El Sistema de Archivos

Este capítulo trata sobre la gestión del almacenamiento en un sistema operativo. Nos centraremos en la organización de la información en los sistemas de archivos por parte del SO para abstraer al usuario de las complejidades hardware de la amplia variedad tecnológica en la que están basados estos dispositivos.

Tabla de contenido

El Sistema de Archivos

1 Introducción

Una necesidad evidente de casi cualquier aplicación informática es la del almacenamiento y posterior recuperación de información. Los medios de almacenamiento masivo (disco duro, CD-ROM, memorias flash, ...) posibilitan su almacenamiento y recuperación.

Hay dos problemas a la hora de almacenar y recuperar información de un medio de almacenamiento permanente:

1. La naturaleza tecnológica de los medios de almacenamiento (disco duro magnético, disco duro de estado sólido, unidades ópticas) es muy diferente. Resulta imprescindible introducir algo que permita abstraer las propiedades físicas y tecnológicas de los dispositivos de almacenamiento, es decir, algo que permita manipular la información de la misma forma, independientemente del medio en que ésta esté almacenada.

2. Grandes cantidades de información deben estar estructuradas para facilitar su manipulación

La solución para el primer problema se encuentra en los controladores software del dispositivo, que hacen posible abstraer los complejos detalles de los dispositivos. El controlador software se encarga de atender las solicitudes que haga el sistema operativo al dispositivo correspondiente, actuando como capa colchón entre ambos. Así, el sistema operativo puede olvidarse de conceptos como pistas, sectores, cilindros, etc... y centrarse en lo que realmente interesa, almacenar y acceder a la información.

Para el segundo problema surgen los sistemas de archivos, como mecanismo o herramienta de gestión de la información sobre los dispositivos de almacenamiento. Una parte fundamental del sistema operativo es el sistema de archivos, el cual se encarga de

administrar y facilitar el uso de los dispositivos del almacenamiento.

La gestión de la información se realiza:

- ✔ Nivel lógico ➜ El sistema de archivos.
- ✔ Nivel físico ➜ El dispositivo de almacenamiento
- ✔ Capa intermedia entre nivel lógico y físico ➜ Los controladores de dispositivos.

2 El sistema de archivos

El sistema de archivos es el componente del sistema operativo encargado de administrar y facilitar el uso de los dispositivos de almacenamiento. Estructuran la información guardada en un dispositivo de almacenamiento, que luego será representada ya sea textual o gráficamente utilizando un gestor de archivos. Sus principales funciones son:

- ✔ La asignación de espacio de almacenamiento a los archivos
- ✔ La administración del espacio libre de la partición.
- ✔ La administración del acceso a los datos guardados.
- ✔ Seguridad o permisos: listas de control de acceso (ACLs) por usuarios y grupos.
- ✔ Asignación de los atributos extendidos (ej.: archivos de solo lectura, etc.)
- ✔ Mecanismo para evitar la fragmentación (desperdicio de espacio en disco).
- ✔ Integridad del sistema de archivos con sistemas de archivos transaccionales (*Journaling*)
- ✔ Soporte para cuotas de discos.

Como una partición puede almacenar miles de archivos, estos necesitan estructurarse para su rápida localización. De todo esto se encarga el sistema de archivos para lo que debe definir lo siguiente:

- ✔ La estructura del almacenamiento, generalmente en forma de árbol con carpetas y subcarpetas.
- ✔ La nomenclatura de los archivos. Caracteres que se admiten en el nombre de los archivos, longitud máxima, etc.

La mayoría de los sistemas operativos manejan su propio sistema de archivos. Generalmente, cada sistema de archivos ha sido diseñado para obtener el mejor rendimiento con un sistema operativo concreto, sin embargo, es usual que el mismo sistema operativo sea capaz de reconocer y gestionar múltiples sistemas de archivos.

2.1 Elementos en un sistema de archivos

En un sistema de archivos se manejan los siguientes elementos:

- ✔ Partición ➜ Cada una de las divisiones de un dispositivo de almacenamiento y a la que se asigna un sistema de archivos. La operación de crear y asignar un sistema de archivos en una partición se denomina formateo. Hasta que una partición no tiene un sistema de archivos creado no se puede almacenar información.
- ✔ Volumen ➜ En los sistemas operativos modernos se emplea el volumen como conjunto de particiones a las que se asigna un sistema de archivos y un nombre. La ventaja de los volúmenes sobre las particiones es que pueden ocupar espacio no contiguo en el dispositivo de almacenamiento y pueden ser redimensionados de forma flexible además de permitir ciertas configuraciones para proveer mayores tasas de acceso al disco, seguridad e integridad de la información.
- ✔ Archivo ➜ Conjunto de información relacionada que se almacena como flujo de bits en el dispositivo de almacenamiento y es tratada por el sistema de archivos como entidad única.
- ✔ Bloque, cluster, unidad de asignación, grupo ➜ La unidad mínima de almacenamiento de un archivo en una partición. Está formada por uno o varios sectores contiguos del dispositivo de almacenamiento. Esto quiere decir que el espacio real ocupado por un archivo en disco será siempre múltiplo del tamaño del grupo.
- ✔ Directorio ➜ El directorio en un sistema de archivos es un índice con información de los archivos y carpetas de la partición o volumen. Generalmente contiene el nombre del archivo y el inicio del espacio en dispositivo ocupado por el archivo.

