NTFS And 4K Disks

ntdebug 28 Jun 2011 8:36 AM | 3

Since the 1960's, hard disks have always used a block size of 512 bytes for the default read/write block size. Recently drive manufacturers have been moving toward a larger block size to improve performance and reliability. Currently there are two types of disks available with a 4KB sector size: 512 byte emulated, and 4KB block sized disks.

Disks with 4KB block size and 512 bytes per sector emulation
For performance reasons, drive manufacturers have already produced disks with 4KB native block size, which use firmware to emulate 512 bytes per sector. Because of the emulated 512 byte sector size, the file system and most disk utilities will be blissfully unaware that they are running on a 4KB disk. As a result, the on-disk structures will be completely unaffected by the underlying 4KB block size. This allows for improved performance without altering the bytes per sector presented to the file system. These disks are referred to as 512e (pronounced "five-twelve-eee") disks.

Disks with 4KB block size without emulation
When the logical bytes per sector value is extended to 4KB without emulation, the actual file system will have to adjust to this new environment. Actually, NTFS is already capable of functioning in this environment provided that no attached FS filter drivers make false assumptions about sector size. Below are the highlights of what you should expect to see on a disk with a 4KB logical sector size.

1. It will not be possible to format with a cluster size that is smaller than the 4KB native block size. This is because cluster size is defined as a multiple of sector size. This multiple will always be expressed as 2n .
2. File records will assume the size of the logical block size of 4KB, rather than the previous size of 1KB. This actually improves scalability to some degree, but the down-side is that each NTFS file record will require 4KB or more in the MFT.
3. Sparse and compressed files will continue to have 16 clusters per compression unit.
4. Since file records are 4 times their normal size, it will be possible to encode more mapping pairs per file record. As a result, larger files can be compressed with NTFS compression without running into file system limitations.
5. Since the smallest allowable cluster size is 4KB, NTFS compression will only work on volumes with a 4KB cluster size.
6. Bytes per index record will be unaffected by the 4K block size since all index records are 4KB in size. The on-disk folder directory structures will be completely unaffected by the new block size, but a performance increase may be seen while accessing folder structure metadata.
7. The BIOS Parameter Block (BPB) will continue to have the same format as before, but the only positive value for clusters per File Record Segment (FRS) will be 1. In the case where clusters per FRS is 1, the FRS byte size is computed by the following equation:

$$BytesPerFrs = BytesPerSector * SectorsPerCluster * ClustersPerFrs$$

$$BytesPerFrs = 4096 * 1 * 1 = 4096$$

```
NTFS BIOS Parameter Block Information

  BytesPerSector      :        4096
  Sectors Per Cluster :           1
  ReservedSectors     :           0
  Fats                :           0
  RootEntries         :           0
  Small Sectors       :           0 ( 0 MB )
  Media Type          :         248 ( 0xf8 )
  SectorsPerFat       :           0
  SectorsPerTrack     :          63
  Heads               :         255
  Hidden Sectors      :          64
  Large Sectors       :           0 ( 0 MB )

  ClustersPerFRS      :           1
  Clust/IndxAllocBuf  :           1
  NumberSectors       :                 50431 ( 196.996 MB )
  MftStartLcn         :                 16810
  Mft2StartLcn        :                     2
  SerialNumber        :  8406742282501311868
  Checksum            :                     0 (0x0)
```

If the cluster size is larger than the FRS size, then ClustersPerFrs will be a negative number as shown in the example below (0xf4 is -12 decimal). In this case, the record size is computed with the equation:

$$BytesPerFrs = 2^{-ClustersPerFrs}$$

$$BytesPerFrs = 2^{-(-12)} = 4096$$

In short, NTFS will always force a 4096 byte cluster size on disk with a 4KB sector size regardless of the cluster size.

```
NTFS BIOS Parameter Block Information

  BytesPerSector      :        4096
  Sectors Per Cluster :           4
  ReservedSectors     :           0
  Fats                :           0
  RootEntries         :           0
  Small Sectors       :           0 ( 0 MB )
  Media Type          :         248 ( 0xf8 )
  SectorsPerFat       :           0
  SectorsPerTrack     :          63
  Heads               :         255
  Hidden Sectors      :          64
  Large Sectors       :           0 ( 0 MB )

  ClustersPerFRS      :          f4
  Clust/IndxAllocBuf  :          f4
  NumberSectors       :                 50431 ( 196.996 MB )
  MftStartLcn         :                  4202
  Mft2StartLcn        :                     1
  SerialNumber        :  7270585088516976380
  Checksum            :                     0 (0x0)
```

