NTFS

From Wikipedia, the free encyclopedia

NTFS	
Developer	Microsoft
Full name	New Technology File System
Introduced	July 1993 (Windows NT 3.1)
Partition identifier	0x07 (MBR) EBD0A0A2-B9E5-4433-87C0-68B6B72699C7 (GPT)
Structures	
Directory contents	B+ tree
File allocation	Bitmap/Extents
Bad blocks	Bitmap/Extents
Limits	
Max file size	16 TiB with current implementation (16 EiB architecturally)
Max number of files	4,294,967,295 (2^{32}-1)
Max filename size	255 characters
Max volume size	256 TiB with current implementation (16 EiB architecturally)
Allowed characters in filenames	Unicode (UTF-16), any character except "/"
Features	
Dates recorded	Creation, modification, POSIX change, access
Date range	January 1, 1601 - May 28, 60056
Forks	Yes
Attributes	Read-only, hidden, system, archive
File system permissions	ACLs
Transparent compression	Per-file, LZ77 (Windows NT 3.51 onward)
Transparent encryption	Per-file, DESX (Windows 2000 onward), Triple DES (Windows XP onward), AES (Windows XP Service Pack 1, Windows 2003 onward)
Supported operating systems	Windows NT family (Windows NT 3.1 to Windows NT 4.0, Windows 2000, Windows XP, Windows Server 2003, Windows Vista)

NTFS or **New Technology File System** is the standard file system of Windows NT and its descendants: Windows 2000, Windows XP, Windows Server 2003 and Windows Vista.

NTFS replaced Microsoft's previous FAT file system, used in MS-DOS and early versions of Windows. NTFS has several improvements over FAT such as improved support for metadata and the use of advanced data structures to improve performance, reliability and disk space utilization plus additional extensions such as security access control lists and file system journaling. The

exact specification is a trade secret of Microsoft.

NTFS has five versions: v1.0 (http://www.scotsnewsletter.com/07.htm) , v1.1 (http://www.pcguide.com/ref/hdd/file/ntfs/verNTFS11-c.html) and v1.2 found in NT 3.51 and NT 4, v3.0 found in Windows 2000 and v3.1 found in Windows XP, Windows Server 2003, and in current pre-release versions of Windows Vista (http://mlf.linux.rulez.org/mlf/ezaz/ntfsresize.html#vista) . These versions are sometimes referred to as v4.0, v5.0 and v5.1, after the version of Windows they ship with. Newer versions added extra features. For example, Windows 2000 introduced quotas.

Contents

- 1 Internals
  - 1.1 Interoperability
  - 1.2 Compatibility with FAT
- 2 Features
- 3 Limitations
- 4 Authors
- 5 Notes
- 6 References
- 7 See also
- 8 External links

Internals

In NTFS, everything that has anything to do with a file (file name, creation date, access permissions and even contents) is stored as metadata. This abstract approach allowed easy addition of filesystem features during the course of Windows NT's development — an interesting example is the addition of fields for indexing used by the Active Directory software. File names are stored in Unicode (encoded as UTF-16, although limited to the Basic Multilingual Plane in early versions before Windows 2000). The downside of this approach is that it can be difficult to recover from corruption of a disk.

Internally, NTFS uses B+ trees to index file system data. Although complex to implement, this allows faster access times in some cases. A file system journal is used in order to guarantee the integrity of the file system itself (but not of each individual file). Systems using NTFS are known to have improved reliability compared to FAT file systems.

The Master File Table (MFT) essentially contains metadata about every file and directory on an NTFS file system. It includes parameters such as location, size, and permissions. It is used to aid in minimizing disk fragmentation.

Interoperability

Details on the implementation's internals are closed, so third-party vendors have a difficult time providing tools to handle NTFS.

NTFS partitions can be read by Linux since Version 2.2.0. Linux 2.6 contains a new driver written by Anton Altaparmakov (Cambridge University) and Richard Russon. It offers limited write support. At this time (January 2006) it allows only rewriting and some cases of file resize. More write support is avalaible using ntfsmount[1], new userspace driver written by Yura Pakhuchiy in which files and directories can be created, overwritten, renamed, deleted, truncated, and expanded with limited success. Due to the complexity of the internal NTFS structures, both the

built-in 2.6.14 kernel driver and the FUSE driver will stop writing to the volume when it detects too many changes to be safe, thus it should not corrupt the volume. Full write support is available using Paragon[2]'s *NTFS for Linux 3* driver, although criticised for leaving many errors on the volume when mounted read-write. Alternatively the Windows driver `ntfs.sys` can be used with Captive NTFS. Recently, a new beta GPL opensource driver ntfs-3g[3] driver has been developed and is being tested by the linux-ntfs team. This driver is based on ntfsmount with extended support of directory index operations and at the moment offers "unlimited file creation and deletion", but it is not yet tested in the field. Upon successful testing code will be merged back into the ntfsmount. According to the developer, this userspace driver is twice as fast as the kernel ext3 driver (which is the native linux filesystem) and is 10 times faster than the commercial Paragon Software driver.

