# 202 - Permisos umask/chown/chgrp

Curs 2020 - 2021

ASIX M01-ISO UF1-A01-03 Permisos umask / chown / chgrp



---

# Permisos

---

## Descripció

**Conceptes clau:**

- ❏ màscara de permisos
- ❏ fitxers / directoris
- ❏ canvi de propietari
- ❏ canvi de grup

**Ordres a treballar:**

- ❏ umask
- ❏ chown
- ❏ chgrp

## Permisos per defecte: umask

### Permisos per defecte: umask

Quan es crea un fitxer o un directori té assignats uns permisos per defecte que depenen de cada usuari. No són sempre els mateixos per tothom sinó que cada usuari es pot personalitzar quins són els permisos per defecte que han de tenir els fitxers nous que es

creen i els directoris nous que es creen. Tampoc són els mateixos permisos per a fitxers que per a directoris.

```
$ cal > calendat.pdf
$ ls -l calendat.pdf
-rw-rw-r-- 1 ecanet ecanet 174 19 des 19:57 calendat.pdf

$ mkdir bdades
$ ls -ld bdades/
drwxrwxr-x 2 ecanet ecanet 40 19 des 19:57 bdades/

$ umask
0002
```

- observeu que en aquest exemple el fitxer s'ha creat amb permisos 664 i en canvi el directori amb permisos 775. L'umask d'aquest usuari és 0002.

Cada usuari pot personalitzar els permisos per defecte amb l'ordre ***umask***. Diferents usuaris poden tenir valors d'umask diferents i per tant els permisos amb els que crearan fitxers i directoris no seran els mateixos els d'un usuari que els d'un altre.

```
[user01@a36 m01]$ date > date.txt
[user01@a36 m01]$ ls -l date.txt
-rw-r--r-- 1 user01 ecanet 29 Dec 19 20:00 date.txt

[user01@a36 m01]$ mkdir hardware
[user01@a36 m01]$ ls -ld hardware/
drwxr-xr-x 2 user01 ecanet 40 Dec 19 20:01 hardware/

[user01@a36 m01]$ umask
0022
```

- en aquest usuari els fitxers es creen amb els permisos per defecte 644 i els directoris 755. L'umask d'aquest usuari és 0022.

**Com funciona umask?**

Umask és un valor numèric personalitzable per a cada usuari, es pot definir per una sessió o de manera permanent en els fitxers de configuració de l'entorn d'usuari (s'estudia més endavant).

És una màscara dels permisos que es volen obtenir, és a dir, no es diu quins permisos es volen sinó un valor del que amb una resta s'obtenen els permisos efectius.

El càlcul dels permisos efectius donada una màscara es realitza de manera diferent per als fitxers que per als directoris.

**Màscara: càlcul per a fitxers**

```
 6 6 6
-umask
------------
permisos-de-fitxer-per defecte
```

Per calcular els permisos per defecte dels fitxers es resta la màscara del valor simbòlic ***666***. El resultat són els permisos de fitxers.

Exemple-1:
donada la màscara 002
càlcul: 666
-002
resultat: 664 rw- rw- r--

Exemple-2:
donada la màscara 022
càlcul: 666
-022
resultat: 644 rw- r-- r--

Exemple-3:
donada la màscara 026
càlcul: 666
-026
resultat: 640 rw- r-- ---

Com es pot observar en el cas dels permisos per defecte de fitxers:

- la màscara es resta del valor 666
- els valors més típics de la màscara són 0, 2 i 6 però en realitat es pot usar qualsevol valor octal.
- usualment no interessa que els fitxers tinguin activat el permís x d'execució per defecte.

**Màscara: càlcul per a directoris**

*7 7 7*
*-umask*
---
permisos-de-directori-per defecte

Per calcular els permisos per defecte dels directoris es resta la màscara del valor simbòlic ***777***. El resultat són els permisos de fitxers.

Exemple-1:
donada la màscara 002
Càlcul: 777
-002
resultat: 775 rwx rwx r-x

Exemple-2:
donada la màscara 022

Càlcul: 777
-022
resultat: 755 rwx r-x r-x

Exemple-3:

donada la màscara 027
càlcul: 777
-027
resultat: 750 rwx r-x ---

Com es pot observar en el cas dels permisos per defecte de directoris:

- la màscara es resta del valor 777.
- els valors més típics de la màscara són 0, 2 i 7 però en realitat es pot usar qualsevol valor octal.
- usualment interessa que els directoris tinguin activat el permís x per permetre l'accés/llistat del directori.

