The history of basic file systems

# Introduction

In the twenty-first century humanity has introduced a breakthrough in computer technologies. The computing capability of a watch we wear on our hand every day is several hundred times more than the capability of all computers scientists had in the eighties. Gadgets have not only become a constant part of our lifestyle, but also an essential tool for work, studying and even art.

Computers, laptops and smartphones are being used by most humans every day, and with every click on the device we change something in device memory. We use our storage for everything from images, videos and music to game scores and contacts. We keep transferring files between different types of storages – CD and DVD disks, flash drives and memory cards. However, only few actually know, how memory works and how to use it with maximum efficiency.

As computers and all other devices work in binary system, each file is basically a line of ones and zeros. However, since the beginning of computer age, multiple ways to store those in the memory have been suggested and used. Knowing how some of the file systems work may allow everyone to use storage in a much more effective way.

# FAT

“FAT” stands for “File Allocation Table”, and it is one of the first file systems. A file allocation table is basically a table of contents of the disk that the operating system uses to look up the location of a file, even if the file is broken up in pieces (sectors) scattered across the disk's surface.

FAT has been in use since early eighties and since then several modifications and extensions of FAT had been introduced. Though nowadays FAT cannot deliver the same performance, reliability and scalability as some modern file systems do, it is still very popular. FAT supported for compatibility reasons by nearly all currently developed [operating systems](https://en.wikipedia.org/wiki/Operating_system) for personal computers and many [mobile devices](https://en.wikipedia.org/wiki/Mobile_device). FAT is a well-suited format for data exchange between computers and devices of almost any type and age from 1981 up to the present. FAT is no longer the default file system for usage on [Microsoft Windows](https://en.wikipedia.org/wiki/Microsoft_Windows) computers though, as much more powerful file systems have been introduced since.

As disk drives evolved, the capabilities of the file system have been extended accordingly, resulting in three major file system variants: [FAT12](https://en.wikipedia.org/wiki/File_Allocation_Table#FAT12), [FAT16](https://en.wikipedia.org/wiki/File_Allocation_Table#FAT16) and [FAT32](https://en.wikipedia.org/wiki/File_Allocation_Table#FAT32). The FAT standard has also been expanded in other ways while generally preserving backward compatibility with existing software.

FAT file system is characterized by the separation of the storage into areas of disk storage called clusters. File Allocation Table contains entries for each [cluster](https://en.wikipedia.org/wiki/Cluster_(file_system)). Each file can be only stored in an integer number of clusters. That’s why choosing a large size makes the computer use more memory than is needed, though it makes working with files several times faster.

The difference between FAT, FAT12, FAT16 and FAT32 is mainly in the number of bits for addressing each cluster in the memory. FAT has 8 bits per each address, FAT12 – 12 bits, FAT32 – 32 bits. Increasing the size of addresses obviously increases the number of clusters a disk can have, and the maximum size of the storage is defined by both cluster size and number of possible clusters. For example, Microsoft Windows doesn’t support any clusters larger than 64MB, meaning that the restriction of FAT32 drives is exactly 32GB.

# exFAT

 “exFAT” stands for “extended FAT”, or “extended file allocation table”. exFAT is another Microsoft file system which was specially optimized for USB Flash Drives and SD memory cards. In fact, exFAT has been adopted by the [SD Card Association](https://en.wikipedia.org/wiki/SD_Card_Association) as the default file system for [SDXC](https://en.wikipedia.org/wiki/SDXC) cards larger than 32 [GB](https://en.wikipedia.org/wiki/Gibibyte).

exFAT doesn’t have many disturbing issues of FAT. For example, file size isn’t restricted by 4GB – which is very useful, as continuous recording of HD video can exceed the 4 GB limit in less than an hour. It can hold both larger file sizes, larger partitions and larger numbers of files. It doesn’t, however, have journaling, which means a higher risk of data being lost while “unsafe ejection”.

exFAT introduces metadata integrity through the use of checksums. There are three checksums currently in use. The Volume Boot Record (VBR) is a 12 sector region that contains the boot records, BIOS Parameter Block (BPB), OEM parameters and the checksum sector. (There are two VBR type regions, the main VBR and the backup VBR.) The checksum sector is a checksum of the previous 11 sectors, with the exception of three bytes in the boot sector (Flags and percent used). This provides integrity of the VBR by determining if the VBR was modified. The most common cause could be a boot sector virus, but this would also catch any other corruption to the VBR.

A second checksum is used for the upcase table. This is a static table and should never change. Any corruption in the table could prevent files from being located because this table is used to convert the filenames to upper case when searching to locate a file.

The third checksum is in the directory file sets. Multiple directory records are used to define a single file and this is called a file set. This file set has metadata including the file name, time stamps, attributes, address of first cluster location of the data, file lengths, and the file name. A checksum is taken over the entire file set and a mismatch would occur if the directory file set was accidentally or maliciously changed.

When the file system is mounted, and the integrity check is conducted, these hashes are verified. Mounting also includes comparison of the version of the exFAT file system by the driver to make sure the driver is compatible, and to make sure that none of the required directory records are missing If any of these checks fail, the file system should not be mounted, although in certain cases it may mount read-only.

