (or modifies) a rule to the ACL for the script00.sh file that gives student04 read and write and execute permissions to that file.

setfacl -m g:sysop:r-x script00.sh => Adds (or modifies) a rule to the ACL for the script00.sh file that gives sysop read and execute permissions to that file.

setfacl -m o::6 script00.sh => Adds (or modifies) a rule to the ACL for the script00.sh file that gives others read and write permissions to that file.

setfacl -m u:student04:rx script00.sh => Adds (or modifies) a rule to the ACL for the script00.sh file that gives student04 read and execute permissions to that file.

setfacl -m u:student04:rx folder00 => Adds (or modifies) a rule to the ACL for the folder00 folder that gives student04 read and execute permissions to that folder.

setfacl -m u:student06:5 script00.sh folder00 => Adds (or modifies) a rule to the ACL for the folder00 folder and file script00.sh that gives student06 read and execute permissions to that folder and that file.

#### Removing rules

The -x option removes rules in a file or folder's ACL.

Examples:

setfacl -x u:student04 script00.sh => Removes a rule that gives student04 permission to access the files script00.sh.

setfacl -x g:sysop script00.sh => Removes a rule that gives sysop permission to access the files script00.sh.

setfacl -x u:student04 folder00 => Removes a rule that gives student04 permission to access the folder foldert00.

setfacl -x u:student06:5 script00.sh folder00 => Removes a rule that gives student06 permission to access the folder folder00 and the file script00.sh.

### 01. Setup

Create 2 additional users: **alice**, and **bob**. Create the group **nice-people** and add both alice and bob to it.

### 02. Getfacl

Create a folder called important-files in the home folder of the user student. Display the ACL of important-files. At the moment, are there any differences between using ls -la and getfacl?

### 03. Setfacl

Login as **alice**.

* Try to add a new folder called alice-files inside important-files. Can you create this folder? Why?
* Add a new rule to the ACL of the folder important-files that gives **alice** read, write and execute permissions to that folder.
* Display the ACL of the important-files folder. Display the permissions of important-files using ls -la. Do you see anything different?
* Login again as **alice** and try again to create the alice-filesfolder. Did it work?

### 04. Test ACLs

Login as **bob**.

* Try to create a file called bob.txt in the alice-files directory. Did it work?
* Add a rule to the ACL of the folder alice-files that gives to the **nice-people** group read, write and execute permissions to that folder. Try again to create the bob.txt file.

### 05. More rules

Login as **alice** and create a file called alice.txt in alice-files.

* Login as **bob** and try to modify alice.txt. Did it work?
* Add a rule specifying that any file created in the alice-filesdirectory can be modified by the group nice-people.
* As **alice**, create a new file named alice2.txt. Can **bob**modify this file?

### 06. Removing ACLs

* Remove all the rules related to the **nice-people** group in the ACL of the alice-files directory.
* Login as **bob** and check if you can still modify alice2.txt.
* Remove the ACL of the alice-files directory.