❖CONTROLLING ACCESS TO FILES WITH ACCESS CONTROL LISTS (ACLS)

➢Access Control Lists((ACLs) are a way to assign fine tuned permissions in Linux apart from using the chmod command. When the chmod command is used only one owner and one group can be assigned permissions on a file or directory. If multiple users need access to a resource we need to place them in a group and then give that group the necessary permissions. But with File ACLs in Linux we can assign fine grained permissions to each user and group on a file and even deny access to a particular user even if the file has world permissions. This tutorial on Linux File ACL will explain the usage of the commands getfacl and setfacl.

❖CONTROLLING ACCESS TO FILES WITH ACCESS CONTROL LISTS (ACLS)

➢First let us understand the purpose of each permission on files and directories

•Files –

•r (read) - The contents of the file can be viewed
w (write) - The file can be edited and new content can be inserted
x (execute) – The file can be executed.

•Directories –

•r (read) - The contents of the directory can be viewed with the “ls” command
w (write) - New file can be created inside the directory and existing files can be deleted
x (execute) – The user with this permission can change directory (cd) into this directory

❖CONTROLLING ACCESS TO FILES WITH ACCESS CONTROL LISTS (ACLS)

If you get a command not found error for getfacl and setfacl it means the acl package is not installed, so use yum or apt-get according to your operating system to install the package

yum install acl

apt-get install acl

❖CONTROLLING ACCESS TO FILES WITH ACCESS CONTROL LISTS (ACLS)

➢To view the access control list of a file/directory, use the getfacl command

```
[sverma@Server1 tmp]$ getfacl test1
# file: test1
# owner: sverma
# group: sverma
user::rwx
group::rwx
other::r-x
default:user::rwx
default:user:priya:rw-
default:group::rwx
default:mask::rwx
default:other::r-x
```

❖CONTROLLING ACCESS TO FILES WITH ACCESS CONTROL LISTS (ACLS)

- Before using the setfacl command, acl has to be enabled on the filesystem, else you'll receive the following error.

- setfacl: /path/to/file: Operation not supported

File system mount option

The file system needs to be mounted with ACL support enabled. XFS file systems have built-in ACL support. Ext4 file systems created on Red Hat Enterprise Linux 7 have the **acl** option enabled by default, but ext4 file systems created in earlier versions of Red Hat Enterprise Linux may need the **acl** option included with the mount request, or set in the superblock.

❖CONTROLLING ACCESS TO FILES WITH ACCESS CONTROL LISTS (ACLS)

➢Let's see what ACLs can do for us. First we will create a file named text.cfg and we will give it as an argument to the getfacl command. Let's see what the output of this command shows:

```
[sverma@Server1 tmp]$ rm text.cfg
[sverma@Server1 tmp]$ touch text.cfg && getfacl text.cfg
# file: text.cfg
# owner: sverma
# group: sverma
user::rw-
group::rw-
other::r--
```

❖CONTROLLING ACCESS TO FILES WITH ACCESS CONTROL LISTS (ACLS)

- As you can see, since we didn't set any ACL permission on the file, the command just displays the standard permissions values, plus the file owner and the group owner, both having read and write permissions.

- Now let's imagine we want to give a specific user

- [sverma@Server1 tmp]$ setfacl -m u:priya:rw text.cfg
- Here, -m which allows us to change the ACL's of a file/Directory
- u stands for user, g ➔ It could have been for group or an "o" for others
- priya ➔ User Name
- rw ➔ Permissions
- Text.cfg ➔ The name of the file on which we want to apply the permissions.

❖CONTROLLING ACCESS TO FILES WITH ACCESS CONTROL LISTS (ACLS)

➢getfacl text.cfg

```
[sverma@Server1 tmp]$ getfacl text.cfg
# file: text.cfg
# owner: sverma
# group: sverma
user::rw-
user:priya:rw-
group::rw-
mask::rw-
other::r--
```

❖CONTROLLING ACCESS TO FILES WITH ACCESS CONTROL LISTS (ACLS)

```
[sverma@Server1 tmp]$ getfacl text.cfg
# file: text.cfg
# owner: sverma
# group: sverma
user::rw-
user:priya:rw-
group::rw-
mask::rw-
other::r--
```

An entry for the user "priya" has been added and showing permissions whatever we have assinged to her.We can notice one more entry i.e. mask has appeared.

The mask associated with an ACL limits the set of permissions that can be assigned on the file for the the named groups and users and for the group owner, but has no effect on the permissions for the file owner and the other permission group.

- In this case, only reading and writing permissions could be assigned with setfacl command. Of course we can change this option, using setfacl program itself:

- setfacl -m mask:r text.cfg

```
[sverma@Server1 tmp]$ setfacl -m mask:r text.cfg
[sverma@Server1 tmp]$ getfacl text.cfg
# file: text.cfg
# owner: sverma
# group: sverma
user::rw-
user:priya:rw-                  #effective:r--
group::rw-                      #effective:r--
mask::r--
other::r--
```

- After changing the mask value to "r" and permissions has been effective for group owner and priya user. user have reading and writing permissions on the file, by changing the mask, we have effectively limited their permissions to read only. As the output of the command shows, they now are only allowed to read the file.

- Other than explicitly changed with the command above, the ACLs mask also gets automatically re-calculated when we assign or change permissions with setfacl (unless the -n option is specified). Let's demonstrate that: we will change the permissions of the priya user to rwx and then check the getfacl output:

```
[sverma@Server1 tmp]$ setfacl -m mask:rwx text.cfg && getfacl text.cfg
# file: text.cfg
# owner: sverma
# group: sverma
user::rw-
user:priya:rw-
group::rw-
mask::rwx
other::r--
```

- As you can see the mask got re-calculated and it now reflects the maximum permissions present for the named user priya. Obviously, since now no previously set permissions are higher than the mask, there is no need for showing the #effective permission status.

```
[sverma@Server1 tmp]$ setfacl -m mask:rwx text.cfg && getfacl text.cfg
# file: text.cfg
# owner: sverma
# group: sverma
user::rw-
user:priya:rw-
group::rw-
mask::rwx
other::r--
```

- Default ACLs

- The default ACL is a specific type of permission assigned to a directory, that doesn't change the permissions of the directory itself, but makes so that specified ACLs are set by default on all the files created inside of it. Let's demonstrate it: first we are going to create a directory and assign default ACL to it by using the -d option:

- $ mkdir dir && setfacl -d -m u:dummy:rw dir

- Default ACLs

```
[sverma@Server1 tmp]$ getfacl testfile
# file: testfile
# owner: sverma
# group: sverma
user::rwx
group::rwx
other::r-x

[sverma@Server1 tmp]$ setfacl -d -m u:priya:rw testfile/
[sverma@Server1 tmp]$ getfacl testfile/
# file: testfile/
# owner: sverma
# group: sverma
user::rwx
group::rwx
other::r-x
default:user::rwx
default:user:priya:rw-
default:group::rwx
default:mask::rwx
default:other::r-x
```

- Default ACLs

- The default permissions has been assigned correctly. Now we can verify them by creating a file inside of the test directory and checking its permissions by running getfacl:

```
[sverma@Server1 tmp]$ touch testfile/file.cfg && getfacl testfile/file.cfg
# file: testfile/file.cfg
# owner: sverma
# group: sverma
user::rw-
user:priya:rw-
group::rwx                      #effective:rw-
mask::rw-
other::r--
```

- As expected, the file has been created automatically receiving the ACLs permissions specified above.

✓Lab Session