M1 – Implantació de Sistemes Operatius

Unitat Didàctica 4

Administració i assegurament de la informació

Raül Sala / José Luis Antúnez – 2017/2018

Activitat 3

Treball amb fitxers

Propietat i permisos, marques de temps, compactació i compressió, enllaços forts i dèbils, indexació i cerca, quotes.

Entitats i modes

- Es consideren **tres tipus d'usuaris** (entitats), respecte el fitxer:
  - **Propietari (u):** per defecte, quan crea un fitxer, se li assigna.
  - **Grup propietari (g):** per defecte, quan un usuari crea un fitxer, se li assigna al seu grup principal.
  - **La resta d'usuaris (o):** qualevol usuari del sistema, "world permissions".

- Per a cadascun d'ells, es consideren **tres tipus d'operacions** (modes) → **associats al concepte d'inode!**

	Arxiu	Directori
Lectura (`read - r`)	**llegir el contingut de l'arxiu**	**llistar les entrades del directori**
Escriptura (`write - w`)	**modificar el contingut de l'arxiu**	**crear fitxers**
Execució (`exec - x`)	**executar l'arxiu**	**accedir al directori**

El superusuari

- **Superpoders!!!**
  - Les restriccions d'accés no s'apliquen al usuari root.
  - Aquest usuari pot llegir o modificar qualsevol fitxer, fins i tot aquells als que no té permisos, per exemple un fitxer amb permisos 000.

- **“Excepcions”**
  - Es possible que el superusuari no pugui modificar un sistema de fitxers muntat com a només lectura. Cal però, fer notar que ***root podria tornar a muntar el sistema amb uns altres permisos***
  - Fins i tot el superusuari necessita també el bit d'execució per executar un programa (el pot assignar si cal)

Consulta de permisos

```
$ ls -l /bin
total 6216
-rwxr-xr-x 1 root root  729040 2009-03-02 15:22 bash
-rwxr-xr-x 1 root root   30140 2008-11-10 12:51 bunzip2
DUSRGRPALT   USER GRUP
```

- Es presenten de la següent manera (drwxrwxrwx):
  - **Primer caràcter:** indica si és un directori o no.
  - **Primer bloc:** indica els permisos del propietari (u).
  - **Segon bloc:** indica els permisos del grup (g).
  - **Tercer bloc:** indica els permisos de la resta d'usuaris (o).

- Per a cada bloc s'indica si hi ha permisos del tipus indicat (lletra: **r**, **w**, **x**) o no (guió **-**).

Notació

Notació simbòlica	Notació binària	Notació octal
`---`	`000`	`0`
`--x`	`001`	`1`
`-w-`	`010`	`2`
`-wx`	`011`	`3`
`r--`	`100`	`4`
`r-x`	`101`	`5`
`rw-`	`110`	`6`
`rwx`	`111`	`7`

- **Exemples:**
  - **`755`** (**`rwxr-xr-x`**): El propietari ho pot fer tot i la resta llegir i executar. ***Permís per defecte de les carpetes***.
  - **`644`** (**`rw-r--r--`**): El propietari pot llegir i modificar i la resta només llegir. Permís per defecte dels fitxers.
  - **`777`** / **`000`** (**`rwxrwxrwx`** **`---------`**): cap/tots els permisos.

Canviar permisos (**chmod**)

```
$ chmod [-R] <permisos> <fitxer o fitxers>
```

- **Permet modificar els permisos d'un fitxer:**
  - Suporta tant la notació octal com la simbòlica
  - Només el propietari del fitxer o el superusuari poden executar-la.
    - El propietari no pot assignar el fitxer a un altre usuari o canviar-lo de grup si no pertany al grup destí.
  - Notació dels permisos:
    - **r** (read), **w** (write) i **x** (execution) per indicar els permisos.
    - El simbol **-** indica que el permís d'aquella posició no està activat.
    - **u** (user), **g** (group) i **o** (others) per indicar les entitats afectades.
    - El caràcter **a** equival a **ugo** i representa les tres entitats.
  - Afegir o treure permisos:
    - **+/-**: indica que s'ha d'activar/desactivar el permís.
    - **=**: permet definir els tres permisos tal com s'indiquin.
    - Sintaxi: **<entitat(s)>[+-=] <permís(os)>**

