

# Sistemas de Ficheros

Escuela Técnica Superior de Ingeniería de Telecomunicación  
Universidad Rey Juan Carlos

gsyc-profes (arroba) gsync.urjc.es

Diciembre de 2020



©2020 GSyC  
Algunos derechos reservados.  
Este trabajo se distribuye bajo la licencia  
Creative Commons Attribution Share-Alike 4.0

- 1 Introducción
- 2 FHS Filesystem Hierarchy Standard
  - Directorios de usuarios
  - Programas y mandatos
  - Configuración del sistema
  - El Hardware
  - Documentación
  - Ficheros Temporales
  - Otros directorios relacionados con el S.O.
  - Puntos de Montaje
- 3 Montaje de sistemas de ficheros
  - mount, df, lsblk
  - /etc/fstab
  - Tipos de sistemas de ficheros
  - Sistemas de Ficheros en Espacio de usuario
  - sshfs
  - upstart

# Introducción

- Un sistema de ficheros es una forma de almacenar y organizar ficheros para permitir su uso
- Pueden usar un dispositivo de almacenamiento (disco, cdrom), la red o ser sólo un interfaz para acceder a datos
- Para poder empezar a almacenar información en un sistema de ficheros, éste tiene que ser *inicializado*
- En Unix, para poder usarlo, hay que *montarlo* en alguna parte de la jerarquía de directorios, un árbol cuya raíz es el directorio llamado / .

*On a UNIX system, everything is a file; if something is not a file, it is a process*

Los ficheros pueden ser

- Ficheros normales
- Directorios
- Ficheros especiales (Entrada y salida. Están en */dev* )
- Enlaces
- Fifos. (Pipes con nombre). Para comunicación entre procesos
- Sockets de dominio. Similares a los sockets TCP/IP

El primer caracter de `ls -l` representa:

-	Regular file
d	Directory
c	Special file
l	Link
p	Named pipe
s	Socket
b	Block device

# Jerarquía del Sistema de Ficheros

Para quien se acerca a Linux resulta confuso un `ls -l /`

```
drwxr-xr-x    2 root    root          4096 ene 30 20:34 bin
drwxr-xr-x    2 root    root          4096 mar 12 19:46 boot
drwxr-xr-x    5 root    root        24576 may 22 06:27 dev
drwxr-xr-x   66 root    root          4096 may 19 00:26 etc
drwxrwsr-x    7 root    staff        4096 abr 16 17:36 home
drwxr-xr-x    6 root    root          4096 feb  1 18:02 lib
drwxr-xr-x    2 root    root       16384 nov  7  2000 lost+found
dr-xr-xr-x    2 root    root          4096 nov 10  2000 mix
dr-xr-xr-x   67 root    root           0 may 19 02:25 proc
drwxr-xr-x   14 root    root          4096 feb 12 19:28 root
drwxr-xr-x    2 root    root          4096 ene 30 20:30 sbin
drwxrwxrwt    9 root    root          4096 may 22 10:19 tmp
drwxr-xr-x   15 root    root          4096 nov  8  2000 usr
drwxr-xr-x   16 root    root          4096 nov  9  2000 var
```

- La estructura de todos los Unix se *parece*
- La estructura de todas las distribuciones Linux se *parece mucho*



# Jerarquía clásica

La jerarquía actual puede resultar algo ilógica, pero hay motivos históricos

En los primeros Unix los discos eran más pequeños y más caros, en uno estaba lo *imprescindible* para que el sistema funcionase:

```
/
/etc
/lib
/tmp
/bin
/root
```

y en un segundo disco, se montaba /usr

/usr/spool

/usr/bin

/usr/include

/usr/tmp

/usr/adm

/usr/lib

# FHS Filesystem Hierarchy Standard

Estándar propuesto para todos los Linux y para los UNIX que quieran unirse. Año 1994. Versión actual: 3.0 (junio 2015)

Dos criterios

¿Un fichero puede almacenarse en una máquina y usarse en otra?

