Introducción

- El almacenamiento secundario es necesario para:
  - Almacenar gran cantidad de datos
  - Almacenar datos persistentes (válidos entre sesiones/reinicios)
  - Compartir datos (si la protección de la memoria no lo permite)

- Los dispositivos de almacenamiento secundario pueden ser muy distintos (discos rígidos, memorias flash, DVD, magnetic tape)

  ⇒

  el sistema operativo debe proporcionar una ***interfaz sencilla y uniforme*** para acceder a dichos dispositivos

Instituto
Politécnico
Superior

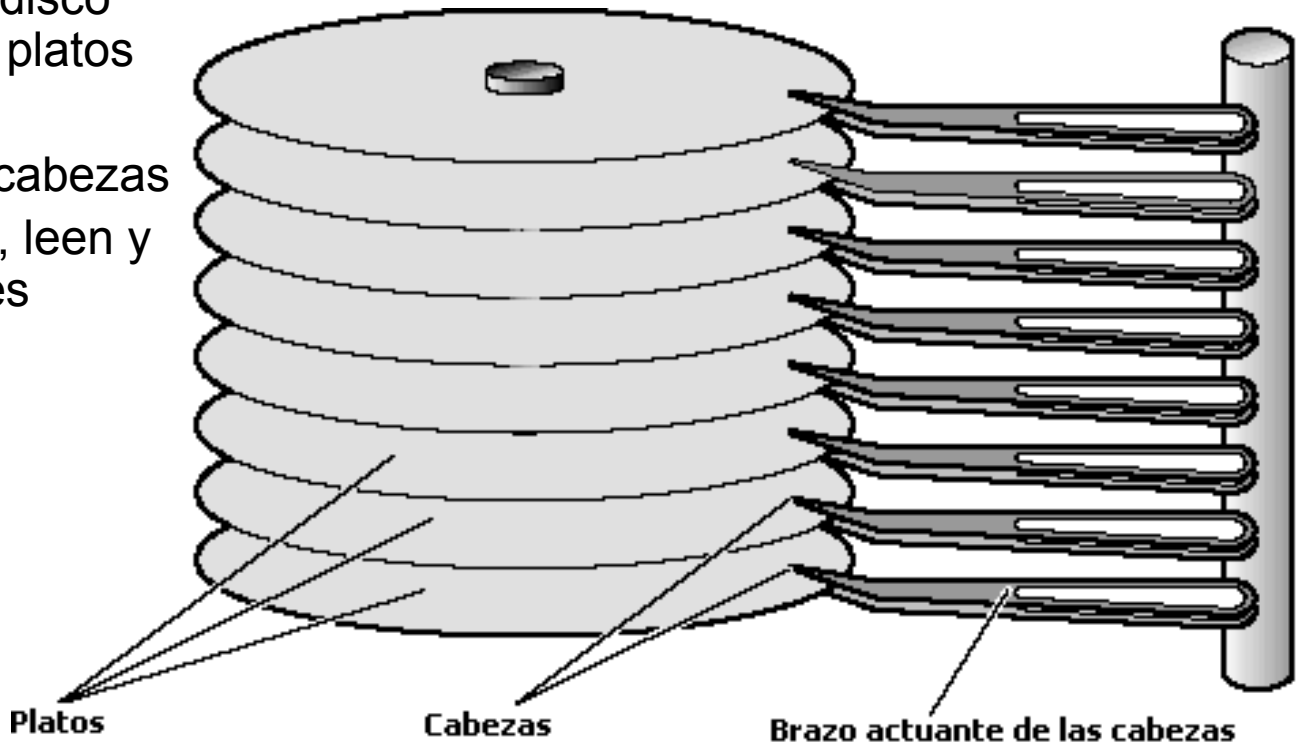
Estructuras de un disco rígido

• De los dispositivos de almacenamiento secundario los discos rígidos, actualmente, son los de uso más habitual

• Un disco rígido posee componentes físicos y lógicos

Componentes Físicos

- El área de almacenamiento del disco está compuesta por uno o más platos
- Los platos giran
- El brazo actuante desplaza las cabezas
- Las cabezas lectoras/escritoras, leen y escriben datos en las superficies magnéticas de ambos lados de los platos