Exemples

- **`chmod 777 fitxer ≡ chmod a=rwx fitxer`**
  - Tots els permisos a tots els usuaris.
- **`chmod 644 fitxer`**
  - Permís de lectura i escriptura a l'usuari, lectura al grup i a la resta.
- **`chmod u+x fitxer`**
  - Afegeix el permís d'execució a l'usuari.
- **`chmod g-w fitxer`**
  - Treu el permís d'escriptura al grup.
- **`chmod o-rw fitxer`**
  - Treu els permisos de lectura i escriptura a la resta.
- **`chmod ug-x fitxer`**
  - Treu el permís d'execució a l'usuari i el grup.
- **`chmod u=rw,go= fitxer`**
  - Estableix el permís d'usuari a lectura i escriptura (no execució). La resta d'usuaris no tindran permisos.

Canviar el propietari/grup (**chown** i **chgrp**)

```
$ chown [OPTION]... [OWNER][:[GROUP]] FILE...
```

- **Permet modificar el propietari d'un fitxer:**
  - Només pot canviar el propietari d'un fitxer el superusuari del sistema.
    - El propietari d'un fitxer pot canviar el grup d'un fitxer sempre que pertanyi al grup destí.

```
$ chown noupropietari:nougrup fitxer //propietari i grup
$ chown noupropietari fitxer //només propietari
$ chown :nougrup fitxer //només grup
```

```
$ chgrp [-R] <grup> <arxiu ...>
```

- **Permet canviar només el grup:**
  - El propietari pot cedir l'arxiu a qualsevol grup al qual pertanyi

```
$ chgrp nougrup fitxer //equivalent a chown :nougrup fitxer
```

Permisos especials: **SUID**

- **SUID (Set User ID):**
  - S'utilitza **conjuntament amb fitxers amb permisos d'execució**
  - Indica al sistema operatiu que el programa **s'ha d'executar amb els permisos del propietari del fitxer** i no pas amb els permisos de l'usuari que executa el fitxer.
  - Es pot utilitzar per tal d'**executar fitxers com a superusuari sense ser root**. El fitxer ha de pertànyer al superusuari, ser executable i tenir el bit especial d'execució SUID.
    - Aquesta opció es força utilitzada en alguns serveis i programes
    - Els programes que s'executen d'aquesta forma són anomenats (SUID root, *Set User ID root*).
    - No funciona amb directoris, nomes amb executables.
  - Els fitxers amb aquest permís s'indiquen amb una **s** al bit d'execució del propietari. (o S majúscula si no és executable i té el bit SUID marcat!!)

```
$ ls -l /bin/ping
-rwsr-xr-x 1 root root 30856 2007-12-10 18:33 /bin/ping
```

Permisos especials: **SGID**

- **SGID (Set Group ID),** trobem dos casos**:**
  - ***Per a Fitxers:*** És similar a SUID però estableix que el grup del programa executable és el grup del fitxer i no pas el grup de l'usuari que executa el fitxer. S'indica amb una **s** al bit d'execució del grup**.**

```
-rwxr-sr-x 1 root root 30856 2007-12-10 18:33 /bin/foo.sh
```

  - ***Per a directoris:*** Quan el bit SGID s'estableix en un directori, els fitxers nous o directoris creats en aquest directori heretaran el grup del directori i no pas el grup de l'usuari que crea el directori o fitxer.

- **Tant el SUID com el SGID són bits potencialment perillosos:**
  - només s'han d'utilitzar en casos necessaris i amb un bon control de la situació.

Permisos especials: **Sticky bit**

- **Sticky bit (bit enganxifós)**
  - Ha tingut diferents significats durant la història de Unix.
  - S'utilitza per **evitar que certs fitxers siguin esborrats per persones que no siguin propietàries del fitxer**.
  - Quan aquest bit està present en un directori, els fitxers del directori només poden ser eliminats pels seus propietaris o pel superusuari.
  - S'indica amb una **t** al bit d'execució de l'entitat altres. (o T majúscula si el directori no és executable per als ALTRES i té el bit Sticky marcat!!)
  - **S'ignora en els fitxers**.