- Sí: Compartibles. (*shareable*)
- No: No compartibles. (*unshareable*)

¿Un fichero puede cambiar sin intervención del administrador?

- Sí: Dinámicos.
- No: Estáticos. Pueden almacenarse el modo sólo-lectura.  
Copias de seguridad menos frecuentes

- 1 Directorios de usuarios
- 2 Programas (incluyendo mandatos y librerías)
- 3 Configuración del sistema
- 4 El Hardware
- 5 Documentación
- 6 Ficheros Temporales
- 7 Otros directorios relacionados con el S.O.
- 8 Puntos de montaje

# Directorios de usuarios

- Directorio del administrador  
/root
- Usuarios locales  
/home/jperez  
o bien  
/home/profesores  
/home/alumnos

# Programas y mandatos

- Mandatos útiles para todos los usuarios

`/bin`

`/usr/bin`

- Mandatos útiles para el root

`/sbin`

`/usr/sbin`

(Todo lo que haya bajo `/usr` debería ser sólo lectura)

- Programas
  - Software no incluido en la distribución Linux  
/usr/local
  - Grandes aplicaciones en la distribución  
/opt

- Librerías estáticas y dinámicas
  - /lib
  - /usr/lib
  - /usr/local/lib
- Ficheros de cabecera (para compilar)
  - /usr/include
- Ficheros independientes de la arquitectura
  - /usr/share



# Configuración del sistema

## Directorio /etc

- Información sobre el sistema de ficheros (puntos de montaje, opciones)  
/etc/fstab
- cuentas de usuarios  
/etc/passwd
- Passwords de los usuarios  
/etc/shadow
- Scripts para arranque del sistema  
/etc/init.d
- ...

# El Hardware

## Los dispositivos del sistema /dev

/dev/hda	IDE primario master
/dev/hdb	IDE primario slave
/dev/hdc	IDE secundario master
/dev/hdd	IDE secundario slave
/dev/hda1	Primera partición primaria del hda
/dev/hda2	...
/dev/sda	Primer disco SCSI
/dev/sdb	Segundo disco SCSI
/dev/sda1	...
/dev/nvme0n1	Primer disco nvme
/dev/nvme0n2	Segundo disco nvme
/dev/nvme0n1p1	Primera partición del primer disco nvme

```
/dev/cdrom  
/dev/fd0      disquete  
/dev/audio   tarjeta sonido  
/dev/modem  
/dev/mouse  
/dev/input/mouse0  
/dev/ttyN    donde N es el nº de consola (no gráfica)  
/dev/pts/N   Consola (X Window)
```

El estándar no dice mucho sobre /dev, es bastante variable

- Archivos *virtuales* que representan las estructuras del Kernel en ejecución, dan información sobre la cpu...

<code>/proc/cpuinfo</code>	CPU
<code>/proc/pci</code>	Tarjetas PCI
<code>/proc/ioports</code>	Puertos I/O
<code>/proc/meminfo</code>	Información sobre la memoria
<code>/proc/NN</code>	Información sobre el proceso de pid NN

Los directorios `/proc` y `/sys` no se corresponden con discos físicos, sino que son un medio de enviar y recibir información directamente del *kernel*.

Cuando se lee o se escribe algún fichero del `/proc`, se está pidiendo o recibiendo información del kernel

# Documentación

- `/usr/share/doc`  
Documentación sobre el software del sistema
- `/usr/man`  
Ficheros del mandato *man*

# Ficheros Temporales

- Ficheros temporales  
(se borran cuando la máquina arranca)  
/tmp
- Fragmentos de ficheros recuperados  
/lost+found

- Ficheros que cambian con frecuencia, de aplicaciones

`/var`

`/var/log/syslog`

bitácora principal del sistema

`/var/log/messages`

otra bitácora con diversos mensajes

`/var/log/dmesg`

mensajes del sistema al arrancar

`/var/spool/lpd/lp`

spool de la impresora

`/var/tmp`

Ficheros temporales

`/var/mail`

Correo de los usuarios

- Ficheros del sistema que cambian con frecuencia  
`/run`

Esta es la principal novedad respecto a la versión anterior, ( FHS 2.3, año 2004). El directorio equivalente a este era `/var/run`. Resultaba problemático porque `/var/run` normalmente no estaba disponible durante el arranque (`/var` no es especialmente importante, podía estar en una partición distinta)