2.2 Metadatos

Además de alojar los datos de los archivos, el sistema de archivo también almacena y manipula información muy importante sobre los archivos y el propio sistema de archivo, como por ejemplo las marcas de fecha y hora, el propietario del archivo, los permisos de acceso, el tamaño de los archivos y la localización del archivo en el disco, entre otras cosas. Esta información constituye lo que comúnmente denominamos **metadatos**. Estos metadatos generalmente hacen referencia a los atributos de un archivo. Cada sistema de archivos tienen su propio conjunto de atributos de archivo, pero en general son los siguientes:

- ✔ Nombre.- Nombre simbólico del archivo para su referencia por las aplicaciones.
- ✔ Identificador.- Etiqueta unívoca dentro del sistema de archivos y no accesible por el usuario.
- ✔ Tipo.- Tipo de información que almacena el archivo.
- ✔ Ubicación en el dispositivo.- Bloques del dispositivo donde se almacena el archivo.
- ✔ Tamaño.- Tamaño del archivo expresado como un número de bloques en el dispositivo.
- ✔ Fecha hora de creación y/o modificación.- Fecha y hora en la que el archivo se creó y cuando se modificó por última vez.

- ✔ ACL.- Lista de control de acceso. Recoge los permisos que autorizan, o deniegan, a los usuarios a realizar operaciones sobre el archivo.
- ✔ Propietario.- En sistemas multiusuario el usuario que creo el archivo y lo posee.

2.3 Sistemas de archivos transaccional

Para mejorar el rendimiento de las operaciones de E/S, los metadatos del sistema de archivos son temporalmente almacenados en la memoria RAM. Los problemas surgen si hay un corte de suministro eléctrico antes que los datos modificados en la memoria sean grabados nuevamente al disco. Se generaría una inconsistencia en el estado global del sistema de archivos.

Si la caída del sistema toma al controlador escribiendo en un área de los metadatos, tal como el directorio, por ejemplo, entonces en el problema es aún peor. Ahora en vez de un archivo dañado, habrá un sistema de archivo corrupto, y podría perder el directorio completo y todos los datos de una partición del disco.

El sistema de archivos estándar procura prevenir y recuperarse de una corrupción de los metadatos mediante el análisis exhaustivo del sistema de archivo durante el arranque de la máquina. Dado que el sistema de archivos mantiene copias redundantes de los metadatos críticos, es extremadamente improbable que estos datos se pierdan totalmente. El sistema evalúa dónde se hayan los metadatos corruptos, y luego repara el daño, ya sea copiando de la versión redundante o simplemente borrando el archivo o los archivos cuyos metadatos hayan sido estropeados.

Obviamente, cuanto más grande es el sistema de archivos a verificar, tanto más prolongado será el proceso de revisión. En una partición de varios gigabytes esta revisión de los metadatos durante el arranque puede tardar muchísimo tiempo.

Como las aplicaciones son cada vez más complejas, tanto en grandes servidores como en requisitos de disponibilidad mayores, existe la necesidad de un sistema de archivos más sofisticado que haga aún mejor la tarea de proteger los datos y los metadatos. Los sistemas de archivos transaccionales son una respuesta a esta necesidad.

Los sistemas de archivos más modernos han tomado prestadas del ámbito de las bases de datos las técnicas transaccionales, a fin de mejorar la recuperación de los datos tras una eventual caída. En estos sistemas de archivos las transacciones en disco se escriben de manera secuencial en un área llamada *journal* o *log*, antes de ser escritas en sus localizaciones finales dentro del sistema de archivo. Las distintas variantes de sistema de archivo transaccional difieren en términos de los datos que se escriben en el *log*. Algunas sólo escriben los metadatos del sistema de archivo; mientras que otras lo registran todo en el *journal*.

Ahora, si la caída del sistema ocurriera antes de que la entrada en el *log* hubiese sido efectuada con éxito, los datos originales todavía aparecerían intactos en el disco, y solamente se perderían los cambios más recientes. Si la caída se diera justo durante la actualización real del disco (es decir, después de que la entrada en el *log* se hubiera dado), la entrada en el *journal* mostraría lo qué supuestamente debió de haber ocurrido. De tal

suerte que cuando el sistema arranque de nuevo, pueda simplemente rehacer las entradas en el *journal* y completar la actualización que fue interrumpida.

En cualquier caso, se obtienen datos válidos y no una partición completa estropeada. Y como el tiempo de recuperación asociado con este sistema basado en *log* es muchísimo más corto, la máquina estará en línea en unos cuantos segundos.

Es importante observar también que el empleo de un sistema de archivo transaccional no vuelve obsoleto, ni mucho menos, el uso de los programas que sirven para revisar el sistema de archivo. Los errores de hardware y software que al azar llegan a corromper algunos bloques del sistema de archivo, generalmente no se pueden recuperar con el simple *log* de la transacciones en disco.

2.4 Fragmentación

Prácticamente, todos los sistemas de archivos introducen el concepto de grupo, cluster, bloque lógico o unidad de asignación. Cada cluster puede almacenar información de un sólo archivo, aunque un archivo puede ocupar varios clusters, los cuales no están necesariamente contiguos. Estas características provocan el problema de la fragmentación. Hay dos tipos de fragmentación que veremos a continuación.

- ✔ Fragmentación interna ➜ Ocurren cuando un archivo tiene un tamaño que no es múltiplo del tamaño de bloque, lo que significa que el último bloque no estará totalmente ocupado por el archivo y se desperdicia parte del espacio del bloque. Este problema se agudiza cuanto mayor sea el tamaño del bloque. Por ello, los archivos consumen más espacio del que realmente necesitan.

- ✔ Fragmentación externa ➜ Cuando se crea un archivo se asignan bloques libres a éste. Estos bloques asignados pueden estar desperdigados por todo el área de datos del disco duro. Normalmente, el sistema de archivos buscará bloques contiguos y dependiendo del tamaño del archivo lo encontrará más o menos que estén contiguos. Conforme se añaden nuevos archivos los fragmentos de bloques contiguos se harán más pequeños hasta que alcanzan un tamaño que no permiten ser asignados más que a archivos pequeños. Si un archivo es muy grande y no puede ser colocado de forma contigua ocupará bloques discontinuos desperdigados por todo el disco. Esto implica una ralentización a la hora de leer los datos del archivo ya que obligará al cabezal de lectura y escritura a moverse con más frecuencia para reunir los fragmentos desperdigados a los largo del disco.

