

# An Overview of File Systems v1.1

By Robert Spotswood

What follows is a cursory summary of the more common Linux and Windows file systems you could put on one of your partitions. The list is far from complete on the Linux side.

File systems exist to allow you to store, retrieve, and manipulate data. There are two basic types of file systems, journaling and non-journaling. Journaling file systems add an extra few steps to every write. Before writing/changing any data to the hard drive, the file system driver writes to a journal what it is going to do, then does the write, then marks the journal entry as done (or erases it). All this extra writing means that journaling file systems' writes are generally slower than an equivalent non-journaling file systems'.

But, what you get for slower writes is seen most dramatically when there is a sudden, unexpected, reboot. In order to ensure the metadata (definition below) for a partition is at least consistent, a non-journaling file system must do a disk check over the entire partition(s). This can take minutes to hours, and the larger the partition, the longer it will take. A journaling file system, on the other hand, just looks at the journal, and replays any transactions that were not complete. This process usually takes only seconds no matter how large the partition. This also dramatically reduces, but does not eliminate, the chance of file corruption and data loss compared to non-journaling file systems.

Some types of journaling only keep a journal of the metadata. Metadata is data about where and how the files are stored. For FAT or NTFS file systems, the metadata is the File Allocation Table (FAT) or the Master File Table (MFT) respectively. Some journaling file systems can also journal the data itself. This means a big performance hit, but almost no chance of data loss except in the case of hard drive failure (or user error).

Some benchmarks of Linux file systems are available on-line at <http://linuxgazette.net/122/TWDT.html#piscz>. Another set of Linux benchmarks is available at <http://kerneltrap.org/node/715>, although these numbers are a bit old. The most important thing to keep in mind when viewing such benchmarks is which is more important: speed or keeping your data intact. Also, be sure to take the actual numbers and scales into account when viewing the graphs.

*Update:* Do not focus solely on performance numbers. With only a few exceptions, the most important thing about file systems is that they don't corrupt your data. Performance should be the number 2 priority. With that in mind, EXT3 performs much better than

ReiserFS and XFS (not covered) in the event of a sudden power failure. Both ReiserFS and XFS do logical journaling while EXT3 does physical block journaling. The type of journaling NTFS does is not clear.

When a PC experiences a sudden loss of power, it does not die right away, nor do all components die at the same time. The first to go is RAM. This get corrupted almost instantly. The hard drive subsystem, which includes controllers and the hard drives themselves, may continue to function for several hundred of milliseconds after the RAM is corrupt. If data is being written to the hard drives from RAM at this moment, garbage is actually written to the hard drive. As the EXT3 journal writes are slower, the window of opportunity for corruption is much, much smaller than for ReiserFS and XFS.

## **FAT16**

The original DOS file system, this has fallen into disuse and been largely replaced by FAT32. There are still a few applications that use FAT16 though (see FAT32 below). This is a non-journaling file system.

## **FAT32**

The replacement for FAT16, this is possibly the best balance between compatibility and features of all file systems. Virtually every modern OS can read and write this, making it the lowest common denominator. DOS (some versions), Windows 3.1, early Windows 95 (prior to OSR2 & 95b), and NT4 can not read FAT32. FAT32 supports long file names (FAT16 supported only 8 characters, then a period, and 3 more characters, "8.3") and more types of characters than FAT16. It first appeared in Windows 95 OSR2. This, and FAT16, are the only file systems supported by Windows 95 OSR2, Windows 98, and Windows ME.

FAT32 supports partitions up to 2 TiB (binary terabytes,  $2^{40}$ , now officially called tebibytes; see [http://linuxreviews.org/dictionary/Binary\\_prefixes/](http://linuxreviews.org/dictionary/Binary_prefixes/)), although the theoretical limit is 8 TiB. Windows 2000 & XP cannot format FAT32 volumes larger than 32 GiB ( $2^{30}$ ), but it can read/write larger FAT32 volumes that are formatted by other operating systems. Windows 2000 & XP won't warn you about this though and will go through the whole formatting process, then fail with a cryptic error message at the end ("Logical Disk Manager: Volume size too big."). The minimum size for a FAT32 partition is about 260 MiB. The largest possible file size for a FAT32 partition is 4 GiB minus 2 bytes.