FreeBSD, and Mac OS X versions 10.3 and later, offer read-only NTFS support.

eComStation offers read-only NTFS support.

Compatibility with FAT

Microsoft currently provides a tool to convert the HPFS (only on Windows NT 3), the FAT16 and — on Windows 2000 and higher — also the FAT32 formats to NTFS, but not the other way around. PartitionMagic by Symantec, PartitionExpert by Acronis and the open source NTFSResize utility (http://mlf.linux.rulez.org/mlf/ezaz/ntfsresize.html) are all capable of safely resizing NTFS partitions. Microsoft has added the ability to resize (expand) NTFS partitions in Windows 2003 using the Diskpart command line tool.

For historical reasons, the versions of Windows that do not support NTFS all keep time internally as local zone time, and therefore so do all file systems other than NTFS that are supported by current versions of Windows. However, Windows NT and its descendants keep internal timestamps as GMT/UTC and make the appropriate conversions for display purposes. Therefore, NTFS timestamps are in GMT/UTC. This means that when files are copied or moved between NTFS and non-NTFS partitions, the OS needs to convert timestamps on the fly. But if some files are moved when summer or "daylight saving" local time is in effect, and other files are moved when winter or "standard" local time is in effect, there can be some ambiguities in the conversions. As a result, especially shortly after one of the days on which local zone time changes, users may observe that some files have timestamps that are incorrect by one hour. Due to the differences in implementation of Daylight Saving between the Northern and Southern hemispheres, this can result in a potential timestamp error of up to 4 hours in any given 12 months.[4]

Features

NTFS v3.0, the third version of NTFS to be introduced to the Windows world, includes several new features over its predecessors: disk usage quotas, sparse file support, reparse points, distributed link tracking and file-level encryption, also known as the Encrypting File System (EFS).

Alternate data streams (ADS)
: Alternate data streams allows files to be associated with more than one data stream. For example, a file such as text.txt can have a ADS with the name of text.txt:secret.txt (of form *filename:ads*) that can only be accessed by knowing the ADS name or by specialized directory browsing programs. Alternate streams are not detectable in the original file's size but are lost when the original file (i.e. text.txt) is deleted, or when the file is copied or moved to a partition that doesn't support ADS (e.g. a FAT partition, a floppy disk, or a network share). While ADS is a useful feature, it can also easily eat up hard disk space if unknown either through being forgotten or not being detected.

Quotas
: Disk quotas were introduced in NTFS v3. They allow the administrator of a computer that runs

a version of Windows that supports NTFS to set a threshold of disk space that users may utilise. It also allows administrators to keep a track of how much disk space each user is using. An administrator may specify a certain level of disk space that a user may use before they receive a warning, and then deny access to the user once they hit their upper limit of space. Disk quotas do not take into account NTFS's transparent file-compression, should this be enabled. Applications that query the amount of free space will also see the amount of free space left to the user who has a quota applied to them.

Sparse files
: Sparse files are files which are mostly filled with zeros. This is called a sparse data set, and most things that generate such data sets are scientific applications, and they can generate very large sparse data sets. Because of this, Microsoft has implemented support for sparse files by only allocating disk space for regions that do not contain blocks of zero data. An application that reads a sparse file reads it in the normal manner with the file system calculating what data should be returned based upon the file offset. As with compressed files, the actual size of sparse files are not taken into account when determining quota limits. [5]

Reparse points
: This feature was introduced in NTFS v3. These are used by associating a reparse tag in the user space attribute of a file or directory. When the object manager (see Windows NT line executive) parses a file system name lookup and encounters a reparse attribute, it knows to *reparse* the name lookup, passing the user controlled reparse data to every file system filter driver that is loaded into Windows 2000. Each filter driver examines the reparse data to see if it is associated with that reparse point, and if that filter driver determines a match then it intercepts the file system call and executes its special functionality. Reparse points are used to implement Volume Mount Points, Directory Junctions, Hierarchical Storage Management, Native Structured Storage and Single Instance Storage:

Volume mount points
: Similar to Unix mount points, where the root of another file system is attached to a directory. In NTFS, this allows additional file systems to be mounted without requiring a separate drive letter (like C: or D:) for each.

Directory Junctions
: Similar to Volume Mount Points, however directory junctions reference other directories in the file system instead of other volumes. For instance, the directory `C:\exampledir` with a directory junction attribute that contains a link to `D:\linkeddir` will automatically refer to the directory `D:\linkeddir` when it is accessed by a user-mode application. They are the equivalent of a Unix symbolic link, though in Unix a symbolic link can be applied on files as well as on directories [6].

