203 - Permisos simbòlics

Curs 2020 - 2021

ASIX M01-ISO UF1-A01-03 Permisos simbòlics i executables

Permisos

Descripció

Conceptes clau:
- ❏ permisos executables
- ❏ permisos simbòlics

Ordres a treballar:
- ❏ chmod +x
- ❏ chmod

Permisos simbòlics i executables

Permisos executables

Els programes executables han de tenir activat el permís x per poder-se executar. Els programes executables són fitxers binaris o de text (scripts) que per executar-se necessiten tenir activat el bit d'execució x.

Per defecte en els fitxers no s'activa mai el permís x per evitar que es puguin executar involuntàriament programes maliciosos. Cal sempre *assignar manualment* el permís d'execució. No s'ha d'establir mai una umask que l'activi per defecte.

El mecanisme més senzill d'activar el permís d'execució és amb l'ordre:

chmod +x programa

Exemple d'execució d'un programa python:

```
$ cat programa.py
#! /usr/bin/python
print("hello world!")

$ ./programa.py
bash: ./programa.py: Permission denied

$ chmod +x programa.py
$ ls -l programa.py
-rwxwxr-x 1 ecanet ecanet 42 19 des 21:46 programa.py

$ ./programa.py
hello world!
```

- observeu que el codi del programa simplement mostra un missatge de hello.
- en intentar executar directament el programa falla perquè no té permisos d'execució.
- s'assigna al programa permisos d'execució usant la notació simbòlica. S'assigna el permís x a tot (propietari, grup i altres).
- ara el programa s'executa correctament.

Exemple d'execució d'un programa script en bash:

```
$ cat script.sh
#! /bin/bash
echo "hello world again!"

$ ./script.sh
bash: ./script.sh: Permission denied

$ chmod +x script.sh
$ ls -l script.sh
-rwxrwxr-x 1 ecanet ecanet 40 19 des 21:50 script.sh

$ ./script.sh
hello world again!
```

Observem que els executables del sistema (les ordres) també tenen el permís x activat:

```
$ ls -l /usr/bin/date /usr/bin/cal /sbin/useradd
-rwxr-xr-x. 1 root root 133352 14 ago  2017 /sbin/useradd
-rwxr-xr-x. 1 root root  57704 27 mar  2018 /usr/bin/cal
-rwxr-xr-x. 1 root root 106928 20 abr  2018 /usr/bin/date
```

```
$ ls -l /usr/bin/passwd /usr/bin/write
-rwsr-xr-x. 1 root root 27872 12 abr  2018 /usr/bin/passwd
-rwxr-sr-x. 1 root tty  19584 27 mar  2018 /usr/bin/write
```

Permisos simbòlics

Per establir permisos amb l'ordre chmod es poden usar dos mecanismes, els permisos en *octal* i els permisos *simbòlics*. Quan s'han d'assignar tots els permisos a l'element definint-los tots s'utilitzen els simbòlics. Quan es vol modificar només alguna part dels permisos es poden usar els simbòlics.

El funcionament dels permisos simbòlics utilitza els caràcters següents:

+r | +w | +x amb el símbol més s'activen aquests permisos. si no s'indica on s'activen en els tres blocs (propietari, grup i altres) si la màscara o permet.

-r | -w | -x amb el símbol menys es deactiven aquests permisos. si no s'indica on es desactiven en els tres blocs (propietari, grup i altres).

u g o a significa u user, g group, o other i a all. Serveixen per indicar on cal aplicar els permisos.

= l'operador igual serveix per establir en un bloc els mateixos permisos que hi ha en un altre bloc.

La millor manera d'entendre el funcionament és observar-ne uns quants exemples:

chmod +x elem

activa el permís x a tots els blocs d'elem, és a dir a propietari, grup i altres. els altres permisos es queden tal i com estaven. Sempre que l màscara ho permeti.

chmod +rw elem

activa el permís rw al propietari el grup i altres (si la màscara ho permet). El permís x en cada cas es queda tal i com estava.

chmod +rx,-w elem

activa el permís rx al propietari grup i altres (si la màscara ho permet) i desactiva el permís x també al propietari, grup i altres.

chmod u+rw,o-x elem

activa el permís rw només al propietari (u user). Desactiva el permís x a altres (o others). La resta de permisos es queda tal i com estava.

chmod go+r,a-x elem

activa el permís r a group i others i desactiva la x a tot (propietari, grup i altres). la resta de permisos es queden tal i com estaven.

chmod a=rw

assigna a tots els blocs els permisos rw.

chmod g=u elem

assigna a group els mateixos permisos que té el propietari (siguin els que siguin) els altres permisos es queden tal i com estaven.

chmod ug=o elem

assigna al propietari i al grup els mateixos permisos que té others.

chmod g=r,o=wx elem

assigna a grup els permís r i a altres els permisos wx.