Cada usuari es pot canviar la mascara amb l'ordre umask. Cada vegada que es crea un fitxer o directori se li apliquen els permisos segons correspongui d'aplicar el càlcul amb la umask que té assignada l'usuari en aquell moment.

**$ umask**
0002

**$ ls -l calendar.pdf**
-rw-rw-r-- 1 ecanet ecanet 174 19 des 20:20 calendar.pdf

**$ ls -ld bdades/**
drwxrwxr-x 2 ecanet ecanet 40 19 des 19:57 bdades/

**$ umask 0022**

**$ cal > calendar.pdf**
$ ls -l calendar.pdf
-rw-r--r-- 1 ecanet ecanet 174 19 des 20:22 calendar.pdf

**$ mkdir bdades2**
$ ls -ld bdades2
drwxr-xr-x 2 ecanet ecanet 40 19 des 20:22 bdades2

**$ umask 0027**

**$ cal > calendar3.pdf**
$ ls -l calendar3.pdf
-rw-r----- 1 ecanet ecanet 174 19 des 20:25 calendar3.pdf

**$ mkdir bdades3**
$ ls -ld bdades3
drwxr-x--- 2 ecanet ecanet 40 19 des 20:23 bdades3

# Canvi de propietari i/o grup

## Canvi de propietari / grup amb chown

Hem vist fins ara que tot element del sistema de fitxers pertany a un usuari i a un grup. Identifiquem el propietari o owner en els llistats llargs com el valor de la tercera columa. El propietari és qui ha creat el fitxer o directori.

L'ordre que permet modificar el propietari d'un element és:

*chown user file|dir* on user és el nom del nou propietari del file o dir

Es pot modificar el propietari d'un fitxer? Per exemple l'usuari ecanet pot passar la propietat del fitxer calendari.txt a l'usuari guest?

```
$ ls -l calendar.pdf
-rw-rw-r-- 1 ecanet ecanet 174 19 des 20:20 calendar.pdf

$ chown guest calendar.pdf
chown: changing ownership of 'calendar.pdf': Operation not permitted
```

El sistema operatiu no ha permès a l'usuari ecanet canviar la propietat del fitxer calendar.pdf que és seu i assignar-li la propietat a guest. Perquè? ben senzill, un usuari no pot assignar a un altre la propietat d'un element.

Pensem en això en el món real, si tinc deutes o he de pagar una hipoteca faig un chown de la hipoteca i li passo el deute a un altre usuari (jo em quedo el pis).

Només un usuari del sistema està autoritzat a poder canviar la propietat dels elements del sistema de fitxers, ***root***. El superusuari root si pot canviar la propietat dels fitxers i directoris assignant-los a altres usuaris.

```
[user01@a36 m01]$ su -
Password:

[root@a36 ~]# cd /tmp/m01/

[root@a36 m01]# chown guest calendar.pdf

[root@a36 m01]# ls -l calendar.pdf
-rw-rw-r-- 1 guest ecanet 174 Dec 19 20:20 calendar.pdf
```

```
[root@a36 m01]# chown guest bdades

[root@a36 m01]# ls -ld bdades
drwxrwxr-x 2 guest ecanet 40 Dec 19 19:57 bdades
```

- observeu que en quests dos exemples s'ha canviat la propietat del fitxer i del directori, però no el grup. Tots dos elements continuen pertanyen al grup ecanet.

De fet l'ordre chown permet modificar no només el propietari sinó també el grup d'un element. Es pot assignar tot de cop (propietari i grup), només el propietari o només el grup.