8. Aside from the 4KB file record size, there are a few other things to know about 4KB drives. The code for implementing update sequence arrays (USA's) has always worked on a 512 byte assumed sector size and it will continue to do so. Since file records are 4 times their normal size, the update sequence arrays for file records now contain 9 entries instead of 3. One array entry is required for the sequence number (blue) and eight array entries for the trailing bytes (red). The original purpose of USA is to allow NTFS to detect torn writes. Since the file record size is now equal to the block size, the hardware is capable of writing the entire file record at once, rather than in two parts.

```
    _MULTI_SECTOR_HEADER MultiSectorHeader {
            ULONG      Signature              : 0x454c4946 "FILE"
            USHORT     SequenceArrayOffset    : 0x0030
            USHORT     SequenceArraySize      : 0x0009
    }


0x0000   46 49 4c 45 30 00 09 00-dd 24 10 00 00 00 00 00   FILE0...Ý$.....
0x0010   01 00 01 00 48 00 01 00-b0 01 00 00 00 10 00 00   ....H...°......
0x0020   00 00 00 00 00 00 00 00-06 00 00 00 00 00 00 00   ................
0x0030   02 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00   ................
0x0040   00 00 00 00 00 00 00 00-10 00 00 00 60 00 00 00   ...........`...
0x0050   00 00 18 00 00 00 00 00-48 00 00 00 18 00 00 00   .......H......
0x0060   f8 f1 5b 89 36 d2 cb 01-f8 f1 5b 89 36 d2 cb 01   øñ[‰6ÒË.øñ[‰6ÒË.
0x0070   f8 f1 5b 89 36 d2 cb 01-f8 f1 5b 89 36 d2 cb 01   øñ[‰6ÒË.øñ[‰6ÒË.
0x0080   06 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00   ................
0x0090   00 00 00 00 00 01 00 00-00 00 00 00 00 00 00 00   ................
0x00a0   00 00 00 00 00 00 00 00-30 00 00 00 68 00 00 00   ........0...h...
0x00b0   00 00 18 00 00 00 03 00-4a 00 00 00 18 00 01 00   .......J......
0x00c0   05 00 00 00 00 00 05 00-f8 f1 5b 89 36 d2 cb 01   ........øñ[‰6ÒË.
0x00d0   f8 f1 5b 89 36 d2 cb 01-f8 f1 5b 89 36 d2 cb 01   øñ[‰6ÒË.øñ[‰6ÒË.
0x00e0   f8 f1 5b 89 36 d2 cb 01-00 00 01 00 00 00 00 00   øñ[‰6ÒË.........
0x00f0   00 00 01 00 00 00 00 00-06 00 00 00 00 00 00 00   ................
0x0100   04 03 24 00 4d 00 46 00-54 00 00 00 00 00 00 00   ..$.M.F.T.......
0x0110   80 00 00 00 48 00 00 00-01 00 40 00 00 00 01 00   €...H.....@.....
0x0120   00 00 00 00 00 00 00 00-ff 00 00 00 00 00 00 00   ........ÿ.......
0x0130   40 00 00 00 00 00 00 00-00 00 10 00 00 00 00 00   @.............
0x0140   00 00 10 00 00 00 00 00-00 00 10 00 00 00 00 00   .............
0x0150   22 00 01 aa 41 00 ff ff-b0 00 00 00 50 00 00 00   "..ªA.ÿÿ°...P...
0x0160   01 00 40 00 00 00 05 00-00 00 00 00 00 00 00 00   ..@.............
0x0170   01 00 00 00 00 00 00 00-40 00 00 00 00 00 00 00   ........@.......
0x0180   00 20 00 00 00 00 00 00-08 10 00 00 00 00 00 00   . .............
0x0190   08 10 00 00 00 00 00 00-21 01 a9 41 21 01 fd fd   .......!.©A!.ýý
0x01a0   00 69 b4 05 80 fa ff ff-ff ff ff ff 00 00 00 00   .i´.€úÿÿÿÿÿÿ....
0x01b0   00 00 10 00 00 00 00 00-22 00 01 aa 41 00 ff ff   ......."..ªA.ÿÿ
0x01c0   b0 00 00 00 50 00 00 00-01 00 40 00 00 00 05 00   °...P.....@.....
0x01d0   00 00 00 00 00 00 00 00-01 00 00 00 00 00 00 00   ................
0x01e0   40 00 00 00 00 00 00 00-00 20 00 00 00 00 00 00   @........ ......
0x01f0   08 10 00 00 00 00 00 00-08 10 00 00 00 00 02 00   .............
.
.
.
0x03c0   00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00   ................
0x03d0   00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00   ................
0x03e0   00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00   ................
0x03f0   00 00 00 00 00 00 00 00-00 00 00 00 00 00 02 00   ................
.
.
.
0x05d0   00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00   ................
0x05e0   00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00   ................
0x05f0   00 00 00 00 00 00 00 00-00 00 00 00 00 00 02 00   ................
.
.
.
0x07d0   00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00   ................
```

```
0x07e0   00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00   ................
0x07f0   00 00 00 00 00 00 00 00-00 00 00 00 00 00 02 00   ................
.
.
.
0x09d0   00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00   ................
0x09e0   00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00   ................
0x09f0   00 00 00 00 00 00 00 00-00 00 00 00 00 00 02 00   ................
.
.
.
0x0bd0   00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00   ................
0x0be0   00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00   ................
0x0bf0   00 00 00 00 00 00 00 00-00 00 00 00 00 00 02 00   ................
.
.
.
0x0dd0   00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00   ................
0x0de0   00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00   ................
0x0df0   00 00 00 00 00 00 00 00-00 00 00 00 00 00 02 00   ................
.
.
.
0x0fd0   00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00   ................
0x0fe0   00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00   ................
0x0ff0   00 00 00 00 00 00 00 00-00 00 00 00 00 00 02 00   ................
```