Todos los sistemas de archivos sufren ambos tipos de fragmentación, pero la fragmentación externa lo sufre especialmente NTFS y FAT, y en menor medida ext2,3,4. Algunos sistemas operativos incluyen herramientas para desfragmentar el disco duro, que intenta compactar los archivos de forma que ocupen espacio contiguo. Sin embargo, esta operación supone la paralización del sistema mientras se realiza y poco después el disco volverá a estar fragmentado.

2.5 Gestión del espacio libre

El espacio en disco es limitada, por tanto es necesario reutilizar el espacio usado por los archivos eliminados para los nuevos archivos. Para controlar el espacio libre en disco, el sistema mantiene una lista de espacio libre, que registra todos los bloques del disco que están libres, es decir, que no están asignados a un archivo. Para crear un archivo, se recorre la lista de espacio libre hasta encontrar espacio suficiente, y lo asignamos al nuevo archivo; luego se elimina este espacio de la lista. Al eliminar un archivo se agrega su espacio en disco a la lista de espacio libre.

Para gestionar el espacio libre disponemos de las siguientes estrategias:

- ✔ Mapa de bits ➜ En muchos casos la lista se espacio libre se implementa como un mapa de bits o un vector de bits, donde cada bloque se representa con un bit. El bit es 0 si el bloque está libre, si está ocupado, el bit es 1.
- ✔ Lista enlazada ➜ Otra estrategia consiste en enlazar todos los bloques libres del disco, manteniendo un puntero al primer bloque libre, el cual contiene otro puntero al siguiente bloque libre, etc.
- ✔ Agrupamiento ➜ Una variante de la estrategia de la lista enlazada es almacenar en el primer bloque las direcciones de *n* bloques libres. Los primeros *n-1* bloques están libres y el último contiene la dirección en disco de otro bloque que contiene las direcciones de otros *n* bloques libres.
- ✔ Recuento ➜ Consiste en mantener una lista donde cada elemento es una dirección de bloque libre y un número *n* con los bloques libres contiguos siguientes. Así, cada entrada de la lista de espacio libre consiste en una dirección en disco y un recuento o contador.

2.6 Asignación de archivos

Una de las aspectos más cruciales en la gestión de los discos es asignar bloques a un archivo. El problema consiste en cómo asignar espacio a estos archivos para que se utilice eficazmente el espacio en disco y se pueda acceder a los archivos con rapidez. Generalmente un sistema de archivos solo emplea un método de asignación. Existen tres métodos de uso común para asignar el espacio en disco:

- ✔ Asignación contigua ➜ Cada archivo se almacena como un conjunto de bloques contiguos en disco. Tiene la ventaja de la rapidez en el acceso al archivo y el inconveniente de encontrar *n* bloques contiguos libres para un nuevo archivo. Además los archivos no pueden crecer y hay fragmentación externa.
- ✔ Asignación enlazada ➜ Cada archivo es una lista enlazada de bloques de disco que pueden encontrarse en cualquier parte del disco. El directorio contiene un puntero al primer y último bloque del archivo. Cada bloque del archivo contiene un puntero al siguiente. Para leer un archivo, basta con leer los bloques siguiendo los punteros, aunque el acceso a un bloque concreto requiere la lectura de los bloques que le preceden.

- ✔ Asignación enlazada con tabla en memoria ➜ Consiste en mantener en memoria una tabla con un elemento por cada bloque del disco. El elemento *i* contiene el índice del siguiente bloque del archivo. El elemento del último bloque del archivo contiene -1 o un marcador especial. A esta tabla se le conoce como FAT (*File Allocation Table*). El número del primer bloque del archivo se mantiene en el directorio.

- ✔ Asignación indexado ➜ Cada archivo tiene su propio bloque de índices, el cual es un vector de direcciones de bloques en disco. La entrada *i* en el bloque de índices apunta al bloque *i* del archivo. El directorio contiene la dirección del bloque de índices. Para leer el bloque *i*, usamos el puntero de la entrada *i* del bloque de índices para localizar y leer el bloque deseado. La asignación indexada permite el acceso directo, sin problemas de fragmentación externa, ya que puede usarse cualquier bloque del disco para atender una solicitud de espacio adicional.

En la asignación indexada cada archivo debe tener un bloque de índices, pero su tamaño limitaría el tamaño del archivo. Para tratar con esta situación se sigue una de las siguientes estrategias:

- ✔ Esquema enlazado ➜ Se mantienen varios bloques índice por archivo. Cada uno tiene el último puntero apuntando al siguiente bloque índice.

- ✔ Índice multinivel. Una variante del anterior consiste en emplear un bloque de índices (nivel 1) que apunte a los bloques de índices (nivel 2), que a su vez apunta a los bloques del archivo. Para lograr el acceso a un bloque, el sistema operativo utiliza el índice del primer nivel para encontrar el de segundo nivel, y así localizar el bloque de datos.

- ✔ Esquema combinado. Otra alternativa que se usa en el sistema Linux/UNIX es combinar en un bloque punteros a bloques de datos y punteros a bloques índices. De esta manera no es necesario un bloque de índices aparte para archivos pequeños. Los punteros a bloques índices apuntan a un índice multinivel lo que garantiza acceso rápido a cualquier bloque de un archivo de gran tamaño.

3 Sistemas de archivos más conocidos

A continuación se describen los sistemas de archivos más conocidos. El sistema FAT y sus variantes se han empleado en sistemas operativos de Microsoft desde MS-DOS y se mantiene en Windows, hasta su reemplazo por NTFS. Los sistemas de archivos ext2,3 y 4 son los habituales en los sistemas GNU/Linux.

3.1 Sistema de archivos FAT

Este sistema de ficheros se basa en una tabla de asignación de archivos (FAT *File Allocation Table*) que actúa de índice para localizar un archivo en el dispositivo de almacenamiento. En ella se almacenan los grupos o cluster utilizados por cada archivo, los grupos libres y los defectuosos.