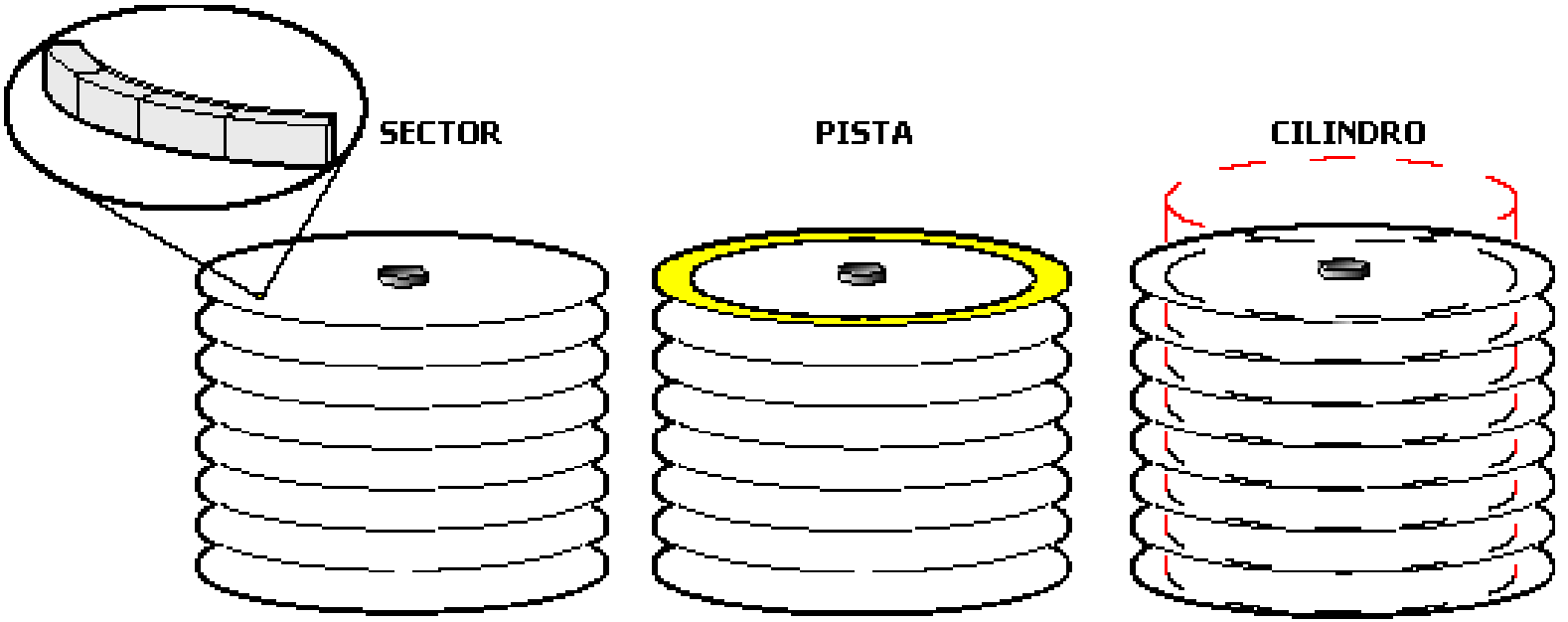
Estructuras de un disco rígido

Componentes lógicos

Sector o bloque de disco: la unidad más pequeña direccionable en un plato. Un sector puede contener 512 bytes de datos.

Pista (*Track*): Serie de sectores posicionados de manera circular. El número de sectores que contiene varía de acuerdo al radio de la misma

Cilindro: Pila o conjunto de pistas ubicadas a igual distancia del centro del plato



Introducción

• Un **sistema de archivos** (**file system** o **filesystem**) es parte integral de los sistemas operativos modernos y que permite almacenar y organizar archivos y los datos que contienen a fin de facilitar la búsqueda y acceso a los mismos
Es una base de datos de propósito especial para el almacenamiento, organización, manipulación y recuperación de datos.

- *Según el usuario*: un file system es un conjunto estructurado de información que comprende archivos y directorios
- *Según el sistema operativo*: está conformado por estructuras de control y bloques de datos que permiten el almacenamiento y organización de los datos

• `man -s5 filesystems`

Archivos

- Es la unidad de almacenamiento
- Los archivos se identifican mediante un nombre

- Existen diversos tipos de archivos:
  - Regulares (ASCII o binarios)
  - Directorios
  - Archivos especiales de caracteres
  - Archivos especiales de bloques

- Pueden ser accedidos de manera secuencial o aleatoria

- También son llamados ficheros

Archivos: atributos

Protección	Quién debe tener acceso y de qué forma
Contraseña	Contraseña necesaria para tener acceso al fichero
Creador	Identificador de la persona que creó el fichero
Propietario	Propietario actual
Bandera «sólo lectura»	0 Lectura/escritura, 1 para lectura exclusivamente
Bandera de ocultación	0 normal, 1 para no exhibirse en listas
Bandera de sistema	0 fichero normal, 1 fichero del sistema
Bandera de biblioteca	0 ya se ha respaldado, 1 necesita respaldo
Bandera ASCII/binario	0 fichero en ASCII, 1 fichero en binario
Bandera de acceso aleatorio	0 sólo acceso secuencial, 1 acceso aleatorio
Bandera temporal	0 normal, 1 eliminar al terminar el proceso
Bandera de cerradura	0 no bloqueado, $\neq 0$ bloqueado
Longitud de registro	Número de bytes en un registro
Posición de la clave	Ajuste de la clave dentro de cada registro
Longitud de la clave	Número de bytes en el campo clave
Tiempo de creación	Fecha y hora de creación del fichero
Tiempo del último acceso	Fecha y hora del último acceso al fichero
Tiempo de la última modificación	Fecha y hora de la última modificación del fichero
Tamaño actual	Número de bytes en el fichero
Tamaño máximo	Tamaño máximo al que puede crecer el fichero

Archivos: Operaciones

- Create: crea un archivo vacío
- Delete: elimina un archivo
- Open: abre un archivo para operar con él
- Close: cierra un archivo abierto
- Read/Write: lee/escribe datos de/en un archivo
- Append: escribe datos al final del archivo
- Seek: especifica el punto de lectura/escritura de datos en un archivo de acceso aleatorio
- Get/Set attributes: obtiene/establece los atributos asociados a un archivo
- Rename: cambia el nombre de un archivo en un directorio
- Truncate: elimina el contenido de un archivo a partir de una posición dada

