ACLs

The critical shortfall for permissions is that there are only three "blocks" to assign permissions: if you are not the owner then you must be a member of the owning group; if you are not the owner and not a member of the owning group, then your permissions are determined by the “everybody else” category. Access Control Lists augment the standard UNIX file permissions by allowing permissions for more than one user or/and more than one group: with ACLs you can assign file permissions (that is: to allow or deny read, write and execute privileges) to several individuals users or/and several individuals groups **in addition** to the owner and the owning group for each file and directory.

ACLs currently are supported out of the box in XFS, BTRFS and current Ext4 (via /etc/mke2fs.conf) filesystems (in fact, "acl" package is a dependency os systemd). Anyway, we could be sure of this simply executing *tune2fs -l /dev/sda1 | grep "Default mount options:"* and seeing "user_xattr acl" values on screen. More about that in *man acl*

> NOTA: If necessary, you could set the default mount options of a filesystem using the *tune2fs -o* option (for example: *tune2fs -o acl /dev/sdXY*). Using the default mount options instead of an entry in /etc/fstab is very useful for external drives because their partition will be mounted with -in this case- "acl" option also on other Linux machines without the need to edit /etc/fstab on every machine.

ACLs are a type of "extended attribute" (specifically, a "system" type), so in theory they could be managed by standard *getfattr* and *setfattr* commands but there are other utilities much more targeted and convenient to do this job: *getfacl* and *setfacl*.

> NOTA: There are two types of ACLs: "normal" ACLs, which are applied to a file or directory, and "default" (optional) ACLs, which can only be applied to a directory. If files inside a directory where a default ACL has been set do not have a ACL of their own, they inherit the default ACL of their parent directory.

To know which ACL is assigned to a specific file, we can do the following:

getfacl /ruta/fitxer (or *getfattr -d -m system /ruta/fitxer*)

To change the ACL of a specific file or folder, we can use the -m ("modify") parameter of *setfacl*:

setfacl -m u:userName:rw /ruta/fitxerOcarpeta <-- Concedeix permisos rw a "userName"
El permís x el deixa com estava.
setfacl -m g:groupName:r-x /ruta/fitxerOcarpeta <-- Concedeix permisos r-x a "groupName"

setfacl -m u:userName:r--,g:groupName:r-- /ruta/fitxerOcarpeta <--Varis permisos de cop

If a directory is managed, *-R* parameter (Recursive down into files and directories) can be used

What about using the *setfacl* command to change normal User, Group, and Other permissions? No problem! This can be used instead of chmod:

setfacl -m u::rwx,g::rwx,o::rwx /ruta/fitxerOcarpeta

To set a default ACL to a directory (which its contents will inherit unless overwritten otherwise), you could add *-d* parameter (optionally with -R to do it recursively) and specify a directory instead of a file name:

setfacl -R -d -m u:userName:rwx /ruta/carpeta

To remove a specific ACL, replace -m in the commands above with -x. For example:

setfacl -x g:grupName /ruta/fitxerOcarpeta

Alternatively, you can also use the -b option to remove ALL ACLs in one step:

setfacl -b /ruta/fitxerOcarpeta

If you see in *a ls* output a "+" sign after permissions column, it reveals that in this particular file there is something different than other files because of adding extended attributes. This trick could be a simple manner to detect wich files have an ACL assigned.

Finally...what's the "mask" entry shown by *getfacl*? This is the "effective rights mask". This entry limits the effective rights granted to all ACL groups and ACL users. Effective masking is used to restrict action(s) for all users and groups, that are defined in ACL. It means, for example, that you can prevent all users from writing to a file by setting the effective mask r-x.The traditional Unix User, Group, and Other entries are not affected. If the mask is more restrictive than the ACL permissions that you grant, then the mask takes precedence. To change it, we must do the following (in this example we change the mask of a file to "r--"...that would prevent any ACL-defined user or group to write on that file regardless of their own permissions):

setfacl -m mask::r-- /ruta/fitxer

EXERCICIS:

0.-a) En una màquina virtual qualsevol crea dos usuaris anomenats "alice" i "bob" (i assigna'ls una contrasenya!).

NOTA: Recorda que per crear un usuari (anomenat "pepe", per exemple) i assignar-li una contrasenya només cal executar (com a root) les comandes: *useradd -m pepe && passwd pepe*

b) Crea una carpeta anomenada "/opt/documents" i comprova que aquesta tingui com a propietari l'usuari i grup "root" (en principi serà així perquè l'hauràs hagut de crear fent servir *sudo*, però per si de cas)

NOTA: Recorda que en el cas de què hagis d'especificar un altre propietari i grup diferent del que tingui en un moment donat una determinada carpeta o fitxer (per exemple, l'usuari "pepe" i grup "pepe") només caldrà executar (com a root) la comanda: *chown pepe:pepe /ruta/carpeta*

c) Fes que la carpeta "/opt/documents" tingui permisos 700

NOTA: Recorda que per fer que una carpeta o fitxer tingui uns determinats permisos (per exemple, 644) només cal executar (com a root) la comanda: *chmod 644 /ruta/carpeta*

d) Crea un grup d'usuaris anomenat "amics" i fica-hi a dins tant "alice" com "bob".