# Otros directorios relacionados con el S.O.

- `/boot`  
Todo lo requerido para el arranque, antes de que el sistema ejecute programas de usuario
- Código fuente
  - Código fuente del software de sistema  
`/usr/src`
  - Código fuente del kernel linux  
`/usr/src/linux`

# Puntos de Montaje

Unidades extraíbles: Disquetes, cdrom, *pendrives*

Solían colocarse en el raíz p.e. /cdrom. Pero esto llena el raíz de directorios

En FHS 2.3 (año 2004) aparece /media

/media/cdrom /media/cdrecorder /media/zip /media/floppy

- Si solo hay uno de un tipo:  
/media/cdrom
- Si hay más de uno del mismo tipo  
/media/cdrom0  
/media/cdrom1  
/media/cdrom -> /media/cdrom1

`/mnt`

Directorio vacío para que el administrador monte un sistema de ficheros que necesita temporalmente. Los programas no deberían usarlo

- `/mnt/cdrom`      ¡No es estándar!  
Es una costumbre reciente, va contra el estándar. Dentro de `/mnt` debe estar directamente el sistema de ficheros temporal, sin subdirectorios

# Montaje de sistemas de ficheros

- Normalmente, no todos los ficheros del árbol de directorios se encuentran en el mismo disco.
- *Punto de montaje*: directorio que pertenece a un disco (o *partición*) distinto, junto con todo su contenido (excluyendo otros puntos de montaje).
- Se pueden consultar los puntos de montaje junto con los discos o particiones que están *montadas* en ellos con las órdenes *mount* y *df*

# mount, df, lsblk

- **mount**: Muestra las particiones, puntos de montaje, tipo de partición y opciones de cada una de ellas:

```
/dev/hda2 on / type ext3 (rw,noatime)
proc on /proc type proc (rw)
sysfs on /sys type sysfs (rw)
devpts on /dev/pts type devpts (rw,gid=5,mode=620)
/dev/hda5 on /scratch type ext3 (ro,noatime)
tmpfs on /tmp type tmpfs (rw)
```

- **df**: Muestra cada una de las particiones *con ficheros reales* montadas en el sistema, el punto en el que está montada, su capacidad y su uso:

Filesystem	1K-blocks	Used	Available	Use%	Mounted on
/dev/hda2	28842780	6957692	20419960	26%	/
/dev/hda5	38448276	32838556	3656620	90%	/scratch
tmpfs	517960	1196	516764	1%	/tmp

- **lsblk -f**: Muestra cada disco (dispositivo de bloques)

## Para montar un sistema de ficheros

- Crear el directorio si no existe:  
`mkdir /var`
- Hacer visible el sistema de ficheros bajo ese directorio:  
`mount -t ext2 -o rw /dev/hda3 /var`  
(es más habitual indicar las opciones en `/etc/fstab`)
- Si queremos desmontar (o hacer invisible) un sistema de ficheros que esté montado en el directorio `/var`:  
`umount /var`

```

# <filesystem> <mount point> <type> <options> <dump><pass>
proc /proc proc defaults 0 0
/dev/hda2 / ext3 noatime 0 1
/dev/hda5 /scratch ext3 noatime,ro 0 1
/dev/hda6 none swap sw 0 0
tmpfs /tmp tmpfs defaults 0 0
/dev/sda1 /media/pendrive vfat defaults,user,noauto 0 0

```

- `mount -a` monta todo lo indicado en este fichero
- En el arranque se ejecuta `mount -a`
- `mount /media/pendrive`  
monta el pendrive con todas las opciones indicadas en `fstab`

<dump> ¿Incluir en las copias de seguridad hechas con *dump*?  
(Normalmente no)

<pass> Orden para el `fsck` del arranque (0: desactivado).