# NTFS

“NTFS” stands for “New Technology File System”, it was developed by Microsoft. Starting with [Windows NT 3.1](https://en.wikipedia.org/wiki/Windows_NT_3.1), it is the default file system of the [Windows NT](https://en.wikipedia.org/wiki/Windows_NT) family. NTFS is supported in other desktop and server operating systems as well. [Linux](https://en.wikipedia.org/wiki/Linux) and [BSD](https://en.wikipedia.org/wiki/BSD) have a [free and open-source](https://en.wikipedia.org/wiki/Free_and_open-source) NTFS driver, called [NTFS-3G](https://en.wikipedia.org/wiki/NTFS-3G), with both read and write functionality. [macOS](https://en.wikipedia.org/wiki/MacOS) comes with read-only support for NTFS; its disabled-by-default write support for NTFS is unstable.

The main feature of the file system is support of multiple data streams. NTFS also uses EFS (Encrypting File System) to store files in an encrypted and secure form.

Sparse file support allows the creation of very large files, but to consume disk space only as needed. A sparse file is a file with an attribute that causes the I/O subsystem to allocate the file's meaningful (nonzero) data. All nonzero data is allocated on disk, whereas all nonmeaningful data (large strings of data composed of zeros) is not. When a sparse file is read, allocated data is returned as it was stored, and non-allocated data is returned, by default, as zeros in accordance with the C2 security requirement specification.

NTFS is in many ways similar to HPFS file system. In fact, in the 80’s [Microsoft](https://en.wikipedia.org/wiki/Microsoft) and [IBM](https://en.wikipedia.org/wiki/IBM) formed a joint project to create the next generation of graphical [operating system](https://en.wikipedia.org/wiki/Operating_system); the result was [OS/2](https://en.wikipedia.org/wiki/OS/2) and [HPFS](https://en.wikipedia.org/wiki/High_Performance_File_System). Because Microsoft disagreed with IBM on many important issues they eventually separated: OS/2 remained an IBM project and Microsoft worked to develop [Windows NT](https://en.wikipedia.org/wiki/Windows_NT) and NTFS. When it was created, many concepts of HPFS were borrowed.

Probably as a result of this common ancestry, HPFS and NTFS use the same [disk partition](https://en.wikipedia.org/wiki/Disk_partitioning) identification type code. Using the same Partition ID Record Number is highly unusual, since there were dozens of unused code numbers available, and other major file systems have their own codes. For example, FAT has more than nine (one each for [FAT12](https://en.wikipedia.org/wiki/FAT12), [FAT16](https://en.wikipedia.org/wiki/FAT16), [FAT32](https://en.wikipedia.org/wiki/FAT32), etc.). Algorithms identifying the file system in a partition type 07 must perform additional checks to distinguish between HPFS and NTFS.

NTFS is optimized for 4 [KB](https://en.wikipedia.org/wiki/Kibibyte) [clusters](https://en.wikipedia.org/wiki/Data_cluster), but supports a maximum cluster size of 64 KB. The maximum NTFS volume size that the specification can support is 264 − 1 clusters, but not all implementations achieve this theoretical maximum.

NTFS is a [journaling file system](https://en.wikipedia.org/wiki/Journaling_file_system) and uses the NTFS Log to record metadata changes to the volume. It is a feature that FAT does not provide and critical for NTFS to ensure that its complex internal data structures will remain consistent in case of system crashes or data moves performed by the [defragmentation](https://en.wikipedia.org/wiki/Defragmentation) API, and allow easy rollback of uncommitted changes to these critical data structures when the volume is remounted. Notably affected structures are the volume allocation bitmap, modifications to [MFT](https://en.wikipedia.org/wiki/NTFS#Master_File_Table) records such as moves of some variable-length attributes stored in MFT records and attribute lists, and indices for directories and [security descriptors](https://en.wikipedia.org/wiki/Security_descriptor).

The [USN Journal](https://en.wikipedia.org/wiki/USN_Journal) (Update Sequence Number Journal) is a system management feature that records changes to files, streams and directories on the volume, as well as their various attributes and security settings. The journal is made available for applications to track changes to the volume. This journal can be enabled or disabled on non-system volumes.

The [hard link](https://en.wikipedia.org/wiki/Hard_link) feature allows different file names to directly refer to the same file contents. [Hard links](https://en.wikipedia.org/wiki/Hard_link) are similar to [directory junctions](https://en.wikipedia.org/wiki/NTFS#Reparse_points), but refer to files instead. Hard links may link only to files in the same volume, because each volume has its own [MFT](https://en.wikipedia.org/wiki/NTFS#Master_File_Table). Hard links have their own file metadata, so a change in file size or attributes under one hard link may not update the others until they are opened. Hard links were originally included to support the [POSIX](https://en.wikipedia.org/wiki/POSIX) subsystem in Windows NT.

