SISTEMAS DE ARCHIVOS

INTRODUCCION

Los sistemas de archivos (file system en inglés), estructuran la información guardada en una unidad de almacenamiento (normalmente un disco duro de una computadora), que luego será representada ya sea textual o gráficamente utilizando un gestor de archivos. La mayoría de los sistemas operativos poseen su propio sistema de archivos.
Estos sistemas de archivos tienen un nombre y una norma que los define. Por ejemplo tenemos los siguientes sistemas de archivos:

FAT 16
FAT 32
NTFS
EXT2
EXT3
VFAT
…..
……
Cada sistema de archivo tiene su propia forma de manejo de tal manera que los sistemas operativos deberán conocer dichas normas para poder trabajar con ellos. Cada sistema operativo normalmente tiene módulos o programas que son usados para interpretar el contenido de un sistema de archivo almacenado en algún dispositivo físico.
Lo habitual es utilizar dispositivos de almacenamiento de datos que permiten el acceso a estos, como una cadena de bloques de un mismo tamaño, a veces llamados sectores,

usualmente de 512 bytes de longitud. Sin embargo el sistema operativo solo maneja discos lógicos (también llamados ―file systems). Se deja para los programas conocidos como ―drivers el manejo de las características de bajo nivel como los son los cilindros, pistas y sectores. También las rutinas del BIOS pueden reconocer las características de bajo nivel de los dispositivos de almacenamiento.
Los sistemas de archivos tradicionales proveen métodos para crear, mover, renombrar y eliminar tanto archivos como directorios.
Todas las aplicaciones necesitan almacenar y recuperar información. Mientras un proceso está ejecutándose puede almacenar cierta cantidad de información dentro de su propio espacio de direcciones. Sin embargo, esa capacidad de almacenamiento está limitada por el tamaño del espacio de direcciones virtual. Para algunas aplicaciones ese tamaño es adecuado, pero para otras, tales como la reserva de billetes de avión, la banca o el registro de las operaciones realizadas por una empresa, resulta demasiado pequeño.
Un segundo problema con el que nos encontramos al guardar la información dentro del espacio de direccionamiento de un proceso es que cuando el proceso termina, la información se pierde. Para muchas aplicaciones (por ejemplo para las bases de datos) la información debe ser retenida durante semanas, meses o incluso para siempre. Es inaceptable permitir que la información se desvanezca cuando termina el proceso que la utiliza. Además, tampoco debe perderse aunque el proceso se destruya repentinamente debido a una caída del sistema.
Un tercer problema es que frecuentemente es necesario que múltiples procesos accedan a (partes de) la información al mismo tiempo. Si disponemos de una guía telefónica almacenada dentro del espacio de direccionamiento de un único proceso, sólo ese proceso va a poder acceder a ella. La manera de resolver este problema es hacer que la información sea ella misma independiente de cualquier proceso.
Entonces tenemos ya tres requerimientos esenciales para el almacenamiento a largo plazo de la información:

- Debe poder almacenarse una cantidad de información muy grande.
- La información debe permanecer tras la terminación del proceso que la usa.
- Debe permitir que múltiples procesos puedan acceder a la información concurrentemente

La solución usual a todos estos problemas es almacenar la información sobre discos y otros medios externos en unidades denominadas archivos. Los procesos pueden entonces leerlos y crear nuevos archivos si es necesario. La información almacenada en los archivos, debe ser persistente, no debe verse afectada por la creación y terminación de los procesos. Un archivo sólo puede desaparecer cuando su propietario lo borre de forma explícita.
Los archivos están gestionados por el sistema operativo. Pero la forma de cómo están estructurados, cómo se nombran, se acceden, se utilizan, se protegen, se transfieren e implementan, son temas principales en el diseño de los sistemas de archivos. Globalmente, a esa parte del sistema operativo que trata los archivos y directorios se la conoce **como administración del sistema de archivos** y es el tema de este capítulo.
Desde el punto de vista de los usuarios, el aspecto más importante de un sistema de archivos es su apariencia, es decir, qué constituye un archivo, como se nombran y se protegen los archivos, qué operaciones se permiten, etc. Los detalles, de si para seguir la pista de la memoria libre se utilizan listas enlazadas o mapas de bits, o el detalle de cuántos sectores hay en un bloque lógico, son cuestiones de menos interés, aunque son de gran importancia para los diseñadores del sistema de archivos. Por esa razón, hemos

estructurado el capítulo en varias secciones. Las primeras secciones tienen que ver con la interfaz del usuario con los archivos y con los directorios, respectivamente. A continuación se discutirá en detalle la forma en la cual se implementa el sistema de archivos.
Finalmente, daremos algunos ejemplos de sistemas de archivos reales.

