Introduction to File systems

File systems are one of the things any newcomer to linux must become acquainted with. In the world of Microsoft you never really have to worry about it, the default being NTFS. Linux however, being built on a world of open source and differing opinions, is not limited in this way and so the user should have an understanding of what a file system is, and how it affects the computer.

At the core of a computer, it's all 1s and 0s, but the organization of that data is not quite as simple. A *bit* is a 1 or a 0, a *byte* is composed of 8 bits, a kibibyte is 1024 (i.e. 2^10) bytes, a mebibyte is 1024 kibibytes and so on and so forth. All these *bits* and *bytes* are permanently stored on a Hard Drive. A hard drive stores all your data, any time you save a file, you're writing thousands of 1s and 0s to a metallic disc, changing the magnetic properties that can later be read as 1 or 0. There is so much data on a hard drive that there has to be some way to organize it, like a library of books and the old card drawers that indexed all of them, without that index, we'd be lost. Libraries, for the most part, use the Dewey Decimal System to organize their books, but there exist other systems to do so, none of which have attained the same fame as Mr. Dewey's invention. File systems are the same way. The ones most users are aware of are the ones Windows uses, the vFat or the NTFS systems, these are the Windows default file systems.

There are several different attributes which are necessary in defining file systems, these include their max file size, max partition size, whether they journal or not.

Journaling

A journaling file system is more reliable when it comes to data storage. Journaling file systems do not necessarily prevent corruption, but they do prevent inconsistency and are much faster at file system checks than non-journaled file systems. If a power failure happens while you are saving a file, the save will not complete and you end up with corrupted data and an inconsistent file system. Instead of actually writing directly to the part of the disk where the file is stored, a journaling file system first writes it to another part of the hard drive and notes the necessary changes to a log, then in the background it goes through each entry to the journal and begins to complete the task, and when the task is complete, it checks it off on the list. Thus the file system is always in a consistent state (the file got saved, the journal reports it as not completely saved, or the journal is inconsistent (but can be rebuilt from the file system)). Some journaling file systems can prevent corruption as well by writing data twice.

Table

Now below is a very brief comparison of the most common file systems in use with the Linux world.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| File System | Max File Size | Max Partition Size | Journaling | Notes |
| Fat16 | 2 GiB | 2 GiB | No | Legacy |
| Fat32 | 4 GiB | 8 TiB | No | Legacy |
| NTFS | 2 TiB | 256 TiB | Yes | (For Windows Compatibility) NTFS-3g is installed by default in Ubuntu, allowing Read/Write support |
| ext2 | 2 TiB | 32 TiB | No | Legacy |
| ext3 | 2 TiB | 32 TiB | Yes | Standard linux filesystem for many years. Best choice for super-standard installation. |
| ext4 | 16 TiB | 1 EiB | Yes | Modern iteration of ext3. Best choice for new installations where super-standard isn't necessary. |
| reiserFS | 8 TiB | 16 TiB | Yes | No longer well-maintained. |
| JFS | 4PiB | 32PiB | Yes (metadata) | Created by IBM - Not well maintained. |
| XFS | 8 EiB | 8 EiB | Yes (metadata) | Created by SGI. Best choice for a mix of stability and advanced journaling. |
| GiB = Gibibyte (1024 MiB) :: TiB = Tebibyte (1024 GiB) :: PiB = Pebibyte (1024 TiB) :: EiB = Exbibyte (1024 PiB) | | | | |

Above you'll see a brief comparison of two main attributes of different filesystems, the max file size and the largest a partition of that data can be.

Of the above file systems the only one you cannot install Linux on is the NTFS. It is not recommended to install Linux on any type of FAT file system, because FAT does not have any of the permissions of a true Unix FS.

Editing Files

Those used to a Windows file system (NTFS, FAT) know that it isn't normally possible to change files while they are open. This restriction does not exist in a Unix file system. This is because in Unix file systems, files are indexed by a number, called the inode, and each inode has several attributes associated with it, like permissions, name, etc. When you delete a file, what really happens is the inode is unlinked from the filename, but if some other program is using the file, it still has a link open to the OS, and will continue to be updated. A file is not really deleted until all links have been removed (even then, the data is still on the disk, but not indexed in anyway and thus very hard to recover). All of this means that you can delete executing programs while they're running without crashing and move files before they're finished downloading without corruption.

Fragmentation

Another common Windows practice that is not needed in Unix is defragmenting the hard drive. When NTFS and FAT write files to the hard drive, they don't always keep pieces (known as blocks) of files together. Therefore, to maintain the performance of the computer, the hard drive needs to be "defragged" every once in a while. This is unnecessary on Unix File systems due to the way it was designed. When ext3 was developed, it was coded so that it would keep blocks of files together or at least near each other.

No true defragmenting tools exist for the ext3 file system, but tools for defragmenting will be included with the ext4 file system.

See Also

[ConvertFilesystemToExt4](https://help.ubuntu.com/community/ConvertFilesystemToExt4) - A guide for converting existing EXT3 filesystems to EXT4.

Other Resources

1. [Wikipedia: Comparison of File Systems](http://en.wikipedia.org/wiki/Comparison_of_file_systems)
2. [Chris Browne's: File System's Explained](http://linuxfinances.info/info/fs.html)
3. [Fstab](https://help.ubuntu.com/community/Fstab)

Different File Systems on the same disk

If you are migrating from Windows and have more than one ntfs partition on the same disk, you might be tempted to convert one of the partitions to ext3 or 4 and install there the Linux files while leaving the ntfs partition as /home (<https://help.ubuntu.com/community/Partitioning/Home/Moving>).

Avoid this! it's ok to have two partitions on for Linux system files and another for home files in the same disk but they should be both the same filesystem.