No hace falta decir que en unidades de gran capacidad, la tabla es necesariamente

muy grande. Generalmente se carga en memoria para agilizar los procesos, ya que es de uso constante y cualquier operación de lectura/escritura tiene que utilizarla.

Existen varios tipos de este sistema de ficheros dado que ha ido evolucionando y adaptándose. Primero surgió la FAT de 12 bits, posteriormente la de 16 bits y finalmente la de 32 bits. Estos tres tipos de sistemas de ficheros poseen la misma estructura lógica y su funcionamiento es, en esencia, el mismo.

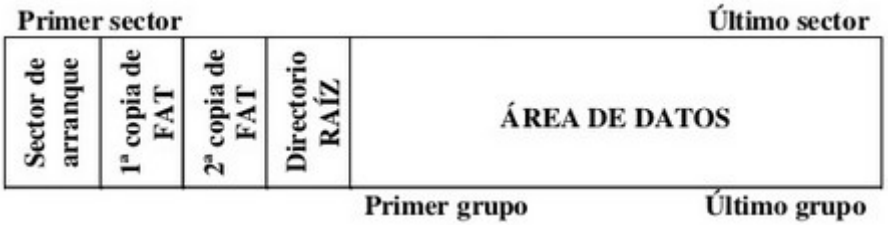


Figura 1.- Estructura del sistema de archivos FAT

La estructura del sistema de ficheros FAT está formada por:

- ✔ El sector de arranque.- Siempre es el primer sector de la partición (volumen) e incluye información básica, punteros a las demás secciones, y la dirección de la rutina de arranque del sistema operativo.
- ✔ Área FAT.- Contiene dos copias de la tabla de asignación de archivos (por motivos de seguridad). Estos son mapas de la partición, indicando qué clusters están ocupados por los archivos.
- ✔ Directorio raíz.- Es el índice principal de carpetas y archivos.
- ✔ Área de datos.- Es el lugar donde se almacena el contenido de archivos y carpetas.

3.1.1 La tabla FAT

La tabla FAT es un índice de todos los clusters existentes en el sistema de archivos. Mediante este índice se tiene información del estado de cada cluster y de los clusters ocupados por cada archivo. Esta tabla se encuentra a continuación del sector de arranque y su copia inmediatamente adyacente.

En la tabla FAT existe un campo o celda (de 12, 16 o 32 bits, según la versión de FAT) por cada cluster de la partición denominado entrada. El campo n de la FAT representa el cluster n de la partición.

El sistema FAT de 32 bits tiene las entradas de 32 bits, aunque los 4 primeros están reservados y no se utilizan, por lo que su capacidad de gestión es de 2^{28} = 268.435.456 clusters distintos. Esto implica que, su máxima capacidad es de 8 TB, considerando un grupo de tamaño máximo teórico de 32 KB. Sin embargo, en la práctica FAT32 solamente permite particiones de 2 TB como máximo.

Cada entrada de la FAT puede indicar:

- ✔ Si el cluster correspondiente está disponible.
- ✔ Si el cluster correspondiente está en uso.

- ✔ Si el cluster correspondiente está dañado.
- ✔ Si es el último cluster de un archivo.
- ✔ Si es el número del siguiente cluster de un fichero.

Hay que darse cuenta de la importancia de esta tabla, ya que su pérdida significa la pérdida de todos los datos del sistema de ficheros. Por ello esta tabla se encuentra duplicada, a continuación de la tabla original, para intentar asegurar su integridad. Todas las operaciones sobre ficheros que impliquen la modificación de la tabla original provocan la misma modificación en la tabla copia. Tan sólo en el caso en que se descubran defectos en la tabla original se procederá a sustituirlos, si es posible, por los datos guardados en la tabla copia.

3.1.2 El directorio raíz

El directorio raíz es un índice con información de los archivos y directorios del volumen. Se encuentra a continuación de la copia de la tabla FAT. Además en el directorio raíz se almacena también la etiqueta del disco. Cada entrada en un directorio es de 47 bytes en FAT32 y contiene información como el nombre y extensión del archivo, sus atributos, el espacio reservado, fecha y hora de la última modificación, el tamaño y el número del primer cluster. Para el tamaño se reservan 4 bytes (32 bits), de ahí que el tamaño máximo de archivo sea de 2^{32} = 4GB.

El número del primer cluster de cada archivo, junto con otros datos del mismo, se anota como entrada en el directorio, como mencionamos anteriormente. A su vez, la entrada correspondiente a dicho primer cluster en la tabla FAT, contiene el número de segundo cluster del fichero, cuya entrada contiene el número del tercero, etc. El proceso se repite sucesivamente con todos los cluster del fichero hasta llegar al último, cuya entrada no puede contener el número del siguiente, porque no existe, de forma que contiene una marca especial EOF (*End of file*) de final de fichero.

Hay que darse cuenta de que es en la entrada de directorio dónde se encuentra el primer clúster dónde se encuentran los datos del fichero. Por lo que una modificación o pérdida de este campo o de la entrada en sí provoca la pérdida de los datos del fichero.

3.1.3 El área de datos

El área de datos almacena todos los subdirectorios y ficheros de los usuarios. Ocupa aproximadamente, el 98 % del espacio del sistema. Tamaño de bloque lo escoge el sistema operativo en función del tamaño a direccionar.