I don't actually own a 4KB disk, but I was able to give you this preview thanks to a nifty tool called VStorControl. Vstor is a tool which allows you to create virtualized SCSI disks with arbitrary block sizes and is available for download with the Windows 7 SDK.

That's all for now,
Dennis Middleton "The NTFS Doctor"

Comments

Alex
6 Jul 2011 7:02 AM

Thanks for an informative post. 1 question: How does one print out the "BIOS Parameter Block" as you have done ? Is that done in Windbg ?

-Alex

[The tool used to dump the BPB is Secinspect.exe. It is available from **http://download.microsoft.com**]

Ivan
10 Jul 2012 5:10 AM

"One byte is required for the sequence number (blue)..."

Please, correct this. Maybe you meant "One array entry"

Thanks for the very useful post.

[Good catch, we'll get this fixed. Thank you.]

Dylan
9 Aug 2013 8:43 AM

Hi Denis,

"NTFS will always force a 4096 byte cluster size on disk with a 4KB sector size regardless of the cluster size."

Layman question: Does this mean, there is strictly no difference and no use in formatting a 4k native Advanced Format HDD (with 4096 physcial+logical sector size) with 64kb clusters, because "NTFS will force a 4k cluster size" ?

I actually wanted to use 64kb clusters on an 3TB external 4k-native HDD where almost all files are videos of several hundred MBs, so wasted slack space is not an issue for me.

Thank you.

[Hi Dylan,

Regardless of physical sector size, large cluster sizes are
ideal for accessing and storing large files, so go ahead and format with a
large cluster size and you will see better performance and less fragmentation
overall. If you are wanting more details on why, I added some additional
technical details below that may help.

NTFS honors the formatted cluster size in all cases, but
internally it uses an MFT record size that is dependent on the physical sector
size. Multiple file records can either reside in a single cluster, or a
single MFT record may reside in multiple clusters. This all depends on
physical sector size vs. cluster size. For example, if you have a 16 KB
cluster size on a 4 KB bytes per sector native disk, there are a possible 4 MFT
records per cluster. If you have a 1 KB cluster size on a 4 KB bytes per

sector native disk, then there are 4 clusters per 1 MFT record (necessitating that we grow the MFT in multiples of 4 clusters for the sake of alignment).

Your video files reside in nonresident data streams. In the nonresident case, NTFS will store the contents of the stream in allocated cluster ranges that line up with your formatted cluster size. However, for resident data streams (files significantly smaller than physical sector size), the stream will be packed inside the owning MFT record because it’s small enough to fit (this prevents us from having to allocate a 64KB cluster to store a 1KB file).

I hope this helps, and thanks for your question!]