Instituto Politécnico Superior

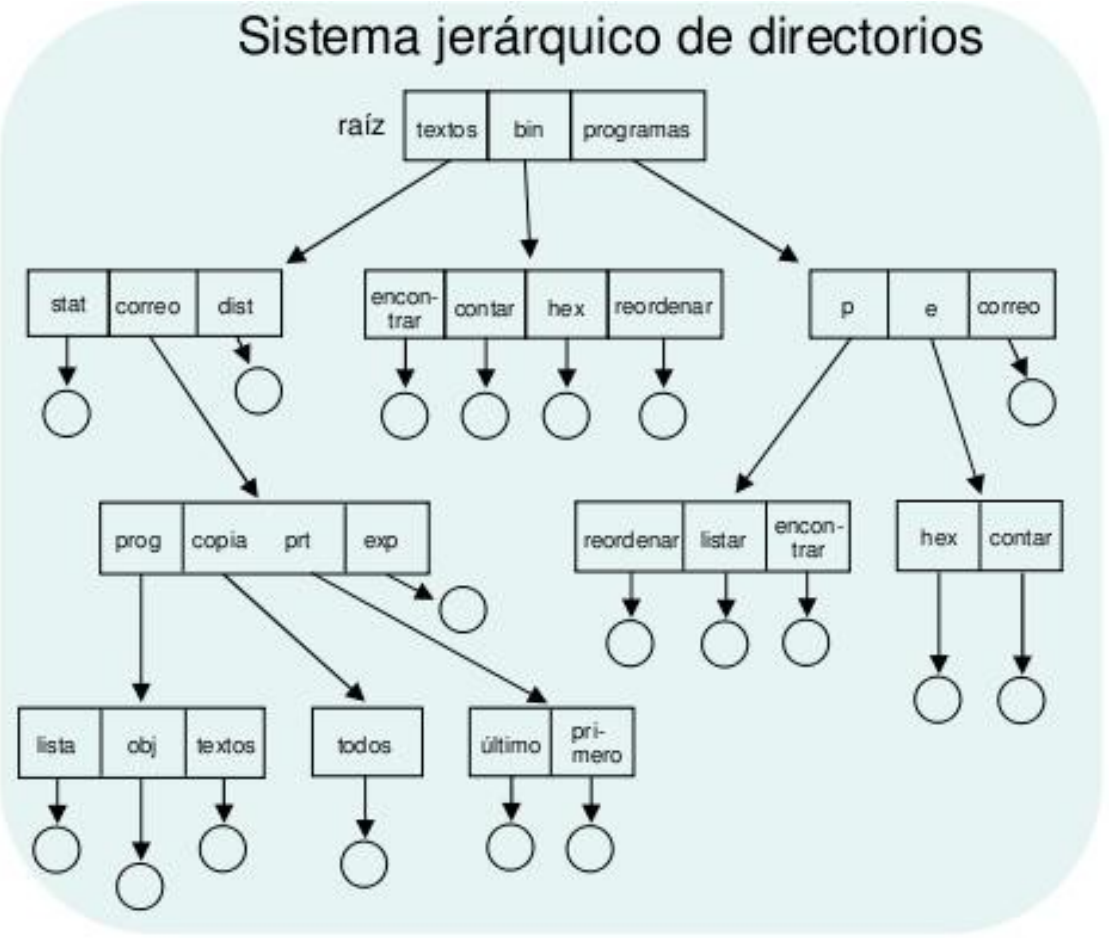
Directorios

- Son un tipo particular de archivo, asocian el nombre de los archivos con información sobre los mismos

- Generalmente se organizan jerárquicamente, dado que los directorios pueden contener información sobre otros directorios

- "Dentro" de los directorios pueden haber archivos y directorios

- Esta estructura tiene como objeto organizar datos relacionados por temas en diferentes directorios o subdirectorios.
  En los sistemas Linux/Unix se mantiene cierta estandarización