Tamaño de la partición	Tamaño de Cluster
< 128 Mb	512 bytes
128 Mb – 256 Mb	
256 Mb – 512 Mb	4 Kb

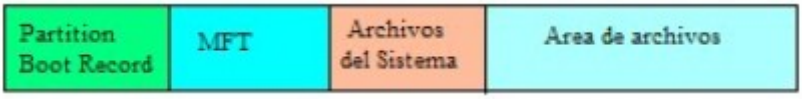
512 Mb – 1 Gb	
1 Gb – 2 Gb	
2 Gb – 8 Gb	
8 Gb – 16 Gb	8 Kb
16 Gb – 32 Gb	16 Kb
32 Gb – 2 Tb	32 Kb

3.2 Sistemas de archivos NTFS

NTFS (*New Technology File System*) es un sistema de archivos de Microsoft incluido en todas las versiones de Windows desde 2000 y XP. Reemplaza al sistema de archivos FAT en sistemas Windows con el objetivo de crear un sistema de archivos eficiente, robusto y con seguridad incorporada desde su base. Entre sus características están:

- ✔ NTFS permite definir el tamaño del clúster de forma independiente al tamaño de la partición.
- ✔ El tamaño mínimo del bloque es de 512 bytes.
- ✔ Este sistema también admite compresión nativa de archivos y encriptación.
- ✔ Permite el uso de nombres extensos, aunque, a diferencia de FAT32, distingue entre mayúsculas y minúsculas.
- ✔ Es un sistema ideal para particiones de gran tamaño. El tamaño mínimo de la partición es de 10 GB y el máximo recomendado en la práctica es 2 TB.
- ✔ Puede manejar volúmenes de hasta 16 TB usando clústeres de 4 KiB.
- ✔ Utiliza una estructura de datos avanzadas (árboles-B) para optimizar el rendimiento, ofrecer mejor estabilidad y aprovechamiento del espacio en disco.
- ✔ Contempla el uso de las listas de control de acceso para garantizar la seguridad de los archivos.
- ✔ Realiza registro de transacciones (Journaling). El concepto de journaling se refiere a que si se arranca el sistema sin haberlo cerrado correctamente no es necesario hacer un chequeo ya que la recuperación sucede de forma automática a partir de su último estado. NTFS es un sistema seguro ante fallos que puede auto corregirse en casi todas las situaciones.

Su principal inconveniente es que necesita para sí mismo una buena cantidad de espacio en disco duro, por lo que no es recomendable su uso en discos con menos de 400 MB. La estructura de una partición con el sistema de archivos NTFS es la siguiente:



- ✔ Boot Partition Record.- En los primeros 8Kb se almacena la información sobre el volumen (tipo de partición, capacidad, etc.), junto con el bloque del código básico para iniciar al sistema operativo.
- ✔ Área MFT (*Metadata File Table*). La tabla maestra contiene el donde y el como están almacenados los archivos junto con todos los atributos asociados a estos.
- ✔ Archivos del Sistema.- Contiene la información sobre los datos y operaciones que se realizan sobre el sistema de archivos: espacio libre, log de transaccionalidad, etc.
- ✔ Área de archivos.- Donde realmente se almacenan los datos del usuario.

En NTFS puede tenerse tamaño de cluster desde 512 bytes hasta 64 KBytes. Los discos NTFS se dividen, simbólicamente, en dos partes:

1. El primer 12% del disco se asigna al área MFT, un espacio en el cual crecen los metafile.
2. El resto es el área de datos. No obstante lo dicho anteriormente cualquier informe del SO acerca del espacio libre en el disco incluye el área MFT. El mecanismo del área MFT es el siguiente: cuando no haya espacio para almacenar mas archivos se toma espacio del área MFT y se reduce su longitud una vez que vuelve a existir espacio el área MFT vuelve a crecer.

3.2.1 MFT

La MFT se describe como una *base de datos relacional* compuesta de filas (los registros) y columnas (los atributos). Habrá un registro (entrada en la tabla) por cada archivo que exista en el sistema de archivos. Ahora bien, el concepto de archivo es bastante amplio: toda entidad almacenada individualmente es considerada un archivo. La propia MFT es considerada un archivo y tiene su propia entrada en la MFT. Las primeras 16 entradas de la MFT están reservadas para almacenar archivos del sistema, es decir, información sobre el propio sistema de archivos. El resto de entradas se corresponden con los archivos y carpetas de datos.

La MFT es un directorio centralizado que contiene información de todos los archivos del disco incluyéndolo a él mismo. Este archivo se divide en registros de longitud fija (usualmente un 1 KByte) y cada registro corresponde a algún archivo. También se le denomina *descriptor* y una copia suya se almacena en el segundo registro. El tercer registro contiene el archivo de registro. Este es un archivo que contiene todas las acciones llevadas a cabo en la partición para el caso de necesitar hacer una recuperación si ocurre un fallo.

A continuación se muestran los registros correspondientes a los archivos del sistema.

Archivos de metadatos almacenados en la MFT

Sistema de archivos	Nombre del archivo	MFT Record	Finalidad del fichero
Tabla maestra de archivos	$ Mft	0	Contiene un registro de archivo base para cada archivo y carpeta en un volumen NTFS. Si la información de asignación de un archivo o carpeta es demasiado grande para caber en un solo registro, otros registros de archivo se asignan también.
Espejo tabla maestra de archivos	$ Su espejo	1	Garantiza el acceso a la MFT en caso de una falla de un solo sector. Es una imagen duplicada de los primeros cuatro registros de la MFT.
Archivo de registro	$ Archivo de registro	2	Contiene información utilizada por NTFS de recuperación más rápido. El archivo de registro es utilizado por Windows Server 2003 para restaurar la consistencia de metadatos NTFS después de un fallo del sistema. El tamaño del archivo de registro depende del tamaño del volumen, pero se puede aumentar el tamaño del archivo de registro mediante el comando Chkdsk.
Volumen	$ Volumen	3	Contiene información sobre el volumen, tales como la etiqueta de volumen y la versión volumen.
Definiciones de atributos	$ AttrDef	4	Listas atribuyen nombres, números y descripciones.
Root índice de nombre de archivo		5	La carpeta raíz.
Bitmap Cluster	$ Bitmap	6	Representa el volumen, mostrando clústeres libres y sin usar.
Arranque sector	$ Boot	7	Incluye el BPB usado para montar el volumen y el código del gestor de arranque adicional que se utiliza cuando el volumen es de arranque.
Archivos de clúster no	$ BadClus	8	Contiene sectores defectuosos de un volumen.
Archivo de seguridad	$ Secure	9	Contiene los descriptores de seguridad únicos para todos los archivos de un volumen.
Mesa upcase	$ Upcase	10	Convierte los caracteres en minúsculas a mayúsculas juego Unicode.
Extensión de archivo NTFS	$ Extend	11	Se utiliza para varias extensiones opcionales, tales como las cuotas, los datos de punto de análisis, y los identificadores de objetos.
		12-15	Reservado para uso futuro.

