

# WINDOWS NT & LARGE PARTITIONS

By Mark E. Donaldson

Windows NT's NTFS disk format allows huge partitions. With a 512 byte cluster size, you can format a partition of 2 terabytes (TB). That's two trillion bytes, enough to hold about 4.5 million books. The maximum 64KB cluster size allows 256TB, or well over half a billion books. But even though Windows NT can address all of this huge space, the boot process starts with your BIOS (Basic Input/Output System), not Windows NT.

The BIOS locates the beginning of a partition by using three numbers: The Starting Side (or Head), the Starting Cylinder, and the Starting Sector. The end of a partition is identified by three similar numbers. Now, the Side value is 8 bits, and can range from 0 to 255 (256 numbers); the Cylinder is 10 bits, and can range from 0 to 1023 (1024 numbers); the Sector is 6 bits, and can range from 1 to 63 (63 numbers). (Note that zero is not a valid sector number.) This means the maximum address on the disk is Side 255, Cylinder 1023, Sector 63. The number of sectors is 256 X 1023 X 63, or almost 16.5 million sectors. Standard sectors are 512 bytes, so we have a size of 7.87GB.

That's the point: The BIOS cannot access anything beyond the first 7.87GB of the hard disk. If any critical boot data, such as the files NTLDR, NTDETECT or BOOT.INI, get moved to a point more than 7.87GB from the start of the hard disk, your computer will not boot. Anything that moves one of the critical files may cause the problem; you may copy the file from another partition, or you may edit the file. But you will not be aware of any problem till the next time you boot.

You are vulnerable to this if the partition on which Windows NT is installed is larger than 7.87GB, or if this partition is not the first partition on the disk and the total of this partition and all of the partitions before it exceeds 7.87GB.

## Fixing the Problem

First thing is to boot up the computer using a boot floppy. This is simply a floppy formatted on a Windows NT machine (that is vital), and containing copies of NTLDR, NTDETECT.COM and BOOT.INI. The BIOS accesses the floppy and finds the data it needs, then the floppy starts Windows NT, and everything runs fine. Of course, the next time you boot, the problem is back, but this gets the machine up and running so you can get your regular work done. Now you can schedule a time to fix it when you won't disrupt production.

The problem is simply that some critical file lies beyond the part of the disk which the BIOS can access. The simplest handling is to use the BIOS to rewrite the file. Start with your Windows NT Setup Floppies and CD, and proceed as if you were installing a second Windows NT installation. When asked which folder to use, specify a new folder, not the one that the existing Windows NT is in. When asked if you want to check the hard disks for errors, select "yes". Once this check is done, you can abort the new installation and Windows NT will boot up successfully.

Another handling is to move the critical files back closer to the beginning of the disk. This can be difficult because you really don't know where the files will be placed when you copy or move them. However, the odds are on your side if the partition lies mostly within the 7.87 GB limit. Trial and error will do it.

## Preventing the Problem

The simplest prevention is to always have the boot partition as the first partition on the hard disk, and ensure that it is less than 7.87 GB. If this is not feasible, take steps to ensure the critical files will not be moved. If the partition is in NTFS format, set the access on those files so that only an administrator can access them, and if you are using a defragmenter, add the critical files to the defragmenter's exclusion list.

# WINDOWS NT & LARGE PARTITIONS

By Mark E. Donaldson

QUESTION: Can you describe how you get around the 2GB FAT-16 volume size limitation upon initial installation?

ANSWER: There is no way around this. In fact, trying to do something about it is dangerous and can render your system unusable or unbootable. (People try to get around this by restoring GHOST images, and using other unusual solutions.)

This is precisely why I am writing this set of articles. If one knows WHY he is running into a certain problem and why it's a limitation in the first place, then the need to bypass it fades rather rapidly.

QUESTION: I am interested in hearing more about formatting NT partitions with 4096-byte clusters. Should I do this on workstations also? How big a drive does it have to be?

ANSWER: 4096-byte clusters are the largest size supported by the safe, built-in NT defragmentation APIs. On my Windows NT4 system volume, the average file size is 64K. That means I'm using, on average, 16 clusters per file. Now, if I was using 512-byte clusters, I'd be using, on average, 128 clusters per file.

I think it's pretty easy to see that I'd have an 8x probability of fragmenting my files when I use a 512-byte cluster size.

Also, it's important to keep in mind that the MFT records are 1024-bytes big. On a 512-byte cluster NTFS volume, the MFT records can split across 2 (sometimes widely separated) clusters. So it takes 2 discrete I/Os to read in a single MFT record! That's NOT good if having the best possible performance is a priority. With a 4096-byte cluster size, it's possible to fit 4 MFT records, with no splits, into a single cluster. This is much better for performance.

You can't take advantage of defragmentation (or compressed files, for that matter) on an NTFS volume with greater than 4096-byte clusters.

So, on any NTFS drive, of any size, Workstation or Server, I always recommend 4096-byte clusters.

QUESTION: Why make the secondary NT installation on a FAT-16 partition? I make (1) a secondary installation in an NTFS partition and (2) a small FAT-16 partition so I can boot DOS from a floppy to run DOS hardware tests, etc. Frequently, my secondary NT installation is a test system, so I don't maintain it as a "virginal" installation.

ANSWER: Below are the reasons I make a fairly large FAT-16 Windows NT installation, followed by another NTFS Windows NT installation on a separate volume.

(1) Having 2 complete bootable instances of NT gives me a rapid way to repair the "other" instance of NT. For example, if my NTFS boot goes bad, I can boot into the FAT volume and run a CHKDSK on the NTFS volume. I can also repair the registry on the NTFS volume from the FAT volume. I can also grab test case data off the failing NTFS volume and get it to Microsoft for Tech Support.

In fact, I can "backup" my NTFS volume to another disk just using XCOPY if I have a FAT bootable volume. However, I usually make a binary image of the NTFS volume onto another volume. It helps speed up the restore. Of course I use the NTFS volume boot to back up the FAT volume boot, too.

# WINDOWS NT & LARGE PARTITIONS

By Mark E. Donaldson

(2) The initial installation of Windows NT 4 creates a FAT volume, which can then be converted to an NTFS volume automatically by the installation process. If that conversion occurs, however, the resultant NTFS volume gets created with a 512-byte cluster size. As discussed above, this is not the value that will give you the best performance when building an NTFS volume. And once it's been converted, it can't be changed.

So, I install to a 2GB FAT-16 volume, then use that 2GB FAT-16 volume's Windows NT installation to create a second volume on the first disk that's an NTFS volume with 4096-byte clusters. This gives me a redundant boot-image system with a primary operating NT system disk that has the best possible cluster size.

(3) With a separate boot of NT I can run Diskkeeper on the "other" boot of NT, defragmenting the files on the "other" boot disk that are normally held open by the kernel, preventing their defragmentation.

And, please remember, I'm trying to describe how \*I\* set up a system and why \*I\* set up a system that way. Your mileage may vary. You may not want a 4096-byte NTFS volume as your system volume, but hey, that's your decision! I just hope I've explained why I lay out things this way.

I'm not trying to enforce some kind of change on your operations. But I am trying to illuminate decisions that may be costly performance-wise.