Lab 03 - Access control

Objectives

* Mandatory Access Control
* Discretionary Access Control
* Unix Permissions & ACLs
* Windows ACLs

Unix Permissions

01. Warmup

File: /etc/passwd; /etc/group; /etc/shadow

Create user, group: useradd, groupadd

Modify user, group: usermod, groupmod. Ex: Add user to group: usermod – a –G user1 group1

Login to user: su user

* Create 3 new users: **bugs**, **daffy** and **tweety**
* Check that the users and their home directories were created and to what groups they belong to.
* Add common names / descriptions for all 3 users: usermod -c

bugs: Bugs Bunny

daffy: Daffy Duck

tweety: Tweety

* Are the user accounts created so far active? Check out the shadow file, which stores passwords.
* Set a password for each of the newly created users. Check the shadow file again.
* Create a new group called **friends**, assign the users daffy and tweety to this group and remove the groups daffy and tweety.

02. Permissions

* Log in as daffy, create a file in daffy's home directory and check out its default permissions.
* Change the file's permissions so that tweety will be able to modify its content but bugs will not be allowed to either modify or see the content.
* Now change it to be the other way: bugs can read and write to the file, while tweety cannot do either.

03. Special Permissions

* Log in as daffy and create a folder called daffysfolder. Set the SETUID, SETGID and Sticky Bit permissions. List
* (*ls -l*) the permissions. What do you notice?
* Remove execute for all, but leave the special permissions (SETUID, SETGIT, Sticky Bit). What changes do you notice?
* Remove the special permissions (SETUID, SETGIT, Sticky Bit).
* Assign the minimum group permissions for daffysfolder so that users from other groups can browse the folder and read the files within it.
* Change the owner and group for daffysfolder to *bugs*.
* Log in as tweety and create a folder called tweety\_likes\_to\_share. Set the permissions for this folder in such a way that tweety can share the files with bugs and daffy. This means that bugs and daffy can browse the folder, read the content of any files, but cannot modify, rename or delete any other files than their own.
* As tweety, create a file called sensitivedata.txt and write a line of text in this file. Set no permissions for group and other for this file. Switch users to bugs, and try to read the data from this file using vi. Bugs has permission to run vi, but not to read sensitivedata.txt. So when vi attempts to read the file a “permission denied” error message will be displayed. However, if you set the SUID bit on the vi, bugs is granted access to the file. How does it work? The UNIX system doesn’t think bugs is reading file via vi, it thinks “root” is the user and hence the access is granted. Test this and use **ps** to monitor what is going on.

Linux Access Control Lists

Create following users:

*student00*, *student01*, *student02*, *student03*, *student04*, *student05* and *student06*.

In that system, users *student01* and *student02* are members of a group called *sysop*.

The user *student00* creates a folder called folder00 and a new file called *script00.sh*. The file is set: -rwxr-xr--

* Adds (or modifies) a rule to the ACL for the script00.sh file that gives student04 read, write and execute permissions to that file.
* Adds (or modifies) a rule to the ACL for the script00.sh file that gives sysop read and execute permissions to that file.
* Adds (or modifies) a rule to the ACL for the script00.sh file that gives others read and write permissions to that file.
* Adds (or modifies) a rule to the ACL for the script00.sh file that gives student04 read and execute permissions to that file.
* Adds (or modifies) a rule to the ACL for the folder00 folder that gives student04 read and execute permissions to that folder.
* 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.
* Removes a rule that gives student04 permission to access the files script00.sh.
* Removes a rule that gives sysop permission to access the files script00.sh.
* Removes a rule that gives student04 permission to access the folder foldert00.
* Removes a rule that gives student06 permission to access the folder folder00 and the file script00.sh.

**Exercise:**

00. Setup

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

01. 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?

02. 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-files* folder. Did it work?

03. 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.

04. 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-files* directory can be modified by the group *nice-people*.
* As **alice**, create a new file named *alice2.txt*. Can **bob** modify this file?

05. 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.

ACL with Windows

Reboot to Windows.

00. Setup

Create users **jack**, **john**, **outsider**. The password should be “student”. Create a new group called **jgroup** and add **jack** and **john** to it. List users, group

01. Windows ACLs (cacls)

Change the permisions for *Up* so that **outsider** has full permissions and **jack** has only read permissions. Log in as Jack. Is he able to edit *Up\Carl.txt*?

02. Setting ACLs

Edit the permissions for *Storks* recursively in such a way that **outsider** has no access. Login as **outsider** and check if he is unable to access the content of *Storks*.

03. Complex ACLs

Grant full rights to **jgroup** for *Zootopia*. Edit the rights for *Zootopia\Judy.txt* so that only **jack** can write and **john** to read, and for *Zootopia\Nick.txt* so that only **john** can write and **jack** to read. Check if the commands were correct.