Figura 2.- Metafiles en NTFS

Los siguientes registros, que constituyen lo que se conoce como el núcleo, hacen referencia a cada archivo y directorio de la partición en la forma de objetos con atributos. Esto implica que la información que concierne a cada archivo se almacena en un archivo y éste se registra dentro de la MFT.

Toda la información acerca de los archivos (todo menos los datos) se almacena en

registros MFT: su nombre, longitud, posición en el disco, etc., para lo cual se pueden usar uno o varios registros MFT que no tienen que estar contiguos. Los datos del archivo pueden estar en un registro MFT, por ejemplo un archivo sin datos o un archivo pequeño que pudiera estar dentro del límite de registros MFT.

3.2.2 Los archivos del sistema

Hemos visto que los 16 primeros archivos en la MFT son archivos del sistema, cada uno de ellos es responsable de algún aspecto del SO y contienen información del sistema de archivos, no siendo accesibles para el SO. Estos archivos se denominan *metafiles* y sus nombres comienzan por $. El primero y más importante es el $MTF que hace referencia al área MFT que está situado en ésta. Estos primeros 16 elementos de la MFT constituyen la única parte del disco que tiene una posición fija. Resulta interesante notar que existe una segunda copia de los tres archivos registros y está almacenada exactamente en la mitad del disco, el resto de los *metafiles* se pueden almacenar en cualquier otro archivo que resida en cualquier parte del disco.

3.3 Sistema de archivos ext2, ext3 y ext4

Los sistemas de archivos ext2 (*second extended filesystem*), ext3 (*third extended filesystem*) y ext4 (*fourth extended filesystem*) son los sistemas de archivos que se han utilizado por el kernel de Linux. Estos sistema de ficheros se basa, principalmente, en el concepto de *inodos* y en la utilización de mapas de bits para la gestión del espacio libre.

Son de los sistemas de ficheros más completos y más eficientes, en cuanto a los objetivos a conseguir. Por el contrario, también son bastante más complejo en cuanto a su estructura y a su funcionamiento. Su estructura es la siguiente:

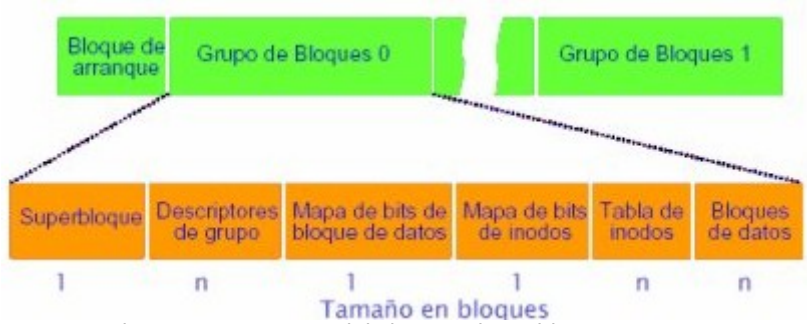


Figura 3.- Estructura del sistema de archivos ext2,3 y 4

Las diferencias entre ext2, ext3 y ext4 con las siguientes.

Sistema de archivos	Características
ext2	✔ No contempla journaling. ✔ Tamaño máximo de partición es de 32 Terabytes. ✔ Tamaño máximo de archivo es de 2 TB.
ext3	✔ Permite journaling, por tanto más seguro y fiable. ✔ Tamaño máximo de partición y archivo igual que ext2.
ext4	✔ Permite journaling. ✔ Tamaño máximo de partición de 1EB.

	✔ Tamaño máximo de archivo de 16 TB. ✔ Permite escritura retrasada.

Ahora veremos los elementos de su estructura.

3.3.1 Grupos de bloques

El primer bloque (bloque de arranque) nunca es manejado por el sistema de archivos, dado que esta reservado para el sector de arranque. El resto de la partición se divide en grupos de bloques, reducen la fragmentación, dado que el kernel intenta mantener los bloques de datos de un archivo en el mismo grupo de bloques, si es posible. Cada bloque en el grupo contiene algunas de las siguientes piezas de información:

- ✔ Una copia del superbloque del sistema de archivos.
- ✔ Una copia del grupo de descriptores de grupos de bloques.
- ✔ Un mapa de bits de bloque.
- ✔ Un grupo de inodos.
- ✔ Un mapa de bits de inodos.
- ✔ Área de datos

Tanto el superbloque como los descriptores de grupo están duplicados en cada grupo de bloques. Sólo el superbloque y los descriptores de grupos incluidos en el grupo de bloques 0 son utilizados por el kernel, mientras que las demás copias se dejan sin modificar; de hecho, nunca las mira.

Cuando se realiza una comprobación de consistencia del sistema de archivos, referencia el superbloque y los descriptores de grupos de bloques de grupo 0 copiándolos en el resto de grupos de bloques. Si se produce una corrupción de datos, y el superbloque y los descriptores de grupos del grupo 0 se hacen inválidos, se puede referenciar las copias de otros grupos diferentes del 0. Usualmente, las copias redundantes tienen suficiente información para permitir al programa de comprobación retornar la partición a un estado consistente.