Jerarquía de Directorios

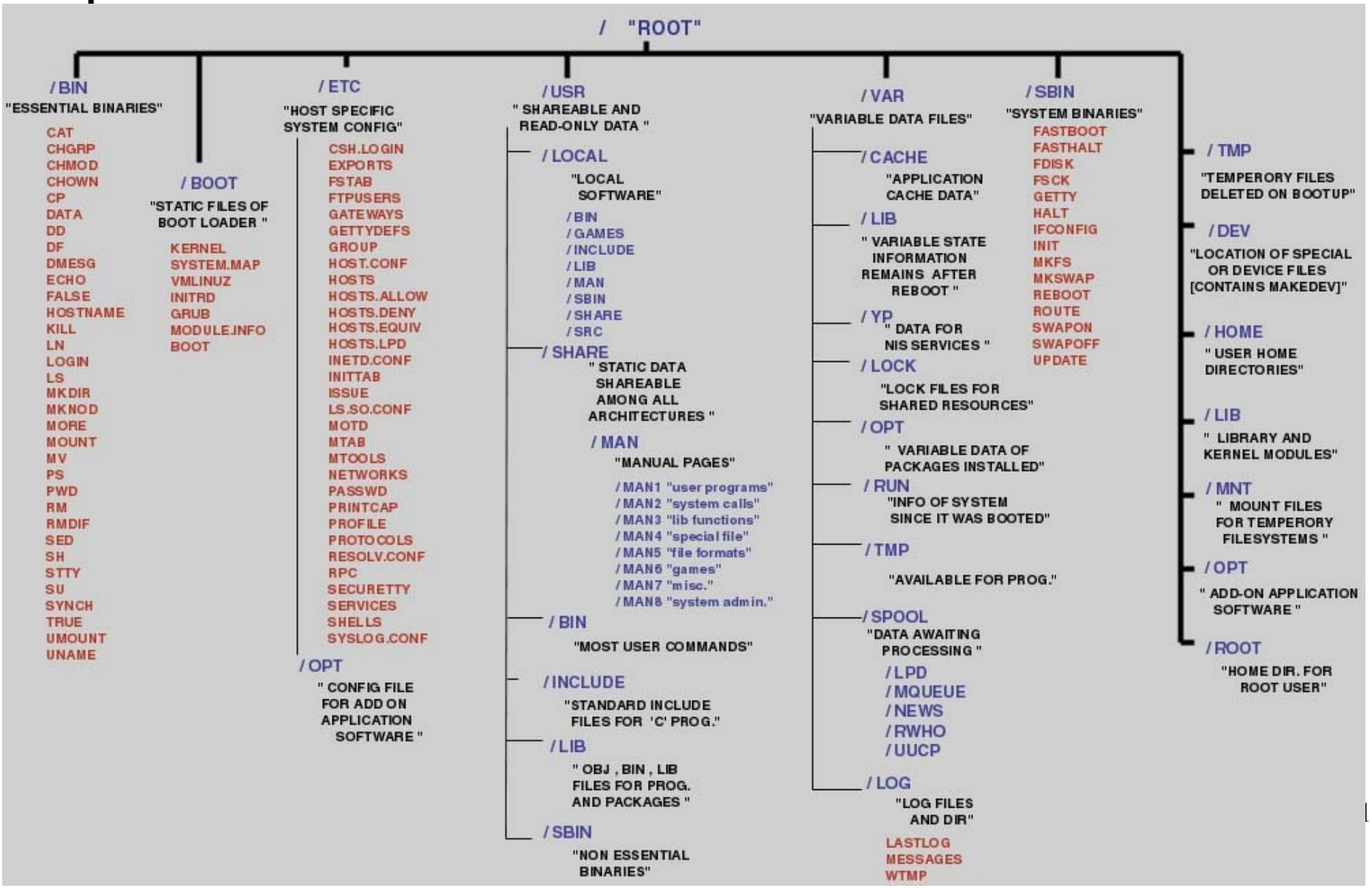
• En las estructuras jerárquicas de directorios, existen dos maneras de referir a un archivo

– Ruta absoluta / Path absoluto:
  • Especifica el camino desde el directorio raiz hasta el archivo
  • El primer carácter de la ruta es el separador («/» en Unix, «\» en Windows).
    Ejemplo: /usr/bin/mozilla

– Ruta relativa / Path relativo:
  • Asociada al concepto de directorio actual o de trabajo (comando pwd)
  • No empieza por el carácter separador y dependen del directorio actual.
    Ejemplo: bin/mozilla, si el directorio actual es /usr

• Existen dos directorios especiales:
  • Directorio “.”: directorio actual
  • Directorio “..”: directorio padre

Jerarquía de Directorios de Linux



Directorios: Operaciones

- Create: crea un directorio vacío
- Delete: borra un directorio vacío
- Opendir: abre un directorio para operar con él
- Closedir: cierra un directorio abierto
- Readdir: lee una entrada del directorio
- Rename: cambia de nombre a un directorio
- Link: crea un enlace físico para un fichero existente
- Unlink: elimina un enlace físico (o el fichero asociado si se elimina el último enlace)

Montaje y desmontaje de file systems

- Punto de montaje: directorio de la jerarquía de directorios que oficia como raíz de un file system

- Montaje: incorporar a la jerarquía de directorios file systems individuales con el fin de accederlos
- Desmontaje: desligar un file system de su punto de montaje