Exercicis d'exemple

Primerament heu de fer l'exercici **00-Creació_de_estructura** per generar els directoris i els fitxers necessaris per fer aquest exercici.

1. Fer actiu el directori */tmp/mp1*.

   Des d'aquest directori realitzarem tots els exercicis, tots. No és permès de canviar de directori amb l'ordre cd. Totes les ordres del sistema es poden realitzar des de qualsevol directori actiu.

   Després de fer aquest directori actiu verificar-ho amb l'ordre *pwd*.

```
$ cd /tmp/m01
$ pwd
/tmp/m01
```

   Recordeu quina és l'estructura de directoris i fitxers que estem utilitzant:

```
$ tree /tmp/m01
/tmp/m01
├── operatius
│   ├── apunts
│   │   ├── carta.txt
│   │   ├── dades.pdf
│   │   ├── dossier.odt
│   │   ├── informe.pdf
│   │   ├── projecte.odt
│   │   └── treball.txt
│   └── exercicis
└── xarxes

$ tree /var/tmp/prova/
/var/tmp/prova/
```

2. Observa els permisos de les ordres ls, cat, id, passwd, write, useradd i userdel.

```
$ ls -l /usr/bin/{ls,cat,id,passwd,write} /sbin/{useradd,userdel}
-rwxr-xr-x. 1 root root 133352 14 ago  2017 /sbin/useradd
-rwxr-xr-x. 1 root root  87344 14 ago  2017 /sbin/userdel
-rwxr-xr-x. 1 root root  36784 20 abr  2018 /usr/bin/cat
-rwxr-xr-x. 1 root root  44928 20 abr  2018 /usr/bin/id
-rwxr-xr-x. 1 root root 133216 20 abr  2018 /usr/bin/ls
-rwsr-xr-x. 1 root root  27872 12 abr  2018 /usr/bin/passwd
-rwxr-sr-x. 1 root tty   19584 27 mar  2018 /usr/bin/write
```

3. Crea un programa executable en python que mostri el teu nom i executa'l.

```
$ cat nom.py
#! /usr/bin/python
print("pere pou prat")
```

```
$ chmod +x nom.py

$ ./nom.py
pere pou prat
```

4. Escriu un script bash que mostri la data i el teu nom. Executa'l.

```
$ cat nom.sh
#! /bin/bash
date
echo "pere pou prat"

$ chmod +x nom.sh

$ ./nom.sh
ds des 19 22:19:37 CET 2020
pere pou prat
```

5. Desactiva simbòlicament els permisos w i activa els permisos x al fitxer elem.txt

```
$ ls -l elem.txt
-rw-rw-r-- 1 ecanet ecanet 28 19 des 22:27 elem.txt

$ chmod -w,+x elem.txt
$ ls -l elem.txt
-r-xr-xr-x 1 ecanet ecanet 28 19 des 22:27 elem.txt
```

6. Assigna al propietari del fitxer elem.txt els permisos rw simbòlicament.

```
$ chmod u=rw elem.txt
$ ls -l elem.txt
-rw-r-xr-x 1 ecanet ecanet 28 19 des 22:27 elem.txt
```

7. Assigna a grup i others els mateixos permisos que té el propietari en el fitxer elem.txt.

```
$ chmod go=u elem.txt
$ ls -l elem.txt
-rw-rw-rw- 1 ecanet ecanet 28 19 des 22:27 elem.txt
```

8. Estableix a grup i altres només el permís r en el fitxer elem.txt

```
$ chmod go=r elem.txt
$ ls -l elem.txt
-rw-r--r-- 1 ecanet ecanet 28 19 des 22:27 elem.txt
```

9. Estableix en el fitxer elem.txt a grup activa el permís w i a altres desactiva el permís r.

```
$ chmod g+w,o-r elem.txt
$ ls -l elem.txt
-rw-rw---- 1 ecanet ecanet 28 19 des 22:27 elem.txt
```

10. Estableix a tots els elements que els permisos siguin rwx en el fitxer elems.txt

```
$ chmod a=rwx elem.txt
$ ls -l elem.txt
-rwxrwxrwx 1 ecanet ecanet 28 19 des 22:27 elem.txt
```