Al finalizar este capitulo, usted estará en condiciones de alcanzar los siguientes objetivos:

- Identificar las distintas funciones del sistema operativo, referente a la administración dearchivos.
- Diferenciar los distintos métodos de administración de archivos por parte del sistema operativo UNIX, Linux y Windows
- Reconocer las características de implementación de los distintos sistemas de archivos.
- Identificar las distintas estructuras, que el sistema operativo construye, para la administración de archivos.
- Reconocer las limitaciones del almacenamiento de los distintos sistemas de archivos.
- Reconocer las posibilidades de operación con archivos y directorios.

En esta unidad veremos los distintos tipos de sistemas de archivos. Esto nos servirá como ejemplo, para comprender como los sistemas operativos modernos, deben solucionar el problema de gestión de archivos y directorios.

Finalmente, conoceremos las distintas estructuras que utiliza el sistema operativo, en particular veremos el caso práctico de UNIX, Linux y Windows para concretar eficientemente la administración de archivos veremos el caso práctico de UNIX, Linux y Windows para concretar eficientemente la administración de archivos

Tipos de sistemas de archivos

Con el tiempo se le ha dotado de distintos sistemas de archivos, de forma que sus usuarios pueden elegir entre una amplia variedad de opciones y tener acceso (en ocasiones restringido) a una enorme cantidad de formatos. El sistema es muy flexible, permitiendo incluso montar un sistema de archivos de raíz única, que comprenda unidades físicas con sistemas distintos entre sí (unidades con formatos distintos). Entre los sistemas de archivo que el s.o. puede utilizar o "entender" se encuentran: fat16, fat32, ntfs, bfs; minix; ext; ext2; ext3; xia; xfs; msdos; umsdos; vfat; jfs; reiserfs; nfs; iso9660; hpfs; sysv; smb, ncpfs, etc. A continuación se comentan brevemente las características de los más usados: *(Antes de continuar con la lectura le sugiero que lea el apunte Sistemas_de_archivos_FAT y consultar con el docente el tema de unidades y otros que Ud tenga dudas)*.

• **fat16**: En 1987 apareció lo que hoy se conoce como el formato FAT16. El tamaño de la está limitado por la cuenta de sectores por clúster, que es de 8 bits (se usan 6 bits para el módulo).
Esto obligaba a usar clusters de 32 Kbytes (2_{15} bytes) con los usuales 512 bytes por sector (0.5K x 2_6). Así que el límite definitivo de FAT16 se situó en los **2 gigabytes** (2_{15} x 2_{16}).

• **fat32:** FAT32 fue la respuesta para superar el límite de tamaño de FAT16 al mismo tiempo que se mantenía la compatibilidad con MS-DOS en modo real. Microsoft decidió implementar una nueva generación de FAT utilizando direcciones de cluster de 32 bits (aunque sólo 28 de esos bits se utilizaban realmente). En teoría, esto debería permitir aproximadamente 268.435.538 *clusters*, arrojando tamaños de almacenamiento cercanos a los ocho terabytes. Sin embargo, debido a limitaciones en la utilidad *ScanDisk* de Microsoft, no se permite que FAT32 crezca más allá de 4.177.920 *clusters* por partición (es decir, unos 124 gigabytes). Posteriormente, Windows 2000 y XP situaron el límite de FAT32 en los **32 gigabytes**. Sin embargo el tamaño máximo de un archivo en FAT32 es **4 gigabytes** (limitado por el atributo del tamaño del archivo que tiene 32 bits).