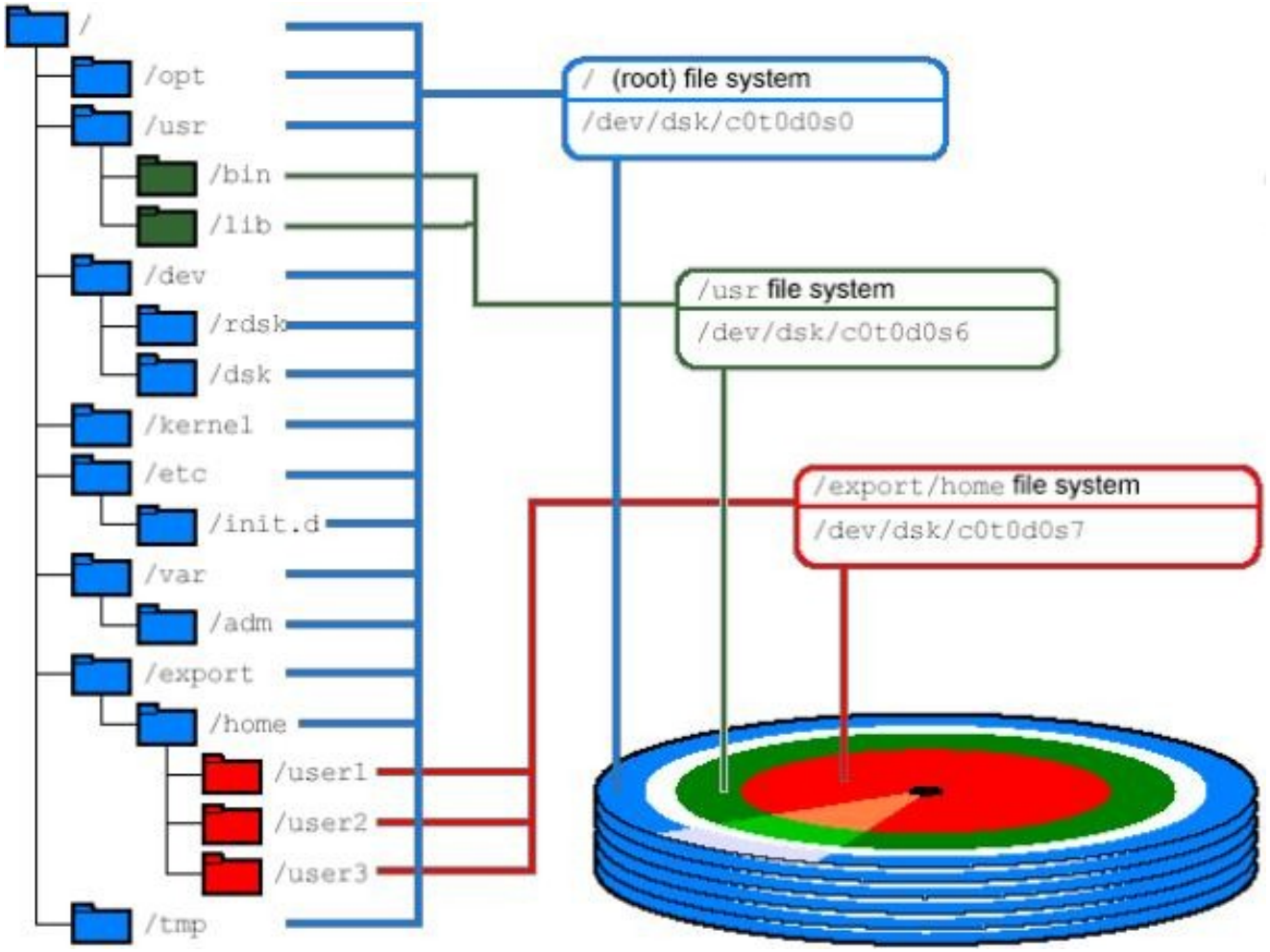
- Los file systems no contienen su propio punto de montaje

Instituto Politécnico Superior

Montaje y desmontaje de file systems

- En el gráfico se observa una jerarquía de directorios, donde diferentes file systems creados en particiones dedicadas de un disco rígido son montados en directorios (puntos de montaje)

- Los file sistems pueden montarse en boot time, desmontarse y volverse a montar en otra ubicación y establecerse diferentes opciones de montaje persiguiendo cuestiones de seguridad, performance u otro requerimiento



Montaje y desmontaje de file systems

- En Linux/Unix se emplean los comandos **mount** y **umount** para montar y desmontar file systems.

- Su uso típico es

```
# mount -t type device directorio
# umount directorio
```

- Para montar un file system debe generalmente especificarse su tipo (ejemplos: adfs, cifs, ext, ext2, ext3, hfs, msdos, nfs, reiserfs, vfat, etc.) a fin de que el sistema operativo pueda acceder a los datos según las especificaciones de cada uno de los file systems que soporte.

- También durante el montaje de un file system pueden precisarse opciones de montaje como ser: acceso sólo lectura, actualizar o no información de i-nodo al acceder al archivo, permitir o no archivos ejecutables, establecer escrituras sincrónicas, etc.

- http://en.wikipedia.org/wiki/Mount_(Unix)

I-nodo o nodo índice

- Los sistemas operativos Unix/Linux emplean una estructura de datos denominada i-nodo. Es un registro donde se almacena la mayor parte de la información de un archivo específico