It is not nearly as robust as the file systems that follow, does not support advanced security features, and is not journaling. During writes its lack of journaling does give it a speed advantage over journaling file systems, though. Nevertheless, it is well understood, and there are many tools available to help you deal with it. Windows 98 FAT32 tools, scandisk and chkdsk, will not work on FAT32 partitions formatted with Linux formatting tools, but the tools in Windows 2000 and later have no problems. FAT32 can be converted to NTFS, but this is a one way trip and won't perform as well as a partition that is formatted with NTFS from the start.

Almost all flash memory uses either the FAT16 or FAT32 file system, depending on the size of the drives. Card based flash devices tend to use FAT 16, while USB flash drives are generally FAT32. At least some digital cameras and other devices will not read FAT32 formatted flash memory.

## **NTFS**

NTFS is the newest Microsoft file system. It has journaling (but only metadata), file compression, encryption, quotas, and a fine degree of file and directory security. Overall, NTFS is much more efficient than FAT32 with disk space and performance is pretty good too. This is one of only two choices (FAT32 is the other) for Windows 2000 and higher and is the default.

You should never fill an NTFS drive more than 85% full, or the MFT (Master File Table) can become badly fragmented, making the drive slow. Windows XP pre-SP1 versions can not make partitions larger than 128 GiB. NTFS in Windows XP (SP1 or later) supports a maximum file size up to the capacity of the partition which can be up to 16 EiB ( $2^{60}$  bytes).

While NTFS has a lot going for it, it suffers from one serious drawback. The file system is proprietary and only Microsoft tools and those who pay the licensing fees can safely write to NTFS file systems. The Linux read drivers are quite stable, but only for uncompressed, unencrypted NTFS file systems. This proprietary nature makes repair and cleaning of NTFS partitions much more difficult (and impossible in some cases) than it has to be. Use NTFS's compression and encryption only with good backups or data you don't care about.

## **EXT2**

The standard Linux file system until the advent of EXT3 (below), EXT2 is a well understood, well tested, stable, and fast file system. Unlike NTFS, it is completely open and anyone can use it, although Microsoft has chosen not to support it. The EXT2 file

system has a maximum file size of 2 TiB, and a maximum partition size of 32 TiB in the more recent versions.

The EXT2 file system supports standard Unix file types: regular files, directories, device special files and symbolic links, and standard Unix permissions. The standard permissions can be extended to give even more granularity in file permissions. EXT2 also supports quotas and file attributes. The EXT2 (and EXT3) file systems keep some percentage of the disk space as "reserved" and only root (administrator) can write to this reserved area. The default behavior for is to reserve 5% of the disk space for root. This can be helpful in the event a disk fills up.

There are a great many tools to work with EXT2, most of them free. Included in this list are several ways to access EXT2 partitions from within Windows. Explore2fs (<http://uranus.it.swin.edu.au/~jn/linux/explore2fs.htm>) is a Windows Explorer like interface for reading EXT2 partitions for all versions of Windows since 95. Both Ext2fsd (<http://sourceforge.net/projects/ext2fsd>) and EXT2IFS (<http://uranus.it.swin.edu.au/~jn/linux/ext2ifs.htm>) provide a Windows 2000 and XP file system driver to allow native support (read and write) to EXT2 partitions. Both drivers should be considered beta quality.

The biggest drawback to EXT2 is the lack of journaling and native Windows support. The native support issue is completely Microsoft's fault. EXT2 partitions can be converted to EXT3. EXT2 does want a full disk check every so often, even if the system has been shut down cleanly. This can be quite time consuming and annoying.