• **NTFS**: el sistema de archivos estándar de Windows NT y de sus descendientes (las gamas 2000, 2003, XP,Vista y 7), las versiones 9x (MS-DOS, Windows 95, Windows 98 y Windows ME), no pueden leer este sistema de archivos de manera predeterminada, pero existen utilidades para salvar esta carencia. NTFS ha reemplazado al anterior sistema de ficheros de Microsoft, llamado FAT, común a MS-DOS y a las versiones tempranas de Windows. NTFS incorpora muchas mejoras sobre el sistema FAT como compatibilidad mejorada con metadatos, y el uso de estructura de datos avanzadas (árboles-B) para optimizar el rendimiento, estabilidad, y el aprovechamiento del espacio en disco, además de nuevas características adicionales, como la **seguridad**, las **listas de control de acceso** o el **registro de transacciones** (journaling). Es un sistema adecuado para las particiones de gran tamaño requeridas en estaciones de trabajo de alto rendimiento y servidores. Puede manejar volúmenes de, teóricamente, hasta 2_{64}–1 clústeres. En la práctica, el máximo volumen NTFS soportado es de 2_{32}–1 clústeres (aproximadamente **16 Terabytes** usando clústeres de 4KB). Permite la **compresión** y **encriptado** de los archivos.

- **ext:** Es una extensión del sistema de archivos Minix (de ahí su nombre), aunque más elaborada. Su desarrollo fue suspendido en favor de ext2.
- **ext2**: Puede decirse que actualmente esta segunda "extensión" del sistema Minix, es el modelo "oficial" y más extendido en el mundo Linux. Es un sistema de archivos de alto rendimiento que, en términos de velocidad y eficiencia, presenta las mejores prestaciones de entre los sistemas soportados por Linux, aunque no está exento de inconvenientes. Los límites son un máximo de **2 TB de archivo**, y de **4 TB de partición**
- **ext3**: Es una versión "journaling" del anterior. Es decir, un sistema capaz de anotar las "transacciones" (operaciones) realizadas, y en caso necesario, disponer de cierta capacidad de rehacer los últimos cambios. El journaling garantiza la consistencia de los archivos, ya que las operaciones de escritura/borrado de disco suponen distintos accesos y actualizaciones; sobre los datos y sobre los metadatos. En caso de accidente. Por ejemplo, un apagado súbito, algunas operaciones pueden quedar sin realizar, produciendo inconsistencias en el sistema de archivos.

  En estos casos, el sistema tradicional consiste en recurrir al empleo de utilidades (como fsck en Linux; fdisk en Windows, o el recover del DOS) que verifican la estructura lógica e intentan reparar los desperfectos (al menos evitar que sean origen de nuevos problemas). El problema es que con frecuencia tales utilidades no resuelven los errores. Además, las rutinas necesarias después del "crash" de un gran sistema pueden requerir horas o incluso días de proceso. En cambio los sistemas journaling pueden rehacer la actualización que quedó incompleta (o anularla totalmente) en cuestión de minutos. El resultado es que estos sistemas son mucho más rápidos en el rearranque después de una incidencia. Algo que es especialmente valorado en determinados entornos de explotación (pensemos por ejemplo, en un gran banco).
- **hpfs**: Es el "High Performance File System" utilizado por OS/2 de IBM.
- **iso9660**: Es el sistema de archivos utilizados en los CD-ROMs. Es un estándar para grabar archivos de datos en los dispositivos ópticos como CDs y DVDs implementado en todos los sistemas que montan estos dispositivos. Esta es la razón por la que los CDs y DVDs son medios realmente universales para difusión de información digital. Como no podía ser menos, Linux soporta las dos versiones del estándar: High Sierra y Rock Ridge.
- **jfs:** Al igual que ext3, jfs es un sistema de archivos "journalista". Desarrollado inicialmente por IBM para sus sistema OS/2 Warp, su código fue donado al Open Source Consortium a principios del 2000 y portado a Linux poco después. Se incluye en los kernel a partir de la versión 2.5.6, aunque puede instalarse como un parche en las versiones 2.4.x. Es un sistema muy adecuado para entornos empresariales que permite volúmenes muy grandes y utiliza técnicas avanzadas para mejorar el rendimiento. Una de sus características es que utiliza un sistema de asignación dinámica para los "inodes") que asigna o libera el espacio según las necesidades. La ventaja es que, al contrario que en otros sistemas. Por ejemplo, ext2, no es necesaria una estimación previa del espacio que se reservará a los inodes al crear el sistema de archivos.
- **minix**: Es el sistema de archivos nativo del SO de igual nombre. Fue el primero de los disponibles para Linux, ya que como hemos indicado, el desarrollo original de éste se basó en aquel. Aunque adolece de importantes limitaciones (Minix es más un sistema pensado para la enseñanza que para explotación comercial), se sigue utilizando en disquetes y en unidades de disco RAM.
- **msdos**: Es la versión Linux del sistema de archivos DOS tradicionalmente usado por los PCs IBM, Windows y PS/2. La versión Linux adolece de las mismas limitaciones que el DOS original.

