**Filesystems**

**What are filesystems?**

A *filesystem* is the methods and data structures that an operating system uses to keep track of files on a disk or partition; that is, the way the files are organized on the disk. The word is also used to refer to a partition or disk that is used to store the files or the type of the filesystem. Thus, one might say ``I have two filesystems'' meaning one has two partitions on which one stores files, or that one is using the ``extended filesystem'', meaning the type of the filesystem.

The difference between a disk or partition and the filesystem it contains is important. A few programs (including, reasonably enough, programs that create filesystems) operate directly on the raw sectors of a disk or partition; if there is an existing file system there it will be destroyed or seriously corrupted. Most programs operate on a filesystem, and therefore won't work on a partition that doesn't contain one (or that contains one of the wrong type).

Before a partition or disk can be used as a filesystem, it needs to be initialized, and the bookkeeping data structures need to be written to the disk. This process is called *making a filesystem*.

Most UNIX filesystem types have a similar general structure, although the exact details vary quite a bit. The central concepts are *superblock*, *inode* , *data block*, *directory block* , and *indirection block*. The superblock contains information about the filesystem as a whole, such as its size (the exact information here depends on the filesystem). An inode contains all information about a file, except its name. The name is stored in the directory, together with the number of the inode. A directory entry consists of a filename and the number of the inode which represents the file. The inode contains the numbers of several data blocks, which are used to store the data in the file. There is space only for a few data block numbers in the inode, however, and if more are needed, more space for pointers to the data blocks is allocated dynamically. These dynamically allocated blocks are indirect blocks; the name indicates that in order to find the data block, one has to find its number in the indirect block first.

UNIX filesystems usually allow one to create a *hole* in a file (this is done with the lseek() system call; check the manual page), which means that the filesystem just pretends that at a particular place in the file there is just zero bytes, but no actual disk sectors are reserved for that place in the file (this means that the file will use a bit less disk space). This happens especially often for small binaries, Linux shared libraries, some databases, and a few other special cases. (Holes are implemented by storing a special value as the address of the data block in the indirect block or inode. This special address means that no data block is allocated for that part of the file, ergo, there is a hole in the file.)

**Filesystems galore**

Linux supports several types of filesystems. As of this writing the most important ones are:

**minix**

The oldest, presumed to be the most reliable, but quite limited in features (some time stamps are missing, at most 30 character filenames) and restricted in capabilities (at most 64 MB per filesystem).

**xia**

A modified version of the minix filesystem that lifts the limits on the filenames and filesystem sizes, but does not otherwise introduce new features. It is not very popular, but is reported to work very well.

**ext3**

The ext3 filesystem has all the features of the ext2 filesystem. The difference is, journaling has been added. This improves performance and recovery time in case of a system crash. This has become more popular than ext2.

**ext2**

The most featureful of the native Linux filesystems. It is designed to be easily upwards compatible, so that new versions of the filesystem code do not require re-making the existing filesystems.

**ext**

An older version of ext2 that wasn't upwards compatible. It is hardly ever used in new installations any more, and most people have converted to ext2.

**reiserfs**

A more robust filesystem. Journaling is used which makes data loss less likely. Journaling is a mechanism whereby a record is kept of transaction which are to be performed, or which have been performed. This allows the filesystem to reconstruct itself fairly easily after damage caused by, for example, improper shutdowns.

**jfs**

JFS is a journaled filesystem designed by IBM to to work in high performance environments>

**xfs**

XFS was originally designed by Silicon Graphics to work as a 64-bit journaled filesystem. XFS was also designed to maintain high performance with large files and filesystems.

In addition, support for several foreign filesystems exists, to make it easier to exchange files with other operating systems. These foreign filesystems work just like native ones, except that they may be lacking in some usual UNIX features, or have curious limitations, or other oddities.

**msdos**

Compatibility with MS-DOS (and OS/2 and Windows NT) FAT filesystems.

**umsdos**

Extends the msdos filesystem driver under Linux to get long filenames, owners, permissions, links, and device files. This allows a normal msdos filesystem to be used as if it were a Linux one, thus removing the need for a separate partition for Linux.

**vfat**

This is an extension of the FAT filesystem known as FAT32. It supports larger disk sizes than FAT. Most MS Windows disks are vfat.

**iso9660**

The standard CD-ROM filesystem; the popular Rock Ridge extension to the CD-ROM standard that allows longer file names is supported automatically.

**nfs**

A networked filesystem that allows sharing a filesystem between many computers to allow easy access to the files from all of them.

**smbfs**

A networks filesystem which allows sharing of a filesystem with an MS Windows computer. It is compatible with the Windows file sharing protocols.

**hpfs**

The OS/2 filesystem.

**sysv**

SystemV/386, Coherent, and Xenix filesystems.

**NTFS**

The most advanced Microsoft journaled filesystem providing faster file access and stability over previous Microsoft filesystems.