EFFICIENT NTFS PARTITIONS

by Lance Jensen, Executive Software Technical Support Director

How you set up and use an NTFS partition can have a great deal of effect on performance. All partitions are not created equal! In fact, two brand-new but differently set up partitions can yield drastically different performances. Response time can degrade over time, even if you keep the partition defragmented. In this article we'll discuss the main factors involved in keeping NTFS partitions efficient.

The Partition Itself

Partitions should be created NTFS, not converted from FAT. On a newly formatted NTFS partition, the Master File Table (MFT) is created at the beginning of the disk, and about 12 1/2% of the disk is reserved for the MFT. But when you convert a FAT partition to NTFS, the MFT is placed wherever there is free space. It almost always ends up badly fragmented.

Large partitions should be avoided. They take a lot longer to back up and to restore, and access to the disk becomes slower (it takes longer to find, read and write files). Of course, there are valid reasons for very large partitions - if you have a 5 GB database or you work with large video files, you'll need big partitions. Just don't make them bigger than you need. It's also a good idea to have specialized partitions: System, Applications, Data, Temp, etc. This will increase safety and performance.

Directories

It's nice to have deep, multi-branched directory trees; I like the logical organization, keeping separate types of files neatly sorted. The bad news: Deep trees can really slow things down. The good news: It's easy to tidy up deep trees so they don't slow things down. Here are the details: Under NTFS, each directory is a file just like any other, but with a special internal structure called a "B+ Tree". It's fairly complicated, but for our purposes it's enough to say that it is a very good structure for a directory tree, but can be weak on handling changes. In other words, the more changes you make, the more complicated it gets internally, so the longer it takes to locate your file. Since files are listed in the directory file alphabetically by file name, adding new files (or directories) can require changes in the middle of the tree structure. Many such changes can make the structure quite complex, and more complexity means less speed.

Files are located by searching through the directories. If you are looking for a file in a tree that is ten levels deep, you have to locate ten directories before you get to the one that points to the fileitself. That takes a lot longer than locating a file that is only three levels deep. Plus, if the directories have been changed a lot so that their internal structure has become complex, finding files can become very slow.

Directories tend to grow, but rarely shrink. Sometimes when you add a new file or directory, it can be fitted into the space left by a deleted file, but often it uses a new space. The directory grows and can fragment, slowing down access even more.

Diskeeper can defragment directories, which provides significant performance gain itself, however, this will not handle the internal directory complexity. To clean that up and restore the directory to its initial perfect state, just copy the directory (with the copy under the same parent directory as the original, of course), giving it a new name, then delete the original, then rename the copy to the original name. This should be done periodically (once or twice a year?) if you frequently create and delete files, or whenever you delete a large number of files from a single directory. Since this changes the location of the directory file, it's a good idea to make a list of all of the directories that you want to clean up, and do them all at once. Then use Diskeeper to do a boot-time consolidation afterwards. This will move the directories together and defragment them.

One additional thing: Long file names can cause directories and the MFT to fragment. The way the file names are stored, each character requires two bytes. For computer efficiency, the DOS 8-dot-3 format is best. On the other hand, for human efficiency, 20 to 30 character names are much better. Of course, there are exceptions, such as files on a CD-ROM or an archive partition where they won't be re-written, but in general, don't go over thirty characters.

Cluster Size

New data regarding the MFT and its internal functions leads me to recommend 4096 bytes as the best cluster size, especially if you will have a very large number of files or will be using compression. Never use less than 1024 bytes, as this will allow MFT records to fragment, and never exceed 4096 bytes, as compression and Diskeeper will not work.