NOTA: Recorda que per crear un grup (anomenat "amics", per exemple) només cal executar (com a root) la comanda: *groupadd amics* I per afegir un determinat usuari (anomenat "pepe", per exemple) a aquest grup, només cal executar (com a root) la comanda *usermod -a -G amics pepe*

1.-a) Obre un terminal i executa *su -l* per ser l'usuari "root", obre un altre terminal i executa *su -l alice* per ser "alice" i obre un altre terminal i executa *su -l bob* per ser "bob".

b) Fes (com a root) una ACL que permeti que "alice" pugui llistar (r), accedir (x) i modificar (w) el contingut de la carpeta /opt/documents. Prova-ho executant les comandes *ls /opt/documents*, *cd /opt/documents* i *echo "a" > a.txt*, respectivament, essent "alice". ¿Podrà "alice" modificar el contingut del fitxer *a.txt* (amb *nano*, per exemple)? Per què?

c) Fes (com a root) una ACL que permeti que "bob" només pugui llistar (r) i accedir (x) a la carpeta /opt/documents. Prova-ho executant les comandes *ls /opt/documents*, *cd /opt/documents* i *echo "b" > b.txt*, respectivament, essent "bob". ¿Què passa ara?

d) Esborra (com a root) totes les ACL de /opt/documents per tornar a començar. Comprova llavors que ara ni "alice" ni "bob" poden fer res en aquesta carpeta (ni llistar el seu contingut, ni accedir-hi, ni afegir-hi/eliminar-hi/renombrar-hi fitxers).

e) Fes (com a root) una altra ACL que permeti que als membres del grup "amics" puguin llistar i accedir a la carpeta /opt/documents. Prova-ho executant les comandes *ls /opt/documents*, *cd /opt/documents* i *echo "c" > c.txt*, respectivament tant essent "alice" com també essent "bob". ¿Què passa en cada cas?

f) Canvia (com a root) la màscara ACL de /opt/documents per a què no tingui cap permís. ¿Què poden fer ara "alice" i "bob" en aquesta carpeta? Finalment, torna a establir la màscara a "rwx"

The basic file commands (*cp* , *mv* , *ls* , *tar*, ...) support ACLs, as do Samba and Nautilus. When archiving a folder with *tar*, however, you must use the *--acls* option to preserve ACLs. Similarly, when using *cp* to copy files with ACLs you must include the *--preserve=mode* option to ensure that ACLs are copied across too (in addition, the *-a* option is equivalent to *-d R --preserve=all*

2.-a) Treu (com a root) el permís d'execució a qualsevol usuari (ja sigui propietari, grup del propietari o la resta) pel binari /bin/ls. Comprova que, efectivament, ara no el puguis executar, sigui quin sigui el teu usuari. Fes (com a root) una ACL que permeti a l'usuari "alice" executar aquest binari. Prova-ho amb *getfacl* i també entrant en una sessió de terminal d'"alice" mitjançant la comanda *su -l alice* i intentant executar *ls* allà

b) Executa (com a usuari "normal") la comanda *cp /bin/ls ~/Escriptori* i comprova amb *getfacl* si la còpia manté l'ACL de l'original assignada a l'apartat anterior (o no). ¿Què passa si executes en canvi la comanda *cp --preserve=mode /bin/ls ~/Escriptori* ?

3.-a) Dedueix (consultant a la pàgina del manual de les comandes oportunes, si calgués) i explica què fa aquest conjunt de comandes, pas a pas (el pots provar, si vols):

```
getfacl -R ~/Escriptori > acls.txt
chmod -R ugo-rw ~/Escriptori
cd ~
setfacl --restore=acls.txt
```

b) Si només es vol "traspassar" una ACL ja definida per escrit a un determinat fitxer (sense fer cap recursió), es pot fer servir en comptes del paràmetre *--restore* de *setfacl* el paràmetre *-M*. Sabent això, explica què fa aquest conjunt de comandes, pas a pas (el pots provar, si vols):

```
echo "u:usuari:rwx" > acl.txt
touch prova.txt
setfacl -M acl.txt prova.txt
```

c) Què fa aquesta canonada de comandes? (tingues en compte els diferents paràmetres que apareixen)

```
getfacl unfitxer.txt | setfacl -b -n -M – prova.txt
```