Hard links
: Hard links are similar to directory junctions, but used for files instead of directories. Hard links can only be applied to files on the same volume since an additional filename record is added to the file's MFT record. Short (8.3) filenames are also implemented as additional filename records that don't have separate directory entries.

Hierarchical Storage Management (HSM)
: Hierarchical Storage Management is a means of transferring files that are not used for some period of time to less expensive storage media. When the file is next accessed the reparse point on that file determines that it is needed and retrieves it from storage.

Native Structured Storage (NSS)
: NSS was an ActiveX document storage technology that has since been discontinued by Microsoft. It allowed ActiveX documents to be stored in the same multi-stream format that ActiveX uses internally. An NSS file system filter was loaded and used to process the multiple streams transparently to the application, and when the file was transferred to a non-NTFS formatted disk volume it would also transfer the multiple streams into a single stream [7].

Volume Shadow Copy
: The Volume Shadow Copy (VSC) service keeps historical versions of files and folders on NTFS volumes by copying old, newly-overwritten data to shadow copy (*copy-on-write*). The old file data is overlaid on the new when the user requests a revert to an earlier version. This also allows data backup programs to archive files currently in use by the file system. On heavily loaded systems, Microsoft recommends setting up a shadow copy volume on separate disk to reduce the I/O load on the main volume.

File compression
: NTFS can compress files using a variant of the LZ77 algorithm (also used in the popular ZIP file format). [8] Although read-write access to compressed files is transparent, Microsoft recommends avoiding compression on server systems and/or network shares holding roaming profiles because it puts a considerable load on the processor. [9]
Single-user systems with limited hard disk space will probably use NTFS compression successfully. The slowest link in the 2.5 inch drive of a notebook computer is not the CPU, but

the speed of the drive, so NTFS compression allows the limited storage space to be better used.

Single Instance Storage (SIS)
: When there are several directories that have different, but similar, files, some of these files may have identical content. Single instance storage allows identical files to be merged to one file and create references to that merged file. SIS consists of a file system filter that manages copies, modification and merges to files; and a user space service (or *groveler*) that searches for files that are identical and need merging. SIS was mainly designed for remote installation servers as these may have multiple installation images that contain many identical files; SIS allows these to be consolidated but, unlike for example hard links, each file remains distinct; changes to one copy of a file will leave others unaltered [10].

Encrypting File System (EFS)
: EFS provides strong and user-transparent encryption of any file or folder on an NTFS volume. EFS works in conjunction with the EFS service, Microsoft's CryptoAPI and the EFS File System Run-Time Library (FSRTL). As of February 2004, its encryption has not been compromised. EFS works by encrypting a file with a bulk symmetric key (also known as the File Encryption Key, or FEK), which is used because it takes a relatively smaller amount of time to encrypt and decrypt large amounts of data than if an asymmetric key cipher is used. The symmetric key that is used to encrypt the file is then encrypted with a public key that is associated with the user who encrypted the file, and this encrypted data is stored in an alternate data stream of the encrypted file. To decrypt the file, the file system uses the private key of the user to decrypt the symmetric key that is stored in the file header. It then uses the symmetric key to decrypt the file. Because this is done at the file system level, it is transparent to the user. [11] Also, in case of a user losing access to their key, support for recovery agents that can unencrypt files has been built in to the EFS system.

Symbolic links
: Symbolic links were included for POSIX compatibility and were not accessible to Win32 applications until Windows 2000.

Limitations

The following are a few limitations of the NTFS file system.

Reserved File Names
: Though the file system supports paths up to ca. 32,000 Unicode characters with each path component (directory or filename) up to 255 characters long, certain names are unusable, since NTFS stores its metadata in regular (albeit hidden and for the most part inaccessible) files; accordingly, user files cannot use these names. These files are all in the root directory of a volume (and are reserved only for that directory). The names are: $MFT, $MFTMirr, $LogFile, $Volume, $AttrDef, . (dot), $Bitmap, $Boot, $BadClus, $Secure, $Upcase, and $Extend [12]; . and $Extend are both directories, the others are files.

Maximum Volume Size
: In theory, the maximum NTFS volume size is 2^{64}-1 clusters. However, the maximum NTFS volume size as implemented in Windows XP Professional is 2^{32}-1 clusters. For example, using 64 KiB clusters, the maximum NTFS volume size is 256 TiB minus 64 KiB. Using the default cluster size of 4 KiB, the maximum NTFS volume size is 16 TiB minus 4 KiB. Because partition tables on master boot record (MBR) disks only support partition sizes up to 2 TiB, you must use dynamic volumes to create NTFS volumes over 2 TiB.