## <options>

- `rw`: Permisos de lectura y escritura.
- `ro`: Sólo lectura.
- `auto/noauto`: ¿Montar automáticamente con `mount -a`?
- `user/nouser`: ¿Los usuarios normales pueden montar y desmontar? (o hace falta ser `root`)
- `exec/noexec`: ¿Se pueden ejecutar binarios?
- `sync`: Al modificar un fichero, se escribe físicamente de inmediato
- `async`: Se usan buffers
- `defaults`: `rw, suid, dev, exec, auto, nouser, async`
- ...



# Tipos de sistemas de ficheros

## ● Tradicionales

- `msdos`: El usado por MS-DOS y Windows pre-95, sin permisos ni dueños, nombres de fichero de 8 caracteres con extensiones de 3 caracteres
- `vfat`: Usado a partir de Windows-95, compatible con MS-DOS pero con posibilidad de nombres de fichero largos
- `ntfs`: Desde Windows NT hasta Windows XP. Añade características de seguridad (permisos, dueños, etc). Los primeros *drivers* para Linux tenían limitaciones, en la actualidad se puede leer y escribir con normalidad
- `iso9660`: Sistema de fichero utilizado en los CDs de datos
- `minix`: usado por MINIX y por los primeros Linux
- `ext2`: Sistema de ficheros tradicional en Linux

- Con *journal*
  - ext3: Siguiete versión del ext2, idéntico pero con adición de *journal*. El más utilizado actualmente
  - reiserfs, jfs, xfs
    - Mejores prestaciones, pero incompatibles con ext2
- Con características especiales :  
romfs, cramfs, autofs, umsdos
- No asociados a dispositivo  
proc, sysfs, devfs, devpts, tmpfs, ramfs, usbfs

- Remotos:

- *nfs*: *Network File System*, desarrollado por SUN, el más usado entre los sistemas ficheros remotos en UNIX
- *smb/cifs*: Sistema de ficheros remotos usado por Microsoft
- *ncp*: *Netware Core Protocol*, protocolo sobre IPX para montar sistemas de ficheros de Novell Netware
- *sshfs*: *Secure SHell FileSystem*, protocolo basado en ssh

- Soporte de otras plataformas:

*hfs* (Apple Macintosh), *bfs* (*Boot File System*, SCO), *efs* (SGI, IRIX), *jffs* (*Journaling Flash File System*), *hpfs* (OS/2), *qnx4*, *sysv* (System V), *ufs* (SunOS, FreeBSD, NetBSD, OpenBSD)...

# Sistemas de Ficheros en Espacio de usuario

- Los sistemas de ficheros tradicionales están implementados en el núcleo. Añadir uno sistema de ficheros es complicado, y puede comprometer la integridad del sistema.
- Los sistemas de ficheros en espacio de usuario son aplicaciones *normales*
- Para Linux, FreeBSD, NetBSD, OpenSolaris y Mac OS X existe FUSE *Filesystem in Userspace*. Es un módulo del núcleo que actúa de puente entre el núcleo y el código del sistema de ficheros

## Ejemplos de sistemas de ficheros FUSE

- sshfs
- GmailFS. Almacena los datos sobre correos de gmail. No es fiable porque no está aprobado por google. (Tampoco prohibido, al menos explícitamente)
- Acceso a ficheros empaquetados (tgz, zip, etc)
- Almacenamiento en Bases de Datos
- Encriptación
- Hardware poco común
- Sistemas de versiones de ficheros (CVS, SVN...)
- Monitorización de sistemas de ficheros

Secure SHell FileSystem. Basado en FUSE. Sistema de ficheros de red

- Menos eficiente pero más seguro que NFS
- En el servidor basta disponer del demonio ssh convencional
- En el cliente basta instalar el paquete sshfs

Montar el *home* remoto:

```
sshfs -C usuario@maquina: /punto/de/montaje
```

Montar un directorio remoto

```
sshfs -C usuario@maquina:/un/directorio /punto/de/montaje
```

