Universidad Nacional Autónoma de México

Facultad de Ingeniería

División de Ingeniería Eléctrica

Materia: Sistemas Operativos

Profesor: Ing. Gunnar Eyal Wolf Iszaevich

Semestre: 2025-2

Grupo: 6

Nums. de cuenta:

320257599

117002029

Alumnos:

Jiménez Ayala Yordi Josué

Valenzuela Ascencio Gustavo

Exposición: Sistemas de archivos NTFS y exFat

Fecha de entrega: 1ro de mayo del 2025

Introducción

Sistema de archivos

Un sistema de archivos es un conjunto de métodos y estructuras utilizados por el sistema operativo de una computadora para disponer de los datos de cualquier dispositivo de almacenamiento digital, así como para controlar el almacenamiento disponible y almacenar nuevos datos ahí.

El sistema de archivos, considerado en su totalidad, es una estructura organizada que representa los datos junto con los metadatos que los describen. Este sistema se establece en un dispositivo de almacenamiento durante el proceso de formateo. Para mejorar la eficiencia en el manejo de la información, los sistemas de archivos agrupan los sectores en bloques. Los sistemas modernos suelen utilizar bloques compuestos por entre 1 y 128 sectores, es decir, tamaños que van desde 512 hasta 65,536 bytes, estos bloques se llaman ***clusters***.

¿Por qué existen diferentes tipos de sistemas de archivos?

La primera razón es que no existe un sistema de archivos completamente multipropósito. Todos tienen sus respectivas ventajas y desventajas. Algunos sistemas son de uso general y otros tienen un propósito específico o una especialización hacia cierto tipo de dispositivo.

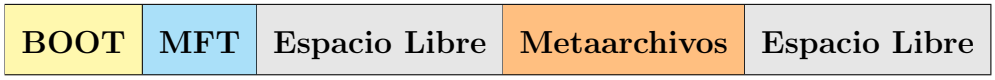
La segunda razón es su relación con el sistema operativo, debido a que los sistemas tienen su conjunto de sistemas de archivos compatibles. Debido a esto los SO de código abierto tienen muchos sistemas de archivos a elección, mientras que los de software propietario tienen muchas menos.

NTFS

NTFS (*New Technology File System*) es un sistema de archivos que usa Microsoft en sus sistemas Windows desde el año 1993. La razón de Microsoft para crearlo, respondía a la creación de Windows NT, sucesor de MS-DOS, debido a que el nuevo S.O de Microsoft requería funcionalidades multiusuario y el sistema de archivos de MS-DOS, FAT, no contaba con funcionalidades multiusuario por ejemplo: a qué usuario pertenece el archivo y los permisos que posee cada usuario sobre cada archivo.

¿Cómo se ve un volumen NTFS?

Cuando recién formateamos un volumen NTFS éste se verá de la siguiente manera.



Antes de explicar qué es cada parte de esta figura, debemos saber que todo objeto de NTFS es un archivo, además de definir qué es un metaarchivo.

Metaarchivo: Un metaarchivo es un archivo que tiene información sobre otros archivos o carpetas. Incluyen información como el nombre del archivo, la fecha de creación o modificación, el tamaño, permisos de acceso, entre otros.

Metaarchivos en NTFS: En el centro del disco se encuentran bastantes metadatos. Los de mayor interés son $MFTMirr y $Logfile. El MFT Mirror es una copia exacta de los 4 primeros registros de la MFT, sirviendo como respaldo en caso de que la MFT se dañe. El Logfile es el registro de todos los eventos que están esperando por ser escritos en el disco.

$Boot: Este archivo se encuentra al inicio del volumen y contiene la información para arrancar el sistema; en esta información se encuentra: el código de arranque del sistema, el tamaño del volumen, la referencia a la MFT, el tamaño de los sectores, etc. Este archivo es el único en el sistema que no puede ser movido.

$MFT: Contiene la información y de todos los archivos y directorios del volumen, incluyendo los propios, lo que la hace funcionar como un índice. A su vez, es el directorio raíz (root) del volumen.