Maximum File Size
: Theoretical: 16 EiB minus 1 KiB (2^{64} bytes minus 1 KiB). Implementation: 16 TiB minus 64 KiB (2^{44} bytes minus 64 KiB)

Alternate Data Streams
: Care must be exercised when copying or moving files from NTFS to other filesystem types. Windows system calls and programs can have varying behavior with regard to alternate data streams and might silently strip those which could not be stored on the destination filesystem. A safe way of copying or moving files is to use the BackupRead and BackupWrite system calls, which allow to enumerate streams, to verify whether each stream could be written to the destination volume and to knowingly skip offending streams.

Authors

Tom Miller, Gary Kimura, Brian Andrew, David Goebel et. al.

Notes

1. ^ "ntfsmount wiki page on linux-ntfs.org (http://wiki.linux-ntfs.org/doku.php?id=ntfsmount) "
2. ^ "Paragon Software Group (http://www.paragon.ag/) "
3. ^ "ntfs-3g announcement in linux-ntfs-dev list (http://sourceforge.net/mailarchive/forum.php?thread_id=23836054&forum_id=2697) "
4. ^ "Beating the Daylight Savings Time bug and getting correct file modification times (http://www.codeproject.com/datetime/dstbugs.asp) " *The Code Project*
5. ^ "Sparse Files (http://msdn.microsoft.com/library/default.asp?url=/library/en-us/fileio/fs/sparse_files.asp) ", *MSDN Platform SDK: File Systems*. Retrieved May 22, 2005.
6. ^ Mark Russinovich, "Inside Win2K NTFS, Part 1 (http://msdn.microsoft.com/library/default.asp?url=/library/en-us/dnw2kmag00/html/NTFSPart "
7. ^ John Saville, "What is Native Structured Storage? (http://www.windowsitpro.com/Article/ArticleID/13785/13785.html) "
8. ^ "File Compression and Decompression (http://msdn.microsoft.com/library/default.asp?url=/library/en-us/fileio/fs/file_compression_and ". *MSDN Platform SDK: File Systems.* Retrieved Aug 18, 2005.
9. ^ "Best practices for NTFS compression in Windows (http://support.microsoft.com/default.aspx?scid=kb;en-us;Q251186) ." *Microsoft Knowledge Base.* Retrieved Aug 18, 2005.
10. ^ "Single Instance Storage in Windows 2000 (http://research.microsoft.com/sn/Farsite/WSS2000.pdf) ". *Microsoft Research* & *Balder Technology Group, Inc.*
11. ^ "How EFS Works (http://www.microsoft.com/resources/documentation/Windows/2000/server/reskit/en-us/Defau " *Microsoft Windows 2000 Resource Kit*
12. ^ "How NTFS Works (http://www.microsoft.com/technet/prodtechnol/windowsserver2003/library/TechRef/8cc5891c " *Windows Server 2003 Technical Reference*

References

- Bolosky, William J.; Corbin, Scott; Goebel, David; & Douceur, John R. (date). "*Single Instance Storage in Windows 2000 (http://research.microsoft.com/sn/Farsite/WSS2000.pdf)* " (PDF). Microsoft Research & Balder Technology Group, Inc..
- Custer, Helen (1994). *Inside the Windows NT File System*. Microsoft Press. ISBN 1-55615-660-X.
- Nagar, Rajeev (1997). *Windows NT File System Internals: A Developer's Guide (1st ed)*. O'Reilly. ISBN 1-56592-249-2.

See also

- Files-11 — ODS-2 is structurally very similar to NTFS (compare `INDEXF.SYS` and `$Mft`, and `BITMAP.SYS` and `$Bitmap`, for examples)
- Comparison of file systems
- HPFS
- Captive NTFS

External links

- Linux-NTFS (http://linux-ntfs.org/) – an open source project to add NTFS support to the Linux kernel (write support is limited, but can be used for simple tasks), and write POSIX-compatible utilities for accessing and manipulating NTFS (ntfsprogs; includes ntfsls, ntfsresize, ntfsclone, etc)
- Captive NTFS (http://www.jankratochvil.net/project/captive/) – a shim which used the Windows NTFS driver to access NTFS filesystems under Linux

- NTFS.com (http://www.ntfs.com/) – documentation and resources for NTFS
- Microsoft NTFS Technical Reference (http://www.microsoft.com/technet/prodtechnol/windowsserver2003/library/TechRef/81cc8a8a
- NTFS4DOS (http://www.datapol.de/dpe/freeware/) – NTFS compatible DOS

Retrieved from "http://en.wikipedia.org/wiki/NTFS"

Categories: Articles with unsourced statements | Disk file systems | Windows disk file systems | Windows components | Compression file systems

---