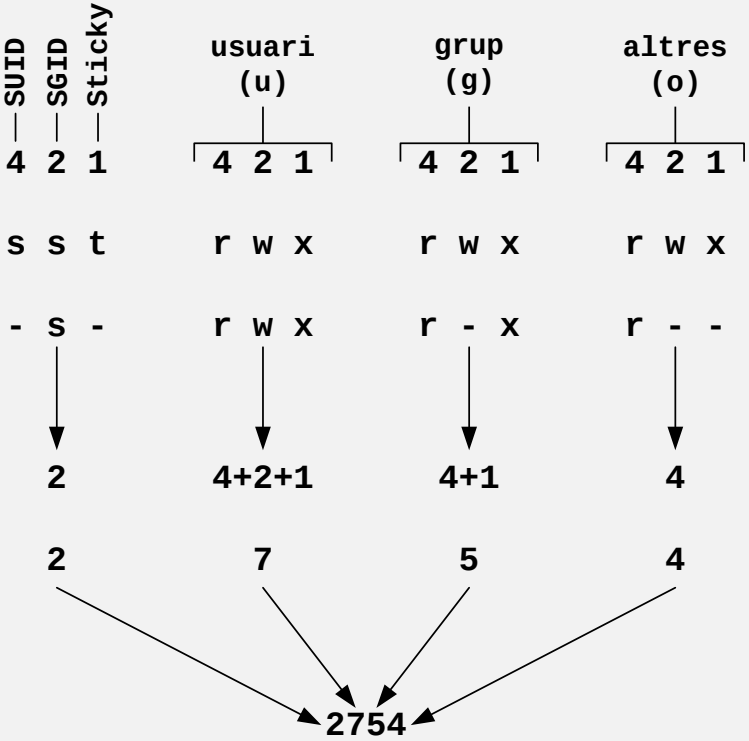
- **Directori `/tmp`:**
  - **Exemple típic d'Sticky bit:** permet que qualsevol hi escrigui, però cal evitar que s'esborrin fitxers d'altres usuaris

```
$ ls -l /
drwxrwxrwt  13 root root  4096 2009-12-27 18:31 tmp
```

Activació de bits especials

- **Activació:**
  - **SUID:** La lletra **s** i la entitat **u**.
  - **SGID:** La lletra **s** i la entitat **g**.
  - **Sticky bit:** La lletra **t** i la entitat.

- **Exemples:**
  - **u+s**: afegeix (**+**) el bit SUID (**s** i **u**).
  - **g+s**: afegeix (**+**) el bit SGID (**s** i **g**).
  - **o+t**: afegeix (**+**) el bit Sticky (**t** i **o**)
  - Cal que tinguin permis d'execució també (**x**) → **si no, apareixerà la lletra en majúscula.**



`umask` (1)

- **Concepte de màscara: permisos màxims assignables.**
  Coneixem **2 formes de definir-la**, en octal i simbòlicament.

  - **En octal:** permisos que no es podran assignar.
    - **Directoris:** valor de referència **`777`**.
      - Màscara **`022`** → permisos màxims = **`755 (rwxr--r--)`**
    - **Fitxers:** valor de referència **`666`** (no s'assigna permís d'execució).
      - Màscara **`022`** → permisos màxims = **`644 (rw-r--r--)`**

  - **Simbòlicament:** permisos màxims assignables.
    - **Fitxers i directoris:** se'ls assignarà el valor indicat, sense valor de referència.
      - Màscara **`u=rw,g=r,o=r`** → permisos màxims = **`644 (rw-r--r--)`**
      - **LA MÀSCARA MAI ASSIGNARÀ PERMISOS D'EXECUCIÓ A FITXERS.**

umask (2)

- **Exemple:**

```
$ umask //mostrem la màscara actual en octal
0022
$ umask –S //la mostrem simbòlicament
u=rwx,g=rx,o=rx
$ mkdir hola; echo “hola” > hola.txt
$ ls –l
drwxr-xr-x 2 estudiant estudiant 4096 2009-12-29 09:00 hola
-rw-r--r-- 1 estudiant estudiant    5 2009-12-29 09:00 hola.txt
$ umask 027 //l’establim en octal
$ mkdir adeu; echo “adeu” > adeu.txt
$ ls –l
drwxr-x--- 2 estudiant estudiant 4096 2009-12-29 09:03 adeu
-rw-r----- 1 estudiant estudiant    5 2009-12-29 09:03 adeu.txt
$ umask u=rwx,g=rwx,o=rwx //l’establim simbòlicament
$ mkdir nova; echo “nova” > nova.txt
$ ls –l
drwxrwxrwx 2 estudiant estudiant 4096 2009-12-29 09:05 nova
-rw-rw-rw- 1 estudiant estudiant    5 2009-12-29 09:05 nova.txt
```

umask (3)

- El valor de la màscara s'aplica només durant la durada de la sessió i per a l'usuari que ha modificat la màscara.