El format complert de l'ordre és:

*chown user.group file|dir*
*chown user:group file|dir*

Observeu que com a separador d'usuari i grup s'accepta tant el punt com els dos punts. Si només s'indica un element és el propietri. si el propietari no s'indica llavors cal indicar el separador i el nom del grup.

| |
|---|
| **[root@a36 m01]# chown guest.users calendar.pdf**<br><br>[root@a36 m01]# ls -l calendar.pdf<br>-rw-rw-r-- 1 guest users 174 Dec 19 20:20 calendar.pdf |
| **[root@a36 m01]# chown user01:guest calendar.pdf**<br><br>[root@a36 m01]# ls -l calendar.pdf<br>-rw-rw-r-- 1 user01 guest 174 Dec 19 20:20 calendar.pdf |
| **[root@a36 m01]# chown .users calendar.pdf**<br><br>[root@a36 m01]# ls -l calendar.pdf<br>-rw-rw-r-- 1 user01 users 174 Dec 19 20:20 calendar.pdf |
| **[root@a36 m01]# chown :ecanet calendar.pdf**<br><br>[root@a36 m01]# ls -l calendar.pdf<br>-rw-rw-r-- 1 user01 ecanet 174 Dec 19 20:20 calendar.pdf |
| **[root@a36 m01]# chown guest calendar.pdf**<br><br>[root@a36 m01]# ls -l calendar.pdf<br>-rw-rw-r-- 1 guest ecanet 174 Dec 19 20:20 calendar.pdf |

- el primer exemple assigna al fitxer el propietari guest i el grup users usant el punt com a separador.
- el segon exemple assigna el propietari user01 i el grup guest usant els dos punts com a separador.
- el tercer exemple assigna únicament el grup users usant el punt com a separador. En estar buit la part de propietari aquest no es modifica.
- el quart exemple assina al fitxer al grup ecanet usant els dos punts de separador (i deixant en blanc la part de propietari que es maté tal i com estava).
- l'últim exemple assigna només el propietari que passa a ser guest.

## Canvi de grup amb chgrp

Es pot canviar la pertinença d'un fitxer o directori a un grup amb les ordres chown i chgrp. Tal i com indica el nom l'ordre chgrp serveix per fer change group. Sabem que root pot canviar-ho tot, però pot un usuari canviar de grup un dels seus fitxers o directoris?

Parlem-ne. Jo sóc soci d'un videoclub (és una cosa del segle passat…) i ara em canvio de grup i em faig soci del congrés de diputats (piscina, bar, etc…). Soc soci del club esportiu del meu poble i em canvio de grup per la cara i passo a ser soci del barça. Bé de fet faig un altre canvi de grup i passo a ser membre de la llotja del barça. Puc?. Posats a fer canvio el meu grup assignat de professor a membre de la casa reial espanyola d'Abudabi. Puc?

Sabem que els usuaris poden pertànyer a diversos grups (d'usuaris, de la classe, d'un projecte, de la biblioteca, etc). Un és el grup principal i es pot pertànyer a cap o múltiples grups secundaris.

Un usuari pot canviar un element seu de grup passant-lo d'un grup al que pertany a un altre grup al que pertany. Evidentment no pot assignar un element a un grup al que no pertany. Root sempre pot fer el que vulgui (com el de Abu Dabi!).

```
[root@a36 ~]# usermod -G users guest
[root@a36 ~]# id guest
uid=1000(guest) gid=1000(guest) groups=1000(guest),100(users)
```

```
[root@a36 ~]# groupadd vclub
[root@a36 ~]# usermod -G users,vclub user01
[root@a36 ~]# id user01
uid=1029(user01) gid=1001(ecanet) groups=1001(ecanet),100(users),1032(vclub)
```

- primerament se li assigna l'usuari guest al grup secundari users.
- després es crea el grup vclub i s'assigna l'usuari user01 als grups users i vclub com a grups secundaris.

```
[guest@a36 m01]$ whoami > /tmp/identitat.txt

[guest@a36 m01]$ ls -l /tmp/identitat.txt
-rw-rw-r-- 1 guest guest 6 Dec 19 21:05 /tmp/identitat.txt
```

```
[guest@a36 m01]$ chgrp users /tmp/identitaat.txt
chgrp: changing group of '/tmp/identitaat.txt': Operation not permitted
**cal tancar sessió i iniciar-la de nou perquè guest reconegui els nous grups als que pertany**
```

```
[guest@a36 ~]$ chgrp users /tmp/identitaat.txt
[guest@a36 ~]$ ls -l /tmp/identitaat.txt
-rw-rw-r-- 1 guest users 6 Dec 19 21:05 /tmp/identitaat.txt
```

```
[guest@a36 ~]$ chgrp vclub /tmp/identitaat.txt
chgrp: changing group of '/tmp/identitaat.txt': Operation not permitted
```

- primerment l'usuari guest cres un fitxer anomenat identitat.txt que és propietat seva i pertany al grup guest.
- a continuació s'intenta canviar el grup del fitxer però com que la sessió de guest ja estava oberta no 'sap' que ara pertany també al grup users. Cal tancar i obir de nou la sessió de guest perquè es carreguin els canvis.
- després l'usuari guest pot assignar al fitxer identitat.txt que pertanyi al grup users, que és un grup al que ell també pertany.
- finalment l'usuari guest intenta assignar el fitxer al grup ecanet però ni pot perquè ell no pertany a aquest grup.