Para prevenir que la MFT se fragmente, el S.O mantiene un *buffer* alrededor de ella, de modo que no se pdorán crear nuevos archivos en esta región hasta que todo el resto del espacio del disco esté en uso. El tamaño de este buffer es configurable y puede ser 12.5%, 25%, 37.5% o 50% del disco.

La MFT tiene un espacio reservado para una futura extensión. Los registros de la MFT del 12 al 15 se marcan como "usados", pero en realidad están vacíos. Los registros del 16 al 23 se marcan como "no utilizados".

$MFTMirr: Este es un archivo del sistema que duplica al menos los primeros cuatro registros de la MFT con el fin de recuperarlos.

Si el tamaño del cluster en el sistema es mayor de 4 veces del tamaño de un sector/registro, el tamaño de la MFT Mirror será del tamaño de un cluster. Es decir, si un sector es de 1024 bytes y un cluster es de 8192 bytes, la MFT Mirror clonará los primeros 8 registros de la MFT.

$Logfile: es un archivo de metadatos en NTFS que almacena un registro de transacciones realizadas en el volumen. Su propósito es permitir la recuperación automática del sistema de archivos en caso de fallos inesperados. Cuando se quiere escribir un archivo en el volumen, se debe actualizar el archivo junto a sus respectivas tablas de información en el sistema, esto representa una transacción. Si la transacción no se realiza debido a una falla, el sistema entra en un estado no definido, por lo tanto el usuario debe restaurarlo en un estado definido, haciendo un *rollback* con ayuda del LogFile.

$Bitmap: Este archivo lista los clusters en uso, Cada bit en este archivo representa un *Logical Cluster Number*, el cual es la posición de un cluster en memoria con respecto al inicio del volumen.

$BadClus: Este archivo contiene una lista de todos los *clusters* corruptos en el volumen. De manera lógica es una representación de todo el volumen, donde los archivos funcionales

se marcan con un cero y los *clusters* dañados son marcados con un apuntador a la zona de memoria con fallas.

Todo cluster dañado, será marcado como utilizado en el archivo $Bitmap con el fin de evitar que cualquier archivo sea escrito en esa localidad.

Este componente es un legado histórico de cómo funcionaban los discos duros cuando NTFS fue lanzado. Los discos duros actuales tienen "controladores inteligentes", los cuales marcan los sectores dañados antes de que los datos lleguen a la computadora, lo que permite la relocalización de regiones dañadas.

Funcionamiento de NTFS

Organización de los archivos en NTFS (Árbol B+)

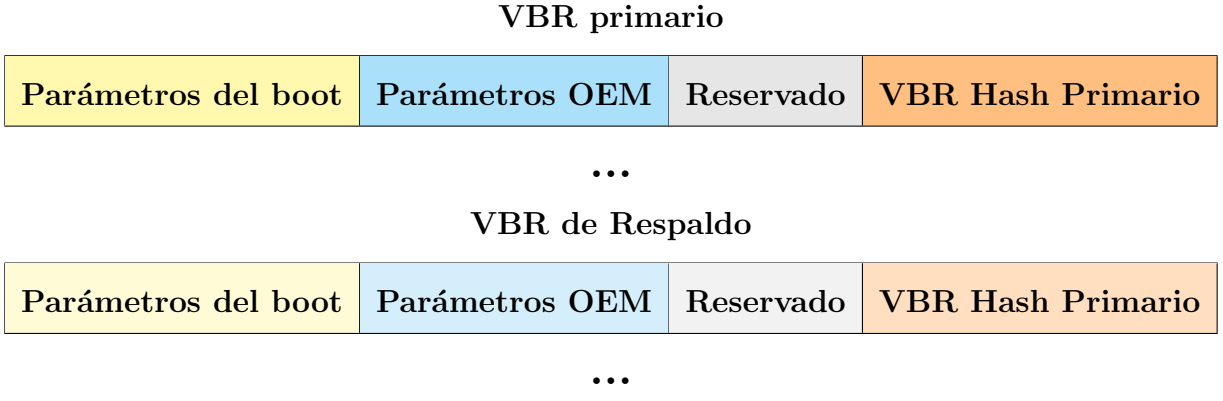
Un árbol B+ está diseñado para mantener los datos ordenados y permite búsquedas, inserciones y eliminaciones en tiempo logarítmico. En este tipo de árbol, los nodos internos no almacenan datos reales, sino únicamente claves que sirven para guiar la búsqueda, mientras que los nodos hoja contienen los datos propiamente dichos (por ejemplo, nombres de archivos o identificadores). Además, las hojas están conectadas entre sí en forma de lista enlazada, lo que facilita recorridos secuenciales eficientes.

