File Access Control Lists (ACLs)

Exercise Setup

These exercises assume you are using RHEL 7.2 or CentOS 7.2

Create the following users and assignment passwords if these users do not already exist on your system: *user1*, *user3*, *user4*.

If */home* is a separate filesystem (check output of *dh*) on your system and is not of type *ext3*, *ext4* or *xfs*, you may need to remount the */home* filesystem with the *acl* option enabled (*mount -o remount,acl ...*) Check your system documentation.

Exercise 10-13   Determine, Set, and Delete ACLs

In this exercise, you will create *file1* as *user1* in */home/user1* and check to see if there are any ACL settings on *file1*. You will apply ACL settings on *file1* for *user3* to allow full access to *file1* by *user3*. You will observe the change in the value of the *file1* ACL mask. You will add *user4* to the ACL settings for *file1*. You will delete the ACL settings for *user3*, and then delete all other ACL settings for *file1*.

1. Log in as *user1* and create *file1*. Look at the default Unix permissions and file ACLs using *ls* and *getfacl*.

```
$ pwd
/home/user1
$ touch file1
$ ls -l file1
-rw-rw-r--. 1 user1 user1 0 Jan 31 21:09 file1
$ getfacl file1
# file: file1
# owner: user1
# group: user1
user::rw-
group::rw-
```

```
other::r--
```

The output of these two commands indicate an absence of additional file ACLs on *file1*. The 3 ACLs displayed are known as the *base* ACLs and match the Unix permissions. If there were additional ACLs on *file1*, the period after the 9 Unix file permissions would have been replaced by a +, and the ACLs would have been displayed by the *getfacl* command as extra lines of output.

2. Add read/write/execute permissions on *file1* for *user3* using *setfacl* and symbolic permissions.

   ```
   $ setfacl --test -m u:user3:7 file1
   file1: u::rw-,u:user3:rwx,g::rw-,m::rwx,o::r--,*
   $ setfacl -m u:user3:rwx file1
   $ ls -l
   -rw-rwxr--+ 1 user1 user1 0 Jan 31 21:09 file1
   $ getfacl file1
   # file: file1
   # owner: user1
   # group: user1
   user::rw-
   user:user3:rwx
   group::rw-
   mask::rwx
   other::r–
   ```

Only *root* or the owner of a file can set ACLs on a file. Note the + sign in the ls output and the extra two lines of output from *getfacl.*

If an ACL contains named user or named group entries, and no mask entry exists, a mask entry containing the same permissions as the group entry is created. The mask value indicates the maximum permissions (in this case *rwx*) allowed for a user, other than the owner, or for a group.

Technically, the mask entry is set to the union of all permissions of the owning group, and all named user and group entries.

The *--test* option to *setfacl* enables you to see the resulting ACLs without actually changing the file ACLs.

3. Add *user4* with read/write permissions to file1 ACLs using numeric permissions.

   ```
   $ setfacl -m u:user4:6 file1
   $ ls -l
   -rw-rwxr--+ 1 user1 user1 0 Jan 31 21:09 file1
   $ getfacl file1
   # file: file1
   # owner: user1
   # group: user1
   user::rw-
   user:user3:rwx
   user:user4:rw-
   group::rw-
   mask::rwx
   other::r--
   ```

4. Delete all ACLs on *file1* for *user3* and confirm.

   ```
   $ setfacl -x u:user3 file1
   $ getfacl file1
   # file: file1
   # owner: user1
   # group: user1
   user::rw-
   user:user4:rw-
   group::rw-
   mask::rw-
   other::r--
   ```

Note that the mask is now *rw-* instead of *rwx*, which reflects the current maximum permissions for *user4.*

5. Delete all ACLs on *file1* and confirm deletion.

```
$ setfacl -b file1
$ ls -l file1
-rw-rw-r--. 1 user1 user1 0 Jan 31 21:09 file1
$ getfacl file1
# file: file1
# owner: user1
# group: user1
user::rw-
group::rw-
other::r--
```

Exercise 10-14 Set, Confirm and Delete Default ACLs

In this exercise, you will create a subdirectory */home/user4/projects* as *user4* and set default ACLs for *user1* and *user3* to allow them read and write permissions on the projects subdirectory. You will create a subdirectory *project1* and a file *file1* under *projects* and observe the effect of default ACLs on *project1* and *file1*. You will delete all the default entries at the end of the exercise and the entire projects subdirectory.

1. Login in as *user4* and create the *projects* subdirectory. Run the *getfacl* command and see what Unix permissions and default ACLs are on the directory.

   ```
   $ id -un
   user4
   $ pwd
   /home/user4
   $ mkdir projects
   $ ls -ld projects/
   drwxrwxr-x. 2 user4 user4 6 Feb  2 05:56 projects/
   $ getfacl projects
   # file: projects
   # owner: user4
   # group: user4
   user::rwx
   group::rwx
   other::r-x
   ```

2. Allocate default read and write permissions to *user1* and *user3* with the *setfacl* command. Use octal notation.

   ```
   $ setfacl -m d:u:user1:6,d:u:user3:6 projects
   $ getfacl -c projects
   user::rwx
   group::rwx
   other::r-x
   ```

```
default:user::rwx
default:user:user1:rw-
default:user:user3:rw-
default:group::rwx
default:mask::rwx
default:other::r-x
```

The *-c* option tells *getfacl* not to display the comment header, i.e. the first three lines of output for a file.

3. Create a subdirectory *project1* under *projects* and observe that it inherited the ACL settings of the parent directory.

   ```
   $ pwd
   /home/user4
   $ cd projects
   $ mkdir project1
   $ ls -ld project1
   drwxrwxr-x+ 2 user4 user4 6 Feb  2 06:10 project1
   $ getfacl -c project1
   user::rwx
   user:user1:rw-
   user:user3:rw-
   group::rwx
   mask::rwx
   other::r-x
   default:user::rwx
   default:user:user1:rw-
   default:user:user3:rw-
   default:group::rwx
   default:mask::rwx
   default:other::r-x
   ```

4. Create a file *file1* in the *projects* subdirectory and observe the inheritance of file ACLs from the directory ACLs.

```
$ pwd
/home/user4/projects
$ touch file1
$ ls -l file1
-rw-rw-r--+ 1 user4 user4 0 Feb  2 06:14 file1
$ getfacl -c file1
user::rw-
user:user1:rw-
user:user3:rw-
group::rwx              #effective:rw-
mask::rw-
other::r--
```

The above output indicates the maximum permissions group members have on the file is read and write. The execute permission is ineffective due to the setting of the mask.

5. Delete all the default ACLs from the subdirectory and confirm.

```
$ pwd
/home/user4/projects
$ cd ..
$ setfacl -k projects
$ getfacl -c projects
user::rwx
group::rwx
other::r-x

$ getfacl file1
# file: file1
# owner: user4
# group: user4
user::rw-
user:user1:rw-
user:user3:rw-
```

```
group::rwx              #effective:rw-
mask::rw-
other::r--

$ getfacl -c project1
user::rwx
user:user1:rw-
user:user3:rw-
group::rwx
mask::rwx
other::r-x
default:user::rwx
default:user:user1:rw-
default:user:user3:rw-
default:group::rwx
default:mask::rwx
default:other::r-x
```

Note that ACLs were not removed from *file1* or *project1.*

6. Remove the *projects* subdirectory.

```
$ pwd
/home/user4/projects
$ cd ..
$ rm -rf projects
```