- Están identificados por un número entero positivo único

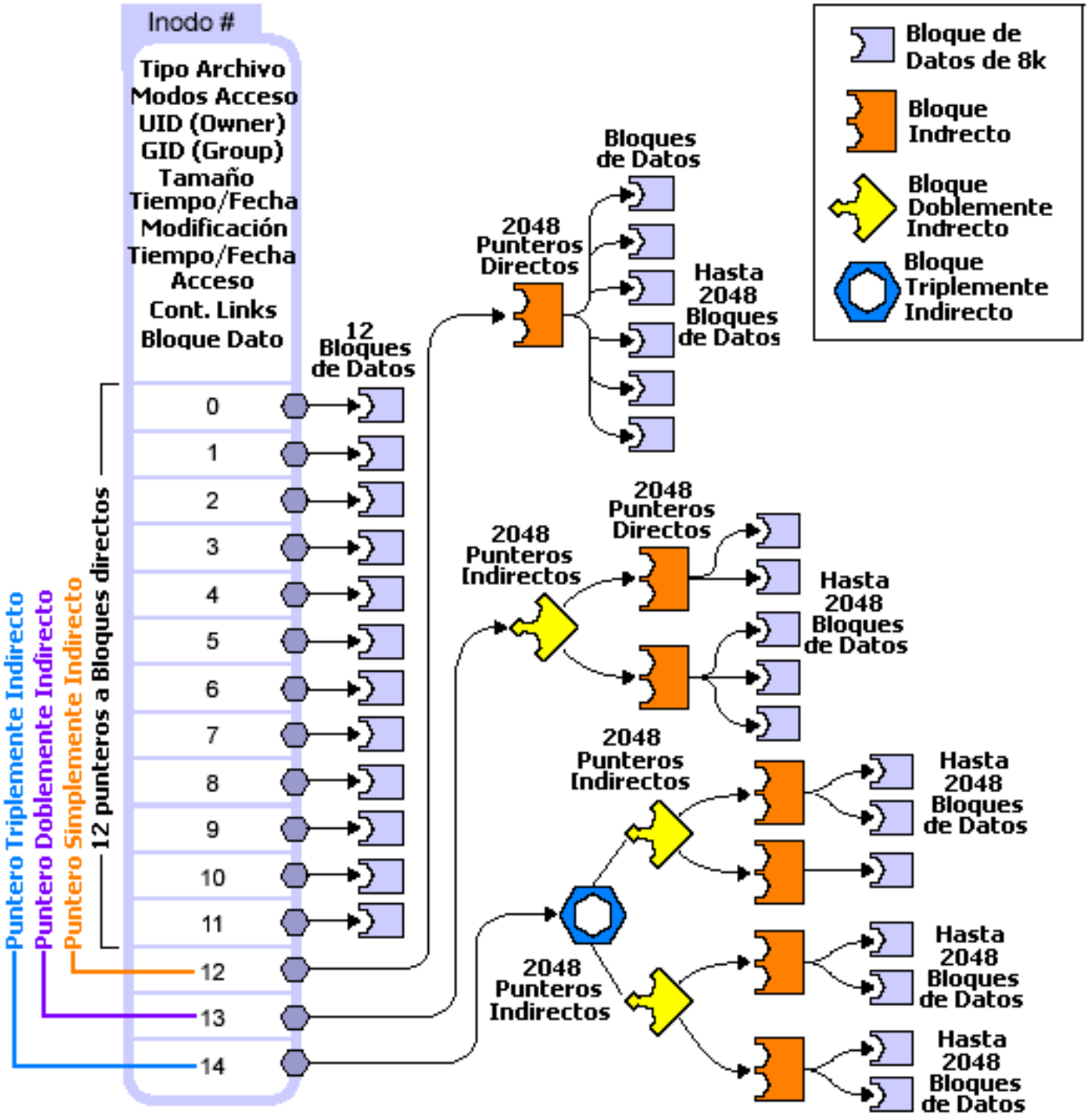
- Cada file system posee su propia lista de i-nodos

- En los sistemas UNIX tienen un tamaño de 64 bytes

- En el disco luego de los i-nodos están los bloques de datos. Cada uno de estos bloques referencia 8KB (fragmentos de 1KB)

- Más información: http://en.wikipedia.org/wiki/Inode

Instituto Politécnico Superior



I-nodo o nodo índice

- Punteros Directos
  - Dentro de cada i-nodo hay doce (12) punteros directos
  - Contienen las direcciones de los primeros 12 bloques del archivo
  - Cada uno referencia un bloque de 8Kb (en total a 96 Kb)

- Punteros Indirectos (existen tres tipos de punteros indirectos)
  - Punteros Simplemente Indirectos
    - Referencian a bloques de file system que contienen punteros a bloques de datos
    - Estos bloques contienen 2048 direcciones adicionales de bloques de datos de 8k
    - Referencian 16 MBytes de datos adicionales
  - Punteros Doblemente Indirectos
    - Referencian a bloques de file system que contienen punteros simplemente indirectos
    - Cada puntero indirecto referencia a bloques de file system que contienen punteros a bloques de datos
    - Referencian 32 Gytes de datos adicionales
  - Punteros Triplemente Indirectos
    - Pueden referenciar 64 Tbytes de datos adicionales

Enlaces (Links)

• Es deseable compartir información mediante archivos pero evitando que existan múltiples copias de los mismos

• En Linux y Unix se emplean dos tipos de enlaces

- **Enlaces Físicos / Hard links**
  - Necesitan estructuras tipo nodo-i
  - Dos entradas de directorio apuntan a un mismo nodo-i, por tanto como cada file system posee una lista de i-nodos propia, los hard links pueden establecerse dentro de un único file system
  - Se tiene un contador de enlaces en el nodo-i, así un archivo es eliminado cuando la última relación *nombre_archivo*↔*nodo-i* es eliminada, es decir cuando se elimina el último hard link

- **Enlaces Simbólicos / Symbolic Links**
  - Es una forma de obtener sinónimos de archivos
  - Un soft link es un archivo independiente al cual apunta y su contenido es una cadena de caracteres con la ruta al archivo apuntado
  - Al ser un archivo distinto tiene su propio nodo-i por lo tanto el eliminar un soft link no tiene efecto sobre el archivo apuntado
  - Un soft link puede apuntar a un archivo existente o no

Enlaces (Links)

```
$ touch archivo
$ ln archivo hard_lnik                  // Creación de hard link

$ ln -s archivo symbolic_link           // Creación de soft link
$ ls -l
-rw-r--r-- 2 diego diego  0 2009-11-29 21:38 archivo
-rw-r--r-- 2 diego diego  0 2009-11-29 21:38 hard_lnik
lrwxrwxrwx 1 diego diego  7 2009-11-29 21:39 symbolic_link -> archivo
```

