**File Security in Linux**

* Linux file security is quite simplistic in design, yet quite effective in controlling access to files and directories.
* Directories and the files which are stored in them are arranged in a hierarchical tree structure.
* Access can be controlled for both the files and the directories allowing a very flexible level of access.

**File Security Model:**

* In Linux, every file and every directory are owned by a single user on that system.
* Each file and directory also has a security group associated with it that has access rights to the file or directory.
* If a user is not the directory or file owner nor assigned to the security group for the file, that user is classified as other and may still have certain rights to access the file.
* Each of the three file access categories, owner, group, and other, has a set of three access permissions associated with it.
* The access permissions are read, write, and execute.
* A user may belong to more than one group.
* Regardless of how many groups a user belongs to if permissions are granted on a file or directory to one of the user's groups they will have the granted level of access.
* You can check what groups a user belongs to with the groups command.

\$ groupstclark

tclark : authors users

* The groups command is called with one argument, the username you want to investigate.
* As you can see in the output above the output lists the username and all the groups they belong to.
* In this output tclark belongs to the groups authors and users.

**Basic File Permissions:**

**Permission Groups:**

Each file and directory has three user based permission groups:

* **Owner** - The Owner permissions apply only the owner of the file or directory, they will not impact the actions of other users.
* **Group** - The Group permissions apply only to the group that has been assigned to the file or directory, they will not affect the actions of other users.
* **All users** - The All Users permissions apply to all other users on the system, this is the permission group that you want to watch the most.

**Permission Types:**

Each file or directory has three basic permission types:

* **Read** - The Read permission refers to a user's capability to read the contents of the file.
* **Write** - The Write permissions refer to a user's capability to write or modify a file or directory.
* **Execute** - The Execute permission affects a user's capability to execute a file or view the contents of a directory.

**Viewing the Permissions:**

You can view the permissions by checking the file or directory permissions in your favorite GUI File Manager (which I will not cover here) or by reviewing the output of the "ls−l""ls−l" command while in the terminal and while working in the directory which contains the file or folder.

**The permission in the command line is displayed as: \_rwxrwxrwx 1 owner:**

1. User rights/Permissions

1.1 The first character that I marked with an underscore is the special permission flag that can vary.

1.2 The following set of three characters (rwx) is for the owner permissions.

1.3 The second set of three characters (rwx) is for the Group permissions.

1.4 The third set of three characters (rwx) is for the All Users permissions.

1. Following that grouping since the integer/number displays the number of hardlinks to the file.
2. The last piece is the Owner and Group assignment formatted as Owner: Group.

**Modifying the Permissions:**

When in the command line, the permissions are edited by using the command chmod. You can assign the permissions explicitly or by using a binary reference as described below.

Explicitly Defining Permissions

To explicitly define permission, you will need to reference the Permission Group and Permission Types.

The Permission Groups used are:

* + u - Owner
  + g - Group
  + o or a - All Users

The potential Assignment Operators are + (plus) and - (minus); these are used to tell the system whether to add or remove the specific permissions.

The Permission Types that are used are:

* + r - Read
  + w - Write
  + x – Execute

**Change File Ownership:**

As stated earlier in this chapter every file and directory in Linux has an owner and a group associated with it.  The need commonly arises where the user or group ownership for files or directories needs to be changed.  For example, if user the sally, in group finance is responsible for a number of files and Sally gets transferred to the purchasing group the ownership of the files might need to be changed to Marge because Marge is the user who is taking place in finance.  The chown command is used to change file or directory ownership.

As another example if a number of files that are currently accessed by the test group are ready for production and need to be changed to the prod group, the chgrp command can be used to give access to the prod group.

Actually the chown command can be used to change both user and group ownership, while the chgrp command can only be used to change group ownership.  This command will be covered later in this chapter.  When using either chown or chgrp commands, the system will first check the permissions of the user issuing the commands to make certain they have sufficient permissions to make the change.

Now we'll look at some examples of how to use the chown and chgrp commands.  We'll start with the chgrp command, then look at chown and then finally see how chown can be used to do the work of both.

**Change Group Ownership:**

The chgrp command is used to change the group with which a file is associated.  The first thing you will need to provide this command is the group which you want to change the file or directory to.  After that you can list a single file or directory to be changed or list separate entities separated by spaces.  The chgrp command will not have any effect on the access granted to the group (the rw- in the middle of the three permissions sets) but will change who can use those permissions.

**Using the chgrp Command on a File:**

# ls -l  
total 12  
-rw-rw-r--    1 tclark   authors      2229 Jan 13 21:35 declaration.txt  
-rw-rw-r--    1 tclark   authors      1310 Jan 13 17:48 gettysburg.txt  
-rw-rw-r--    1 tclark   authors       360 Jan 13 17:48 preamble.txt  
# chgrp presidents gettysburg.txt

# ls -l  
total 12  
-rw-rw-r--    1 tclark   authors      2229 Jan 13 21:35 declaration.txt  
-rw-rw-r--    1 tclark   presidents     1310 Jan 13 17:48 gettysburg.txt  
-rw-rw-r--    1 tclark   authors       360 Jan 13 17:48 preamble.txt

The chgrp command works the same for directories as it does for files.  In the following example, the group ownership of the directory called examples will be changed.  Directories are identified by the letter d in the first column of the ls.

**Using the chgrp Command on a Directory:**

# ls -l  
total 4  
-rw-rw-r--    1 tclark   tclark          0 Jan 13 21:13 example1.fil  
-rw-rw-r--    1 tclark   tclark          0 Jan 13 21:13 example2.xxx  
drwxrwxr-x    2 tclark   tclark       4096 Jan 13 21:35 examples  
# chgrp authors examples

# ls -l  
total 4  
-rw-rw-r--    1 tclark   tclark          0 Jan 13 21:13 example1.fil  
-rw-rw-r--    1 tclark   tclark          0 Jan 13 21:13 example2.xxx  
drwxrwxr-x    2 tclark   authors      4096 Jan 13 21:35 examples