# EXT

Both «EXT» and following «EXT2», «EXT3», «EXT4» are short names for «extended». These file systems were specially developed for computers based on linux kernel.

The early development of the Linux kernel was made as a cross-development under the [MINIX](https://en.wikipedia.org/wiki/MINIX) operating system. The [MINIX file system](https://en.wikipedia.org/wiki/MINIX_file_system) was used as Linux's first file system. The MINIX file system was mostly free of [bugs](https://en.wikipedia.org/wiki/Computer_bug), but used 16-bit offsets internally and thus had a maximum size limit of only 64 [megabytes](https://en.wikipedia.org/wiki/Megabyte), and there was also a filename length limit of 14 characters. Because of these limitations, work began on a replacement native file system for Linux.

[VFS](https://en.wikipedia.org/wiki/Virtual_file_system), a virtual file system layer, was added to the Linux kernel to ease the addition of new file systems and provide a generic file [API](https://en.wikipedia.org/wiki/Application_programming_interface). The extended file system (EXT), was released in April 1992 as the first file system using the VFS API and was included in Linux version 0.96c.

The ext file system solved the two major problems in the Minix file system (maximum partition size and filename length limitation to 14 characters), and allowed 2 [gigabytes](https://en.wikipedia.org/wiki/Gigabyte) of data and filenames of up to 255 characters. But it still had problems: there was no support of separate [timestamps](https://en.wikipedia.org/wiki/Time_code) for file access, [inode](https://en.wikipedia.org/wiki/Inode) modification, and data modification.

# EXT 2

As a solution for EXT problems, two new filesystems were developed in January 1993 for Linux kernel 0.99: [xiafs](https://en.wikipedia.org/wiki/Xiafs) and the second extended file system (ext2), which was an overhaul of the extended file system incorporating many ideas from the [Berkeley Fast File System](https://en.wikipedia.org/wiki/Berkeley_Fast_File_System). ext2 was also designed with extensibility in mind, with space left in many of its on-disk data structures for use by future versions.

Since then, ext2 has been a testbed for many of the new extensions to the VFS API. Features such as the withdrawn [POSIX](https://en.wikipedia.org/wiki/POSIX) [draft ACL proposal](https://en.wikipedia.org/wiki/Access_control_list) and the withdrawn [extended attribute](https://en.wikipedia.org/wiki/Extended_attribute) proposal were generally implemented first on ext2 because it was relatively simple to extend and its internals were well understood.

The space in ext2 is split up into [blocks](https://en.wikipedia.org/wiki/Block_(data_storage)). These blocks are grouped into block groups, analogous to [cylinder groups](https://en.wikipedia.org/w/index.php?title=Cylinder_group&action=edit&redlink=1) in the [Unix File System](https://en.wikipedia.org/wiki/Unix_File_System). There are typically thousands of blocks on a large file system. Data for any given file is typically contained within a single block group where possible. This is done to minimize the number of disk seeks when reading large amounts of contiguous data. Each block group contains a copy of the superblock and block group descriptor table, and all block groups contain a block bitmap, an inode bitmap, an inode table, and finally the actual data blocks.

Every file or directory is represented by an inode. The term "inode" comes from "index node" (over time, it became i-node and then inode). The inode includes data about the size, permission, ownership, and location on disk of the file or directory. The superblock contains important information that is crucial to the booting of the [operating system](https://en.wikipedia.org/wiki/Operating_system). Thus backup copies are made in multiple block groups in the file system. However, typically only the first copy of it, which is found at the first block of the file system, is used in the booting.

The group descriptor stores the location of the block bitmap, inode bitmap, and the start of the inode table for every block group. These, in turn, are stored in a group descriptor table.

# EXT 3

EXT3 is a next version of EXT2 file system. In comparison with EXT2, EXT3 has some features like journaling, online file system growth and specially developed more efficient HTree indexing for larger directories.

The performance (speed) of EXT3 is less attractive than competing Linux filesystems, such as EXT4, [JFS](https://en.wikipedia.org/wiki/JFS_(file_system)), [ReiserFS](https://en.wikipedia.org/wiki/ReiserFS), and [XFS](https://en.wikipedia.org/wiki/XFS), but it has a significant advantage in that it allows in-place upgrades from EXT2 without having to [backup](https://en.wikipedia.org/wiki/Backup) and restore data. Benchmarks suggest that ext3 also uses less CPU power. It is also considered safer than the other Linux file systems, due to its relative simplicity and wider testing base.

# EXT4

EXT4 is a successor to EXT3, it also has features like journaling and high performance. EXT4 first appeared when EXT3 developers wanted to develop some upgrades but didn’t want to effect EXT3’s stability. All upgrades were taken to a new file system called EXT4.

Main EXT4 improvement is considered to be a larger file system, which can support volumes with sizes up to 1 exbibyte and files with sizes up to 16 tebibytes. Traditional block mapping scheme from EXT2 and EXT3 has been replaced by «Extents» - much larger blocks which can map up to 128 MiB of space.

Another useful thing is that EXT2 and EXT3 storages can be mounted as EXT4. That also provides improvements in performance.