3.3.2 El superbloque

El superbloque forma base del sistema de ficheros, la información contenida aquí permite al administrador del sistema de ficheros usar y mantener el sistema. Normalmente sólo se lee el superbloque del grupo de bloque 0 cuando se monta el sistema de ficheros pero cada grupo de bloque contiene una copia duplicada en caso de que se corrompa sistema de ficheros. Fundamentalmente tiene la siguiente información:

- ✔ Tamaño total del sistema de archivos, en bloques o nodos-i.
- ✔ Número de bloques libres del sistema.
- ✔ Número de bloques reservados a nodos-I.

- ✔ Número de nodos-I libres.
- ✔ Dirección del primer bloque de datos.
- ✔ Tamaño de un bloque de datos.
- ✔ Tamaño de un bloque parcial de datos.
- ✔ Hora de la última modificación sistema archivos.
- ✔ Hora integración (montaje) del sistema.
- ✔ Número de versión del sistema.
- ✔ Hora de la última verificación del sistema.

3.3.3 Descriptores de grupo

Cada grupo de bloques tiene una estructura de datos que lo describe denominada **descriptores de grupos**. Como el superbloque, todos los descriptores de grupo se duplican en cada grupo de bloque en caso de corrupción del sistema de fichero. Cada descriptor de grupo contiene la siguiente información:

- ✔ Mapa de bits de los inodos y los bloques de datos.
- ✔ Tabla de inodos.- Contiene el número del primer bloque de la tabla de inodos.
- ✔ Área de datos.- Contiene los bloques de datos de los archivos.

Los **mapas de bits** sirven para determinar el estado, libre o usado, de una parte del sistema de ficheros. En este sistema de ficheros existen dos mapas de bits:

- ✔ El mapa de bits de inodos.
- ✔ El mapa de bits de bloques de datos

Existe una relación entre cada bit del mapa de bits y cada bloque o inodo. De esta forma, si el bit 305 del mapa de bits está marcado como usado, quiere decir que el bloque de datos o el inodo 305 está usado. Así, se consigue localizar bloques o inodos libres, de forma rápida y eficiente, cuando sean necesarios.

3.3.4 Inodo

El inodo o nodo índice es la estructura de datos básica empleada por este sistema de ficheros para localizar la información relativa a cada fichero. Este sistema de ficheros mantiene en cada grupo descriptor todos los inodos en una tabla conocida como tabla de inodos situada a continuación del mapa de bits.

Un inodo contiene toda la información acerca del fichero que representa y tiene acceso, en mayor o menor medida, a los bloques usados por el fichero. Cada fichero guarda su información en un *inodo*, por lo que el número máximo de ficheros está limitado por el número de *inodos* existentes, aunque existan muchos bloques libres en el sistema.

La estructura de un *inodo* es fija y ocupa 128 bytes. Los principales campos de un *inodo* son:

Campo	Tamaño
Tipo y protección	2 bytes
Identificador de usuario	2 bytes
Tamaño del fichero (en bytes)	4 bytes
Fecha último acceso	4 bytes
Fecha última modificación inodo	4 bytes
Fecha última modificación datos	4 bytes
Identificador del grupo de usuario	2 bytes
Número de enlaces	2 bytes
Punteros a bloques	52 bytes

Nótese que el nombre del fichero no se encuentra en el inodo correspondiente, sino que sólo se encuentra en las entradas de directorio. Cada entrada de directorio almacena el nombre del archivo y el número de i-nodo.

Los punteros a bloques indican los números de bloque dónde se encuentran los datos del fichero. Estos punteros tienen la siguiente organización:

- ✔ 10 punteros a bloques de datos, llamados punteros directos.
- ✔ 1 puntero a un bloque con punteros directos a datos, llamado puntero simple.
- ✔ 1 puntero a un bloque con punteros simples apuntando cada uno de ellos a bloques con punteros directos, llamado puntero doble.
- ✔ 1 puntero a un bloque de datos con punteros dobles apuntando a bloques con punteros simples, que a su vez, apuntan a bloques con punteros directos, llamado puntero triple.

Los tres punteros indirectos dependen del tamaño de bloque para su funcionamiento. Dado que, cada uno de ellos apunta a un bloque que contiene punteros, hay que saber cuántos caben en un bloque, y para ello es necesario saber cuál es el tamaño de bloque del sistema. Una vez conocido, supongamos 4 KB, se divide éste por el tamaño de un puntero (4 bytes) y se obtiene el número de punteros por bloque, en este caso 1024 punteros.

3.3.5 Área de datos

El área de datos es la parte del sistema de ficheros destinada a almacenar los datos de los ficheros. Se encuentra a continuación de los *inodos* y ocupa la mayor parte del sistema de ficheros, cerca del 90%. En esta área se encuentra tanto los datos de los ficheros (definidos por el usuario) como los bloques de punteros usados por los *inodos*.

Un directorio se comporta, como ya hemos visto anteriormente como un archivo. El contenido de los directorios no apunta directamente a bloques de datos de los ficheros, sino a un *inodo* dónde se encuentra la información del fichero.

El número de bytes por entrada del directorio depende de la versión, ya que hay

versiones en los que la longitud del nombre de un archivo puede llegar a 255 bytes y otras en las que sólo llega a 14. Pero, todas las entradas contienen:

- ✔ El número de inodo dónde se encuentran los datos del fichero, situado en los dos primeros bytes.
- ✔ El nombre del archivo situado en el resto de la entrada.

Además, cualquier directorio contiene un mínimo de dos entradas:

- ✔ Una referencia a si mismo (.).
- ✔ Una referencia al directorio de que depende o directorio padre (..), salvo en el directorio raíz, que al ser el comienzo de la estructura, esta referencia también es sobre sí mismo.