- **umsdos**: Una extensión Linux del anterior que elimina alguna de sus limitaciones, como la de nombres cortos (8+2). Al tiempo que añade algunas capacidades nuevas. Por ejemplo, la designación de dispositivos de E/S como archivos (algo muy común en el mundo Unix); posibilidad de definir tuberías ("pipelines") para utilizar la salida de un comando como entrada de otro. etc.
- **nfs**: Es el sistema de archivos de red ("Network File System") que permite acceder archivos situados en computadores remotos.
- **ncpfs**: Un sistema de archivos de red que soporta el protocolo NCP ("NetWare Core Protocol") de Novell. Permite acceder y manejar los recursos de un servidor NetWare desde Linux.
- **reiserfs**: Otro sistema de archivos "journalista" de código abierto y disponible en la mayoría de distribuciones de Linux, diseñado por Hans Reiser y colaboradores en Namesys. Aunque presenta unas excelentes prestaciones en el manejo de archivos grandes, una de sus características distintivas es su adecuación para manejar infinidad de archivos pequeños. Su filosofía es que los archivos pequeños estimulan la sencillez del código. ¡En lugar de utilizar una base de datos para manejar su información, utilice el sistema de archivos! reiserfs resulta de 8 a 15 veces más rápido que ext2 en el manejo de archivos de menos de 1 KB, y con la configuración adecuada, puede almacenar un 6% más de datos que aquel en el mismo dispositivo. En lugar de manejar clusters de tamaño fijo, su diseño le permite utilizar el espacio exactamente a medida de la necesidad. Los metadatos están organizados en una estructura de árbol ordenado bastante sofisticada, que además de gestionar los metadatos, almacena y comprime los despuntes (trozos de archivo que no alcanzan a ocupar un cluster).
- **smb**: Otro sistema de archivos de red, que soporta SMB ("Server Message Block"), un protocolo utilizado por Windows para trabajo en grupo (Windows 3.1); Windows NT, y Lan Manager de IBM. Permite compartir archivos; impresoras; puertos serie y otras abstracciones utilizadas para comunicación entre computadores, como tuberías y buzones de correo.
- **sysv**: Es una implementación Linux del sistema de archivos SystemV de Unix. Soporta las variedades Xenix FS (versión Microsoft de Unix); SystemV/Coherent; SystemV/386 y Coherent FS.
- **xfs**: En concordancia con la política de apoyo a Linux emprendida por algunas grandes compañías, Silicon Graphics, más conocida como SGI, ha distribuido una versión Linux de su sistema de archivos XFS, diseñado originariamente para sus sistemas IRIX. En concordancia con sus orígenes, se trata de un sistema "journalista" de altas prestaciones, que soporta bitácora o registro de las transacciones con los metadatos, así como "granjas" de discos ("disk farms") extremadamente grandes. Uno de estos sistemas es capaz de gestionar 18,000 Petabytes y un solo archivo puede alcanzar 9,000 Petabytes.
- **xiafs**: Fue otra extensión del sistema de archivos inicial (Minix) con intención de mejorar sus conocidas deficiencias. Su desarrollo y mantenimiento fueron abandonados a partir de la versión 2.1.21 del kernel.
- **squashfs**: Un sistema de archivos de solo-lectura altamente comprimido, implementado como un módulo del kernel bajo VFS ("Virtual File System"). Puede ser útil en aplicaciones embebidas que requieran gestionar gran cantidad de datos de forma eficiente.