- Es torna a restaurar el valor per defecte en finalitzar una sessió o obrir una nova terminal.
  - Per a poder modificar la màscara per defecte caldrà indicar-ho al fitxer de configuració **`.bashrc`**.

- Tot els processos executats des de una terminal a la que se li ha canviat el valor de la màscara, hereten la nova màscara.

- Si el valor de la màscara supera el valor per defecte (en el cas dels fitxers) el resultat és 0.
  - Per exemple, una màscara **`027`** si es resta al valor per defecte **`666`** queda com a resultat **`640`** i no pas **`64-1`**.

Marques de temps

- **Temps de l'últim accés (`atime`)**
  - Marca l'últim cop que el fitxer ha estat llegit.
- **Temps de l'última modificació (`mtime`)**
  - Marca l'últim cop que els continguts del fitxer van ser modificats.
- **Temps de canvi (`ctime`)**
  - Marca l'últim cop que els permisos del fitxer van ser modificats.

- `touch` canvia **atime** i **mtime** (o crea el fitxer, si no existeix).
- Es poden consultar amb `ls` o `stat`.

```
$ touch noufitxer.txt
$ stat noutfitxer.txt //consulteu les dates
$ chmod 777 noufitxer.txt //canviem permisos → ctime
$ cat noufitxer.txt //accedim al fitxer → atime
$ vi noufitxer.txt //modifiquem el fitxer → mtime
$ stat noufitxer.txt //vegeu com les dates han canviat
```

Empaquetar fitxers (1)

- L'eina més utilitzada en sistemes lliures és **tar** (Tape Archiver).
  - Eina molt antiga → s'utilitzava per a copiar en cintes.
    - És per això que cal indicar el paràmetre –f (file).
  - No comprimeix: només **compacta** un arbre de directoris complet.

Comada		Descripció
`--create`	`c`	Crea un arxiu tar.
`--concatenate`	`A`	Afegeix fitxers tar a un fitxer.
`--append`	`r`	Afegeix fitxers que no són tar a un arxiu tar.
`--update`	`u`	Afegeix només els fitxers que són nous a un arxiu.
`-diff`	`d`	Compara un arxiu amb els fitxers del disc.
`--list`	`t`	Mostra els continguts d'un arxiu.
`--extract`	`x`	Extreu els fitxers d'un arxiu.
`--same-permisions`	`p`	Manté els permisos originals al fitxer.

Empaquetar fitxers (2)

- **Empaquetar**
  - On fitxers pot ser una carpeta :

```
$ tar cf fitxer.tar fitxers
$ tar cvf fitxer.tar directori
```

  - Per defecte ordre silenciosa. -v (--verbose) mostra el que fa l'ordre
  - També podem indicar una llista de fitxers (separada per espais) o utilitzar comodins (*)
  - Si el paquet ja existeix, es sobreescriu.

- **Desempaquetar**
  - Si els fitxers ja existeixen, se sobreescriuen

```
$ tar –xvf fitxer.tar
```

Comprimir fitxers

- **Gzip (.gz):** GNU Zip. Algoritme DEFLATE (LZ77 i el Huffman) algorismes lliures sense patents.
  - `gzip`/`gunzip`

- **Bzip2 (.bz2):** Posterior a gzip. Comprimeix la majoria d'arxius de forma més efectiva que els compressors tradicionals gzip o ZIP, però és més lent.
  - `bzip2`/`bunzip2`

- **Cap dels dos sistemes empaqueta (ja tenim tar per a això).**

- Altres sistemes: **.zip** (`zip`/`unzip`); **.z** (`compress`/`uncompress`).