# 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. Mostrar la umask del vostre usuari. Crear un fiter i un directori i verificar que els permisos assignats són els corresponents amb el càlcul de la umask.

```
$ umask
0002

$ date > date.txt
$ ls -l date.txt
-rw-rw-r-- 1 ecanet ecanet 28 19 des 21:19 date.txt

$ mkdir dir1
$ ls -ld dir1/
drwxrwxr-x 2 ecanet ecanet 40 19 des 21:19 dir1/
```

   - si la màscara de permisos és 002 els permisos per a fitxers són (restem de 666) 664 rw- rw- r--.
   - en el cas dels directoris restem de 777 i els permisos són 775 rwx rwx r-x.

3. Mostra la umask de l'usuari user01 (si no existeix el crees). Crear un fitxer i un directori dins de tmp i verificar que els permisos corresponen a la umask.

```
[user01@a36 m01]$ umask
0022

[user01@a36 m01]$ cal > /tmp/cal.txt
[user01@a36 m01]$ ls -l /tmp/cal.txt
-rw-r--r-- 1 user01 ecanet 174 Dec 19 21:24 /tmp/cal.txt

[user01@a36 m01]$ mkdir /tmp/dir2
[user01@a36 m01]$ ls -ld /tmp/dir2/
drwxr-xr-x 2 user01 ecanet 40 Dec 19 21:24 /tmp/dir2/
```

- en aquest cas la màscara és 022 i els permisos de fitxers són doncs 644.
- els permisos per defecte de directori són 755.

4. Modifica la màscara del teu usuari i estableix que permeti a fitxers i directoris control total per al propietari, lectura per al grup i res per a altres.

```
$ umask 0027

$ uptime > uptime.txt
$ ls -l uptime.txt
-rw-r----- 1 ecanet ecanet 68 19 des 21:27 uptime.txt

$ mkdir dir3
$ ls -ld dir3/
drwxr-x--- 2 ecanet ecanet 40 19 des 21:27 dir3/
```

- perquè els permisos de fitxers i directoris siguin els correctes la màscara ha de ser 0027.
- en fitxers 666 - 027 correspon als permisos 640.
- en directoris 777 - 027 correspon als permisos 750.

5. Amb el teu usuari crea un fitxer anomenat meu.pdf i traspassa la propietat del fitxer a guest.

```
$ id > meu.pdf

$ chown guest meu.pdf
chown: changing ownership of 'meu.pdf': Operation not permitted
```

- només root pot modificar el propietari d'un element

```
$ su -
Password:

[root@a36 ~]# cd /tmp/m01/

[root@a36 m01]# chown guest meu.pdf
[root@a36 m01]# ls -l meu.pdf
-rw-r----- 1 guest ecanet 89 Dec 19 21:31 meu.pdf
```

6. Assigna l'usuari guest al grup users com a grup secundari. Crea un fitxer amb l'usuari guest i modifica el grup al que pertany el fitxer assignant-lo al grup users.

```
[root@a36 m01]# usermod -G users guest
```

```
[root@a36 m01]# id guest
uid=1000(guest) gid=1000(guest) groups=1000(guest),100(users)
```

```
[guest@a36 ~]$ ps > procs.txt
[guest@a36 ~]$ ls -l procs.txt
-rw-rw-r-- 1 guest guest 84 Dec 19 21:35 procs.txt

[guest@a36 ~]$ chgrp users procs.txt
[guest@a36 ~]$ ls -l procs.txt
-rw-rw-r-- 1 guest users 84 Dec 19 21:35 procs.txt
```

7. Fes que l'usuari guest usant l'ordre chown torni a posar el fitxer de l'exercici anterior al grup per defecte de l'usuari.

```
[guest@a36 ~]$ chown .guest procs.txt
[guest@a36 ~]$ ls -l procs.txt
-rw-rw-r-- 1 guest guest 84 Dec 19 21:35 procs.txt
```