Desmontar:

```
fusermount -u /punto/de/montaje
```

- El sistema de arranque tradicional de Linux (System V) no es adecuado para las máquinas actuales
  - Son externos: aparecen y desaparecen
  - Están en red
  - Ahorran energía
  - ...
- *Upstart* es un sistema de arranque basado en eventos, desarrollado por Ubuntu, con el propósito de extenderlo a todos los Linux  
Aparece en Ubuntu 6.10 *edgy* (Octubre de 2006)
- Alternativas: *launchd* (macOS X), *initng*, SMF
- Está previsto que reemplace a *cron* y tal vez a *inetd*, manteniendo siempre la compatibilidad

En *upstart* se modifica la columna <filesystem> de /etc/fstab, incorporando un *Universally Unique Identifier*

```
# <file system> <mount point> <type> <options><dump><pass>
proc /proc proc defaults 0 0
UUID=e8a76033-f833-490d-8a55-ceca132c2ba7 / ext3 defaults,errors=remount-ro 0 1
UUID=e38c8abf-1af7-49be-bba5-bcf45dab8dc2 /home ext3 defaults 0 2
UUID=967cf88c-7b0b-42a9-bf93-deb7b710aad2 /media/sda6 ext3 defaults 0 2
UUID=f5c3bc51-7795-4bc9-b18e-4a16b7496e93 none swap sw 0 0
/dev/hda /media/cdrom0 udf,iso9660 user,noauto 0 0
```



# Codificación de caracteres

Correspondencia entre un carácter de lenguaje natural y un símbolo en otro sistema de representación. En informática, uno o más octetos

A veces se llama *code pages* (IBM, Microsoft)

- EBCDIC: Extended Binary Coded Decimal Interchange Code. IBM, año 1963. 8 bits. Se usa en algunos equipos IBM. Diferentes versiones incompatibles entre sí
- ASCII: American Standard Code for Information Interchange. ANSI, American National Standards Institute, año 1963). 7 bits. Solo inglés

# ASCII extendido

8 bits. Cada conjunto de idiomas necesita su propia variante.

Compatible con ASCII

- Code Pages 437. Inglés. Primeros IBM PC, MS-DOS  
Code Pages 850. Europa occidental. Primeros IBM PC, MS-DOS
- ISO-8859 (Organización Internacional para la Estandarización), año 1992. Habitual en linux hasta mediados de los años *cerenta*  
ISO-8859-1, informalmente conocido como Latin-1  
ISO-8859-2 europa central, ISO-8859-5 cirílico , ISO-8859-6 árabe, ...  
ISO-8859-15 o Latin-9. Año 1998. Muy parecido a Latin-1, incluye el símbolo del euro
- windows-1252. Parecido a ISO-8859-1. Se confunden con frecuencia. Se empleaba en los primeros Windows

Estándar industrial. *Unicode Consortium*, año 1991. Compatible con ISO 10646.

Asocia un número a cada carácter empleado por algún lenguaje escrito del mundo. Más de 100.000 caracteres

Se puede codificar de diferentes maneras

- UTF-8 es la forma en Unix de codificar unicode. Compatible con ASCII. Cada carácter ocupa entre 1 y 4 octetos
- UTF-16. Cada carácter ocupa entre 2 y 4 octetos. Nativo en Windows desde Windows 2000, aunque se seguía usando windows-1252.
- Punycode. RFC 3492. Empleado en la Internacionalización de Nombres de Dominio en Aplicaciones (IDNA). Años 2003-2005. Permite nombres de dominio en unicode.  
españa.es -> xn--espa-rta.es  
ortuño.es -> xn--ortuo-rta.es
- UCS-2, UCS-4, SCSU, ...

# recode

Orden que convierte ficheros entre diferentes codificaciones

- `recode utf-8`  
Lee *stdin*, convierte desde utf-8 hasta las locales actuales y escribe en *stdout*
- `recode latin-1..utf-8`  
Lee *stdin*, convierte desde latin-1 hasta utf-8 y escribe en *stdout*
- `recode utf-8..windows-1252 fichero`  
Modifica el fichero, convirtiendo desde utf-8 hasta windows-1252