Compactació i compressió fitxers

- **.tar.gz**
  - Empaquetar: `tar -czf fitxers.tar.gz fitxers…`
  - Desempaquetar: `tar -xvzf fitxercomprimit.tar.gz`
  - Llistar: `tar -ztvf file.tar.gz`
  - Només comprimir: `gzip fitxers,…`
  - Només descomprimir: `gunzip file.tar.gz`

- **.tar.bz2**
  - Empaquetar: `tar -cvjf tarfile.tar.bz2 fitxers…`
  - Desempaquetar: `tar -jxvf fitxercomprimit.tar.bz2`
  - Llistar: `tar -jtvf file.tar.bz2`
  - Només comprimir: `bzip2 fitxers,…`
  - Només descomprimir: `bunzip2 file.tar.bz2`

Enllaços

- Permeten que un sol fitxer tingui múltiples camins en un sistema de fitxers.

- El fitxer **només existeix un cop** i **només ocupa l'espai un cop**, però pot tenir múltiples camins.
  - Són similars (però més potents) als enllaços directes de Windows o els alies de MAC OS.

- **Útils per a:**
  - Assignar noms (camins) més simples a fitxers amb noms complicats.
  - Assignar diferents noms a un mateix fitxer
  - Algunes ordres que semblen ordres diferents són realment enllaços que porten al mateix executable (per exemple les ordres `fsck`).

Enllaços a Linux: enllaços simbòlics

- **Enllaços simbòlics o symbolic links:**
  - Es creem amb `ln -s` o `--symbolic`.
  - Si s'esborra l'enllaç, no s'esborra el fitxer original.
  - Si s'esborra el fitxer apuntat, aleshores es trenca l'enllaç.

```
$ mkdir prova
$ ln –s prova p
$ ls –l
lrwxrwxrwx 1 estudiant estudiant       5 2009-12-27 23:11 p -> prova
drwxr-xr-x 2 estudiant estudiant    4096 2009-12-27 23:10 prova
```

  - **Els enllaços simbòlics apareixen a ls amb tots els permisos:**
    - Un soft link mai permetrà accedir a un fitxer al que no teníem accessos.
    - Sempre podem crear un enllaç simbòlic a un fitxer que ens sigui visible en una carpeta en la que tinguem permisos d'escriptura.
    - Les operacions de lectura i de modificació de l'enllaç són operacions de lectura i modificació al fitxer enllaçat i s'apliquen les restriccions d'aquest.
    - Només pot eliminar un enllaç el propietari de l'enllaç.

Enllaços a Linux: enllaços forts (1)

- **Enllaços forts o hardlinks:**
  - ln, per defecte, crea enllaços forts.
  - Dos noms de fitxer que apunten al mateix inode.
  - Tots dos són igual de vàlids i cap d'ells té més importància que l'altre.
  - Per a esborrar el fitxer (inode) del disc cal esborrar tots els hardlinks.

```
$ echo "Sistemes Operatius" > so.txt
$ ln so.txt sisops
$ ls -li so.txt sisops
126980 -rw-r--r-- 2 estudiant estudiant 19 2009-12-27 23:18 sisops
126980 -rw-r--r-- 2 estudiant estudiant 19 2009-12-27 23:18 so.txt

$ stat so.txt
...
Device: 801h/2049d        Inode: 126980      Links: 2
...
$ stat sisops
...
Device: 801h/2049d        Inode: 126980      Links: 2
...
```

Enllaços a Linux: enllaços forts (2)

- **Un hardlink no és una copia d'un fitxer?**
  - No és exactament el mateix
  - Els continguts dels fitxer només són emmagatzemats un cop
  - Si es modifica qualsevol dels enllaços durs aleshores es modifiquen tots dos els fitxers.
  - Si es canvia qualsevol de les metadades dels enllaços durs (permisos, propietaris, marques de temps) es modifiquen a tots els fitxers
  - Si s'esborren els continguts d'un dels fitxers s'esborrem el de tots, o si es sobreescriu o s'edita un fitxer, també afecta a tots.