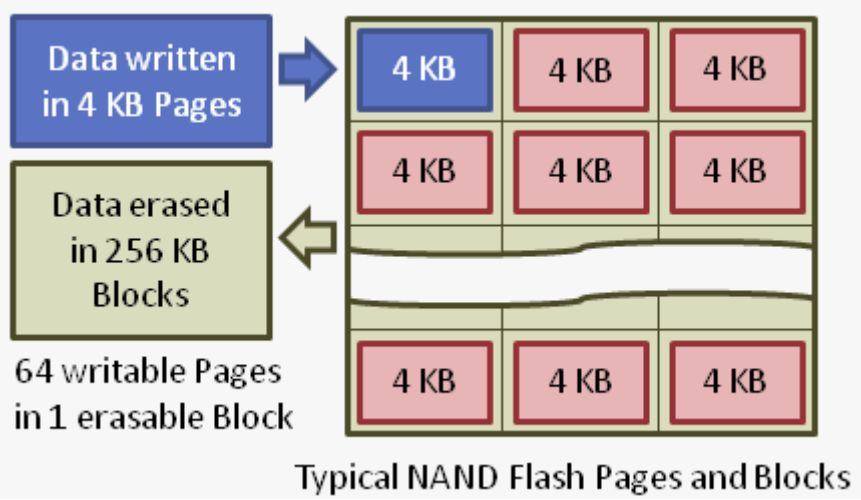
4 Sistemas de archivos en SSD

Los sistemas de archivos se pensaron para trabajar y gestionar sus archivos según las funcionalidades de un disco duro magnético. Sin embargo, este método de gestión de los archivos no es eficaz para en los discos SSD, ya que no aprovecha todo el potencial de este tipo de discos y además algunas características de los sistemas operativos pensadas para los discos magnéticos son muy lesivas para los discos SSD.

Para solucionarlo, diferentes sistemas operativos optimizaron sus sistemas de archivos para trabajar eficientemente con unidades de estado sólido, cuando éstas eran detectadas como tales, en vez de como dispositivos de disco duro magnético. El objetivo es reducir el número de escrituras en un disco SDD ya que acorta la vida del disco.

4.1 TRIM

A diferencia de los discos duros magnéticos, las memorias flash de tipo NAND que se emplean para fabricar un SDD no pueden sobrescribir los datos existentes. Esto significa que primero se han de borrar los datos antiguos antes de escribir los nuevos. Dicha memoria flash esta dividida en bloques de 256 KB, que a su vez están divididos en páginas de 4 KB. La unidad mínima de escritura es una página, pero la unidad mínima de borrado es un bloque.



Conforme pasa el tiempo, el disco SSD irá escribiendo los datos de los archivos en páginas vacías, pero si necesita escribir datos en una página que contenía datos previos de un archivo borrado tendrá primero que borrar los datos antiguos de la página antes de escribir los nuevos. Como el borrado tiene que ser por un bloque completo hará lo siguiente:

1. Copiar las páginas válidas del bloque en una memoria intermedia.
2. Borrar todas las páginas del bloque.
3. Copiar la nueva página y las páginas guardadas en la memoria intermedia en el bloque.

Esto significa que conforme pasa el tiempo, el rendimiento del disco SSD se degrada cada vez más y más, ya que para cada escritura se ha de pasar por un ciclo de lectura-borrado-escritura. Esto se conoce como *amplificación de escritura*.

En ausencia de TRIM, los discos desconocen cuales de los bloques están siendo usados por un archivo o cuales están libres. Esto es debido a que cuando se borra un archivo lo único que hace el sistema operativo es marcar los bloques que ocupaba dicho fichero como borrados dentro del sistema de archivos. Pero en ningún momento informa al disco duro de esta operación. Esto significa que, con el paso del tiempo el rendimiento de un disco SSD irá cada vez a peor, por mucho espacio libre que quede en el sistema de archivos, debido a que el disco SSD desconoce esta información.

TRIM es la solución para este problema. TRIM es el nombre de un comando que el sistema operativo puede enviar al disco SSD para informarle cuales de los bloques están libres en el sistema de ficheros. El disco SSD utiliza esta información para desfragmentar internamente los bloques y así mantener páginas libres disponibles para ser escritas de forma rápida y eficiente.

4.2 Windows

Windows 7 viene optimizado de serie para manejar correctamente los SSD sin perder compatibilidad con los discos duros. El sistema detecta automáticamente si es unidad de estado sólido o disco duro, y cambia varias configuraciones; por ejemplo, desactiva

automáticamente el desfragmentador, el `Superfetch`, el `Readyboost`, cambia el sistema de arranque e introduce el comando `TRIM`, que prolonga la vida útil de los SSD e impide la degradación de su rendimiento.

Para saber si TRIM está activado hay que ejecutar el siguiente comando en la consola

```
fsutil behavior query disabledeletenotify
```

Si el resultado es 0 está activado, si por el contrario es 1 entonces está desactivado. Si aparece deshabilitado podemos habilitarlo con el siguiente comando

```
fsutil behavior set DisableDeleteNotify
```

4.3 Linux

Para comprobar si nuestro disco duro soporta TRIM ejecutamos el siguiente comando:

```
sudo hdparm -I /dev/sda | grep TRIM
```

Si aparece una línea con `TRIM supported` entonces el SDD soporta el comando TRIM. En el ejemplo anterior se emplea `/dev/sda` como archivo de dispositivo del SDD. Si fuera otro habría que indicarlo.

Para habilitar el `trim` en este disco duro editaremos el archivo `/etc/fstab` y ahí añadiremos `discard` a la línea de nuestro disco duro haciendo que quede de la siguiente forma:

```
UUID=19d2ac4f-7889-ff7a-7787-a7622d5e0080     /      ext4
discard,errors=remount-ro        1
```

Como se puede apreciar se ha añadido `discard` a las opciones de montaje. Al reiniciar TRIM quedará habilitado en este disco duro.

5 Bibliografía

TANENBAUM, A. T., *Sistemas Operativos Modernos – 3ª Edición*. 2009 Pearson Prentice Hall, ISBN 978-607-442-046-3