"l" indica soft link

Hay **2** hard links. Dos nombres de archivo relacionados con un mismo i-nodo

Cantidad de caracteres de "archivo"

La salida muestra el archivo que apunta el soft link

```
$ ls -i archivo
3074025 archivo
$ ls -i hard_lnik
3074025 hard_lnik
$ ls -i symbolic_link
3074402 symbolic_link
```

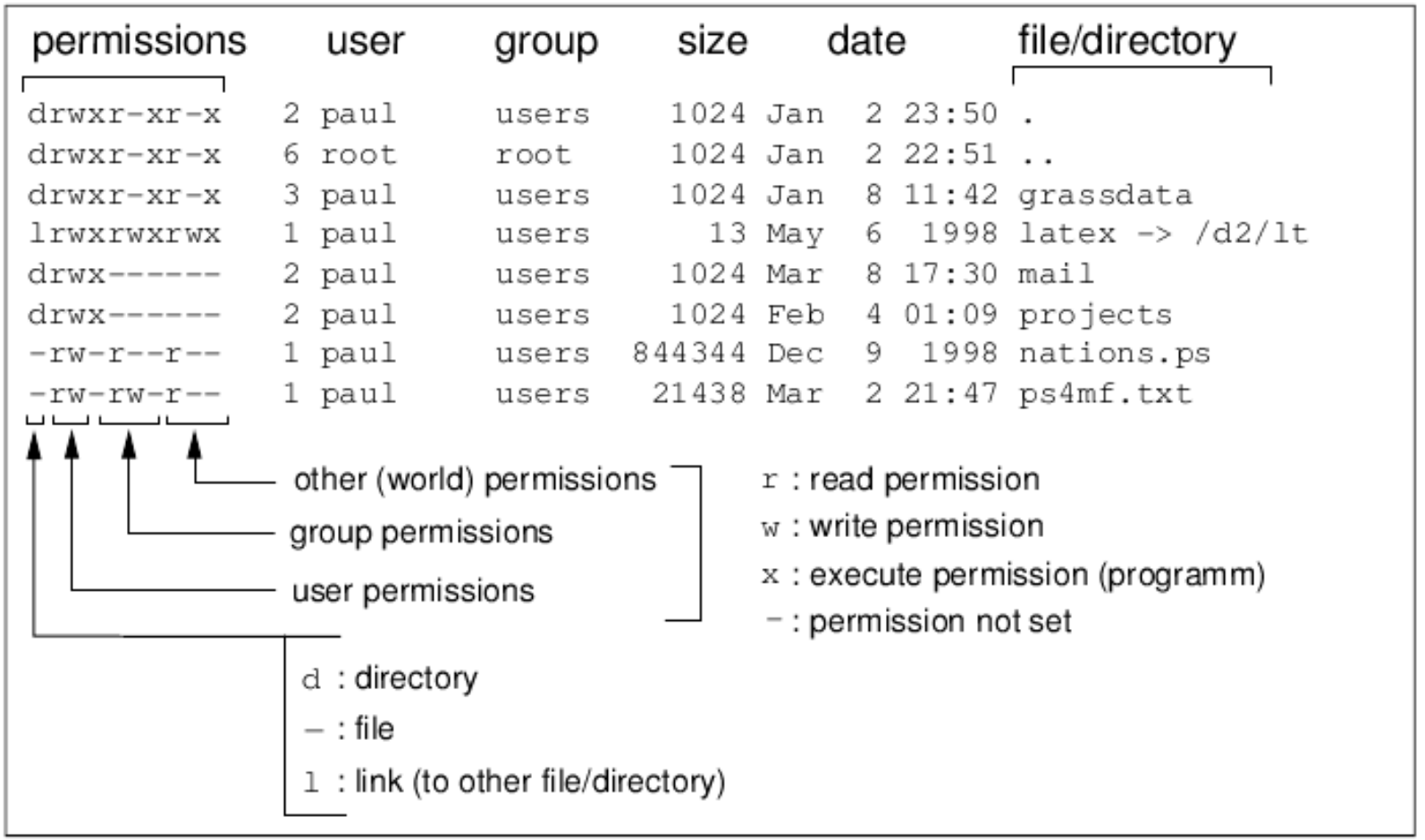
Observar que el hard link posee el mismo i-nodo mientras que el soft link no

Tipos de File Systems

- Existen diferentes tipos de file systems, pueden organizarse según sus funcionalidades y uso, aunque existe un conjunto de características siempre presentes y cada file system responde a éstas de diferentes manera
  - Consistencia e integridad
  - Performance
  - Seguridad
  - Robustez
  - Escalabilidad
  - Flexibilidad
  - Amigabilidad en administración

- Puede encontrarse una tabla comparativa de diferentes file systems en el siguiente enlace http://en.wikipedia.org/wiki/Comparison_of_file_systems

Seguridad: Permisos UNIX estándar



http://en.wikipedia.org/wiki/Unix_permissions#Notation_of_traditional_Unix_permissions

Seguridad: Permisos UNIX estándar

- En Unix la seguridad de los diferentes objetos del sistema viene determinada por usuario y grupo

- A estas entidades el sistemas las representa a través de dos números, el uid (user id) para el usuario, y el gid (group id) para el grupo.