Enllaços forts VS Enllaços simbòlics

- Esborrar un enllaç simbòlic no esborra **mai** el fitxer original.
  - Un enllaç dur només esborra els continguts del fitxer si és l'últim enllaç dur al fitxer.
- Els enllaços simbòlics són una mica més lents en accés que els enllaços durs.
- Els enllaços durs no poden apuntar entre fitxers de diferents sistemes de fitxers.
- Esborrar el fitxer enllaçat per un enllaç simbòlic, deixa l'enllaç simbòlic trencat.
  - Esborrar un dels enllaços durs no afecta a la resta d'enllaços (només decreix el comptador link del inode).
- No es poden crear enllaços durs de directoris.
  - però si enllaços simbòlics.

Cerca de fitxers i directoris

- **Cerca de fitxers:**
  - **Ordre `find`:** busca directament als dispositius del sistema.
  - **Ordre `locate`:** busca en un index (base de dades).

- **Eines per a buscar ordres:**
  - **`which`:** busca executables dins del **`$PATH`**.
  - **`whereis`:** busca executables, manuals i codi font.
  - **`type`:** mostra que s'executarà realment al escriure un text a l'interpret d'ordres.

Buscar fitxers: `find`

- **Permet buscar fitxers:**
  - La cerca és fa en el moment d'utilitzar la comanda.
  - Més lenta, ja que no utilitza índex, com la comanda locate.
  - Disposa de **moltíssimes** opcions.

- **Exemples:**
  - `find / -name nom_fitxer`: fitxers del sistema, per nom.
  - `find /home -type f -name nom_fitxer`: només fitxers, per nom.
  - `find /home -type d -name nom_fitxer`: només directoris, per nom.
  - `find / -maxdepth 2 -type d`: directoris I subdirectoris de l'arrel.
  - `find . -perm 664`: fitxers i directoris amb permisos 644.
  - `find / -user usuari`: fitxers de l'usuari.
  - `find . -type f -exec file {} \;`: executar file per a cada fitxer trobat. `{}` és la ruta de cadascun dels fitxers trobats.

Buscar fitxers: `locate`

- **Permet buscar fitxers:**
  - Utilitza una base de dades (la cerca no es fa al moment).
  - És més ràpida que **`find`**.
  - Accepta comodins i expressions regulars.
  - **`locate nomfitxer`**

- **Fitxers:**
  - Base de dades: **`/var/lib/mlocate/mlocate.db`**
    - S'actualitza amb la comanda **`updatedb`** → s'autoexecuta diàriament.
    - Podem executar a mà: **`sudo updatedb &`**
  - Cron: tasca programada que executa updatedb diàriament (6.25h).
  - Comandes: **`updatedb`** (actualitza db) **`locate`** (cercador).
  - Configuració: **`/etc/updatedb.conf`**
    - Què s'indexa i què no, fins on s'arriba,...

Inconvenients de l'ús de `locate`

- **Espai:** la base de dades ocupa un espai de disc.
  - La mida de la base de dades no es gaire gran i en la majoria de sistemes actuals aquest inconvenient no és gaire important
- **Sincronització:** la base de dades no sempre està al dia.
  - Cada vegada que hi ha una modificació al sistema de fitxers, hauríem de tornar a executar updatedb per tal de mantenir la base de dades sincronitzada.
- **Es necessiten permisos de superusuari per modificar la base de dades.**

Comparació de fitxers

- La comanda **`diff`** permet fer comparacions de fitxers:

```
$ diff fitxer1 fitxer2
1c1
< Prova de comparació
---
> Prova de compativa
```

- Si els arxius són idèntics, no mostra res.
  - Si no, mostra una comparativa entre els dos fitxers, on < marca les línies del primer fitxer i > les del segon.

- Disposa de moltes opcions que permeten personalitzar la nostra comparació.

- Combinat amb la comanda **`patch`** s'utilitza per a generar pegats de programes en Linux.

Bibliografia i recursos utilitzats

- Tur, Sergi (2009). *Apunts del curs Linux Professional Institute Certificate 1 (LPIC – 1). Examen 101.*
  http://acacha.org/mediawiki/index.php/LPI_103.3
  http://acacha.org/mediawiki/index.php/LPI_104.5
  http://acacha.org/mediawiki/index.php/LPI_104.7
- Morancho, Enric (2006). *Unix – Crides al sistema i comandes*. Edicions UPC.
- Smith, Roderich W. (2006). *LPIC 1: Linux Professional Institute Certification*. Sybex.