### **EXT3**

EXT3 is the direct descendant of EXT2. Actually, it is EXT2 but with a journaling overlay. Unless contradicted here, everything said about EXT2 applies equally to EXT3.

EXT3 supports three different journaling modes, including one where all data is written to the journal, rather than just the metadata. EXT3's default journaling mode is slower than those from ReiserFS v3, because it journals data and not just metadata. By default, EXT3 does a sync (sync()) every 5 seconds. This is because EXT3 developers are paranoid about your data and prefer to care about your data rather than win on benchmarks. You shouldn't ever lose more than 5 seconds of work.

The default compile options (at least the ones used by several of the popular packagers) make it incompatible with large files (>4 GiB), although the current maximum file size limit for EXT3 is 2 TiB (same as EXT2). All the tools above for EXT2 also work on EXT3

partitions, with the limitation that any that don't work specifically with EXT3 will not use the journaling features.

Whether EXT3 or ReiserFS (see below) is a better file system has been the subject of more than a few arguments without any clear winner. Some claim data loss with one and never the other. There seem to be about equal numbers of each. It is likely that many of the most problems can be traced to physical hard drive issues.

Overall, EXT3 is still a reasonable choice for Linux users: mature, well supported, and good overall performance. The biggest drawback is the "mandatory" disk check every so often that EXT3 shares with EXT2, even though the journaling makes this unnecessary.

## **ReiserFS**

ReiserFS is another entry in the Linux journaling file system category. It was the first journaling file system for Linux, and some claim it is the most stable of them as a result of having been out the longest. The designers of ReiserFS made a choice to optimize it for small files. If you have lots of small files, this file system gives excellent performance and is very space efficient. There are settings that can be changed to improve performance at a slight expense of disk space. There are two versions in use today, version 3 (the more stable one), and version 4 (the current one).

ReiserFS was not designed for slow or really busy CPU's. A key part of the design was that Hans Reiser (main architect and programmer) realized that CPU's were vastly underused. I/O resources were maxed out while CPU's were sitting idle. So he found ways to use the CPU to make more efficient use of the I/O resources. ReiserFS is a modern file system meant to run on modern machines. It won't perform as well on older machines or busy CPU's.

ReiserFS v3 journals only the metadata, while v4 defaults to journaling everything -- file data as well as metadata. There are reports that ReiserFS v4 deals with the disk so intensely that it uncovers flaws and errors that other file systems may never find, or live with. However, one user did report that "the data recovery abilities of the [ReiserFS v4 disk checking program] appears to be nothing less than magic." Since ReiserFS v4 work is being sponsored by DARPA (the same organization that gave us the Internet), it is designed for military grade security.

If there is a significant drawback to ReiserFS, it is the lack of third party tools. However, unlike EXT2 and EXT3, there is never any "mandatory" disk checking. This makes

ReiserFS much more suited for devices that will be turned on and off often, such as certain USB hard drives.

There is a free Windows driver, rfsd (<http://rfsd.sourceforge.net/>) currently in beta status available to allow reading (but not writing!) ReiserFS from within Windows 2000 or later. Like EXT2 and EXT3, lack of native Windows support is the choice of Microsoft as the file system is open and free for use by everyone.

*Update:* Hans Reiser, the main developer of the file system, was convicted of first degree murder in 2008, leaving the future of the ReiserFS rather uncertain. The company that he founded to develop the file system has been put up for sale to pay his legal expenses, but has not been sold. However, work does continue of the file system by a group of volunteers.

## **Conclusion**

The perfect file system would actually be rather boring. Files would be where you want them to be, would never be corrupted, a power loss would mean no lost work (time waiting for power to be restored maybe, but not work) and no lost time waiting for a disk check to finish, extremely fast reading and writing, and compatibility would not be an issue.

Unfortunately such file a system does not yet exist. There is no clear winner, as all have major strengths as well as deficiencies in some areas. The correct choice is an individual decision. Even if the decision is to let someone else decide, you've still made a choice.