- Cada usuario pertenece a un grupo primario y puede pertenecer a varios grupos secundarios.

- Esta información está almacenada en los archivos **`/etc/passwd`** y **`/etc/group`**. En los Unix más modernos, no se accede a ellos directamente, sino que existe todo una serie de módulos - conocidos como PAM - que permiten obtener la información necesaria de otras fuentes como pueden ser NIS, LDAP u otro sistema de nombres

Seguridad: Permisos UNIX estándar

- Los permisos que pueden existir sobre los objetos de un file system Unix/Linux son
  - lectura (r, **r**ead). Permiso de lectura
  - escritura (w, **w**rite). Permiso de escritura
  - ejecución (x,e**x**ecute/search). Permiso de ejecución. Cuando se aplica a directorios, sirve para que pueda buscarse un archivo dentro de los mismos
  - setuid (setuid). Un archivo con este permiso, cuando lo ejecuta cualquier usuario, lo hace con los permisos del propietario del archivo
  - setuid (setgid). Igual que setuid, pero para los grupos
  - sticky bit (t). Aplicado a un directorio, hace que los archivos que existen dentro del mismo, puedan borrarlos sólo el root o el propietario - frente al caso normal, donde cualquiera con permisos de escritura pueda hacerlo (ejemplo directorio /tmp)

- Cuando un usuario (user) que pertenece a los grupos (group) intenta acceder a un objeto dentro del file system, cuyo propietario es fuser y el grupo es fgroup
  1. usuario. Si user es igual que fuser se comprueba los permisos asociados al propietario.
  2. grupo. Si user pertenece al grupo fgroup. En este caso se comprueban los permisos asociados al grupo.
  3. otros. Si user no cumple con las condiciones anteriores se consideran los permisos asociados a otros.

Seguridad: Listas de Control de Acceso - Access Control Lists (ACLs)

- Un ACL es una lista de permisos referida a un objeto que especifica que usuario o proceso puede acceder a un objeto y que operaciones puede realizar sobre el mismo.

- En la ACL de un file system, la lista es una estructura de datos (usualmente una tabla) con entradas que especifican los derechos que un usuario, grupo o proceso tiene sobre objetos específicos, tales como programa o archivos.
  Estas entradas usualmente se denominan ACEs (Access Control Entries).
  Cada objeto accesible contiene un identificador a su ACL, que especifica quien puede, por ejemplo, leer, escribir o ejecutar un objeto.
  En algunas implementaciones, una ACE puede controlar si los sujetos de la ACL de un objeto particular pueden alterar la misma.

Seguridad: Listas de Control de Acceso - Access Control Lists (ACLs)

• Ejemplo

```
$ ls -l archivo
-rw-r--r-- 1 usuario1 grupoA 10 2010-02-28 20:22 archivo
$ getfacl archivo
# file: archivo
# owner: usuario1
# group: grupoA
user::rw-
group::r--
other::r--
```

(Supongamos que **usuario2** pertenece al grupo **grupoB**)

```
$ setfacl -m mask::rwx u:usuario2:rx archivo
# file: archivo
# owner: usuario1
# group: grupoA
user::rw-
user:usuario2:rw-
group::r--
mask::rwx
other::r--
```

Integridad: Journaling

- Un file system que soporta *journaling* es un file system preparado para afrontar fallas, la integridad de los datos es asegurada gracias a que las actualizaciones de directorios y demás estructuras son escritas constantemente a un log serial (journal) en disco antes de realizar actualizaciones en el file system.

- Con journaling se escribe *metadata* en el journal el cual es volcado a disco antes de que retorne cada comando. En caso de una falla del sistema, un conjunto de actualizaciones serán realizadas por completo al file system (escrito a disco y donde corresponda),
  Otras actualizaciones si no se efectúan por completo son etiquetadas como “not yet fully commited” y el sistema leerá el journal a fin de realizar un *roll up* para alcanzar el punto de consistencia de datos más reciente..

- File Systems con *journaling*
  ext3, ext4, reiserfs, reiser4 de Linux, UFS de Solaris, JFS de AIX, NTFS de Windows NT

- http://en.wikipedia.org/wiki/Journaling_file_system

Flexibilidad: Logical Volume Manager (LVM)

- Si bien no todo file system además posee características de LVM vale la pena mencionar esta funcionalidad que la mayoría de los sistemas operativos ofrecen

- Un LVM es una administrador de volúmenes lógicos.
  Un *volúmen físico (PV)* puede ser una partición, un disco o una unidad lógica de un dispositivo de almacenamiento externo.
  Un *volúmen lógico (LV)* es el equivalente a una partición en un sistema tradicional. El LV es visible como un dispositivo estándar de bloques, por lo que puede contener un file system

- LVM brinda un nivel superior de flexibilidad respecto a la alocación de almacenamiento para las aplicaciones y usuarios

- Muchos LVM permiten definir esquemas RAID obteniendo niveles de redundancia para los datos

- http://en.wikipedia.org/wiki/Logical_volume_manager

LECTURA DE LIBROS

Tanenbaum – Sistemas Operativos Modernos (págs. 165 a 229)

Silberschatz & Galvin – Sistemas Operativos 5ta Ed (págs. 337-393)

Stallings – Sistemas Operativos 4ta Ed (págs. 513 a 549)