Compresión: El algoritmo de compresión que utiliza NTFS se basa en identificar substrings que se encuentren más de una vez en la información del archivo para poder referenciarlas en lugar de escribirlas de manera explícita en el texto. Este algoritmo es una variante del algoritmo de compresión LZ77, llamada LZNT1.

exFat

exFat es un sistema de archivos perteneciente a la familia FAT que fue lanzado en 2006 por Microsoft. es un acrónimo de "Extended File Allocation Table", su arquitectura describe cómo se almacenan y recuperan los archivos de los dispositivos de almacenamiento y multimedia.

¿Cómo se ve un volumen exFAT?



Región FAT y región de datos

Volume Boot Record (VBR): Es la colección de sectores que define los límites y las ubicaciones de las regiones en exFAT. Contiene un apuntador al directorio Root. En total ocupa 12 sectores, de igual modo que lo hace su respaldo.

File Allocation Table (FAT): Es una estructura que actúa como un mapa para rastrear la secuencia de clústeres que componen un archivo. Cada entrada de 32 bits en la FAT representa un clúster del volumen y puede indicar si está libre, es parte de una cadena (apuntando al siguiente clúster), o marca el final del archivo.

Allocation Bitmap Table: La Allocation Bitmap Table en exFAT es una región especial que indica qué clústeres están en uso y cuáles están libres mediante una secuencia de bits, donde cada bit representa un clúster: un 1 indica que el clúster está ocupado, y un 0 que está libre. Este bitmap permite una gestión más eficiente del espacio libre en comparación con recorrer toda la FAT, acelerando operaciones como la asignación de nuevos clústeres.

Cluster Heap: Es un región de datos en el volumen del sistema. Cada archivo o directorio ocupa uno o más clústeres dentro del Cluster Heap, y su ubicación puede ser contigua o fragmentada, dependiendo del uso del espacio. La FAT (si se usa) o los metadatos del archivo en el directorio raíz indican la secuencia de clústeres utilizada. En esencia, el Cluster Heap es el "contenedor de datos" del volumen exFAT, y su correcta interpretación es crucial tanto para el funcionamiento del sistema como para análisis forense, ya que en él pueden encontrarse rastros de archivos eliminados o datos residuales no sobrescritos.

Directorio Root: El directorio Root es usado para definir archivos, sub-directorios, la etiqueta del volumen, la ubicación de la *UP-Case Table* y la ubicación del *Allocation bitmap*. Cada entrada dentro del directorio raíz (y de cualquier otro directorio en exFAT) ocupa 32 bytes y puede formar parte de una estructura compuesta de varias entradas consecutivas que describen un solo archivo.

UP-Case Table: La UP-Case Table es usada para convertir el nombre de los archivos a mayúsculas para ciertas operaciones de búsqueda y comparación.

Organización de los archivos de exFAT

Los archivos contienen listas lineales de entradas, cada una de exactamente 32 bytes, que describen archivos y subdirectorios almacenados en el sistema. Las entradas están ordenadas en el orden en que se crean, sin ningún tipo de ordenamiento o indexación avanzada.

Cada archivo o subdirectorio dentro de un directorio se representa mediante un bloque de entradas consecutivas. Este bloque está compuesto por una entrada principal, como la File Directory Entry, y varias entradas secundarias: una Stream Extension Entry que especifica detalles como tamaño y ubicación, y una o más File Name Entries que contienen el nombre del archivo en formato Unicode.

Dado que no se utiliza ningún tipo de indexación ni estructura de árbol balanceado, la búsqueda de archivos dentro de un directorio es secuencial: el sistema debe escanear una por una todas las entradas hasta encontrar la coincidencia deseada. Esto implica una complejidad computacional de $O(n)$

El diseño simple de exFAT se debe a su objetivo: ser liviano, eficiente y fácil de implementar en dispositivos portátiles con recursos limitados, como memorias USB, tarjetas SD y firmware de cámaras. Al evitar estructuras complejas, se reduce la carga en el procesador y se minimiza el número de escrituras, algo crucial para la durabilidad de medios flash que se desgastan con el tiempo.

Mejoras de exFAT respecto a FAT

Como se mencionó anteriormente, exFAT pertenece a la familia FAT, por lo tanto, al ser el sistema más reciente de la familia FAT, supone una mejora en distintos aspectos a FAT. Obviando mejoras como el tamaño máximo de un archivo y el tamaño máximo de un volumen, estas son las principales mejoras de exFAT respecto a FAT.

- **Tamaño de cluster flexible:** exFAT permite cambiar el tamaño del cluster, esto, gracias a que su tabla de asignación es más grande que la de FAT. En exFAT, el tamaño de esta tabla es de 64 bits, aunque regularmente se usan de 28 a 32. En FAT, además de estar limitados a un tamaño de cluster de 2^{28}, si queremos cambiar el tamaño del cluster, debemos formatear.
- **Relocalización de tablas FAT:** En exFAT las tablas FAT pueden ser relocalizadas, al contrario de FAT, en FAT se almacenan estrictamente al inicio del volumen.
- **Gestión del espacio libre:** Como se mencionó previamente, la tabla Bitmap se encarga de gestionar el espacio libre en el volumen. En FAT no sucede esto, cuando queremos consultar dónde hay espacio libre, debemos recorrer toda la FAT, lo cual hace al sistema más propenso a errores.

Todo muy bonito... Pero ¿por qué FAT sigue siendo el sistema más utilizado del mundo?

La antigüedad de FAT ha hecho que sea compatible con mútliples sistemas operativos y dispositivos, siendo un estándar ampliamente adoptado, incluso se puede ver que múltiples dispositivos como tarjetas SD y memorias USB vienen formateados por defecto en FAT, ya que esto garantiza compatibilidad inmediata.

Otra razón es su simplicidad. Como ya lo mencionamos, FAT es muy sencillo, esto permite que sea implementado sin problemas en dispositivos con recursos limitados y sistemas embebidos.

Finalmente, las patentes que poseía Microsoft sobre FAT ya expiraron, lo que hace a FAT libre de restricciones para su distribución y uso.

Referencias

[1] Linus Torvalds. "Linux". GitHub. 1-may-2025.

[Enlace: https://github.com/torvalds/linux/tree/master]. [Accedido 1-may-2025].

[2] Tuxera. "NTFS 3G". GitHub. 13-jun-2023.

[Enlace: https://github.com/torvalds/linux/tree/master] [Accedido 1-may-2025].

[3] Russon. R, Fledel. Y. "NTFS Documentation", pp 3-7, 35-50, 68-72, october 2024.

[Enlace: https://dubeyko.com/development/FileSystems/NTFS/ntfsdoc.pdf]

[4] Shullich. R "Reverse Engineering the Microsoft Extended FAT File System (exFAT)". GIAC, pp 22-47, december 2009.

[Enlace: https://dubeyko.com/development/FileSystems/NTFS/ntfsdoc.pdf]

[5] UFS Explorer. "Los fundamentos de los sistemas de archivos", november 2024.

[Enlace: https://bit.ly/42YP5MH]

[6] ChaN. "FAT Filesystem", october 2020.

[Enlace: http://elm-chan.org/docs/fat_e.html]