A Quick Look at $MFT Resident Data on Advanced Format Disks

Background

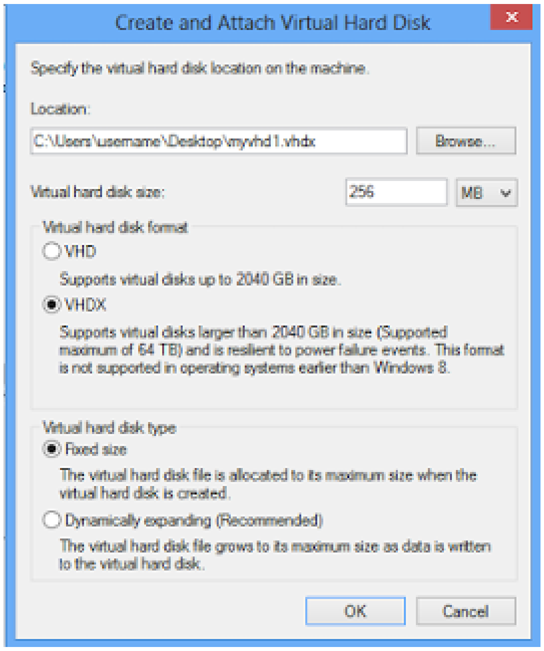
While attending the CTIN Digital Forensics conference Troy Larson, of Microsoft, gave an excellent presentation on Windows 8 changes with respect to the digital forensic artifacts. One topic he mentioned was the possibility that Windows 8/Windows Server 2012 NTFS volumes on 4K advanced format hard drives could contain larger amounts of $MFT resident data than a you would typically find on a standard NTFS volume.

This statement intrigued me since many Incident Responders leverage $MFT resident data during triage forensics scenarios. Typically, the $MFT file is one of several key artifacts IR teams collect as part of a triage process in order to timeline system activity or perform other types of analysis in rapid response scenarios As a result of this review, interesting files may be identified which an intruder may have left behind that are on occasion small enough to be stored in directly the $MFT. In a classic NTFS volume you will find that any file less than around 750 bytes in size will be stored within the $MFT. These files can be viewed in a hex editor, extracted via a script or other means. How will this change with advanced format hard drives?

Simulating an Advanced Format Drive

I used the following process to analyze changes to the $MFT on advanced format hard drives.

1. On a Windows 8 system I created a 256MB virtual disk in VHDX format.



Create and Attach Virtual Hard Disk

Specify the virtual hard disk location on the machine.

Location:

C:\Users\username\Desktop\myvhd1.vhdx

Virtual hard disk size: 256 MB

Virtual hard disk format

○ VHD

Supports virtual disks up to 2040 GB in size.

● VHDX

Supports virtual disks larger than 2040 GB in size (Supported maximum of 64 TB) and is reslient to power failure events. This format is not supported in operating systems earlier than Windows 8.

Virtual hard disk type

● Fixed size

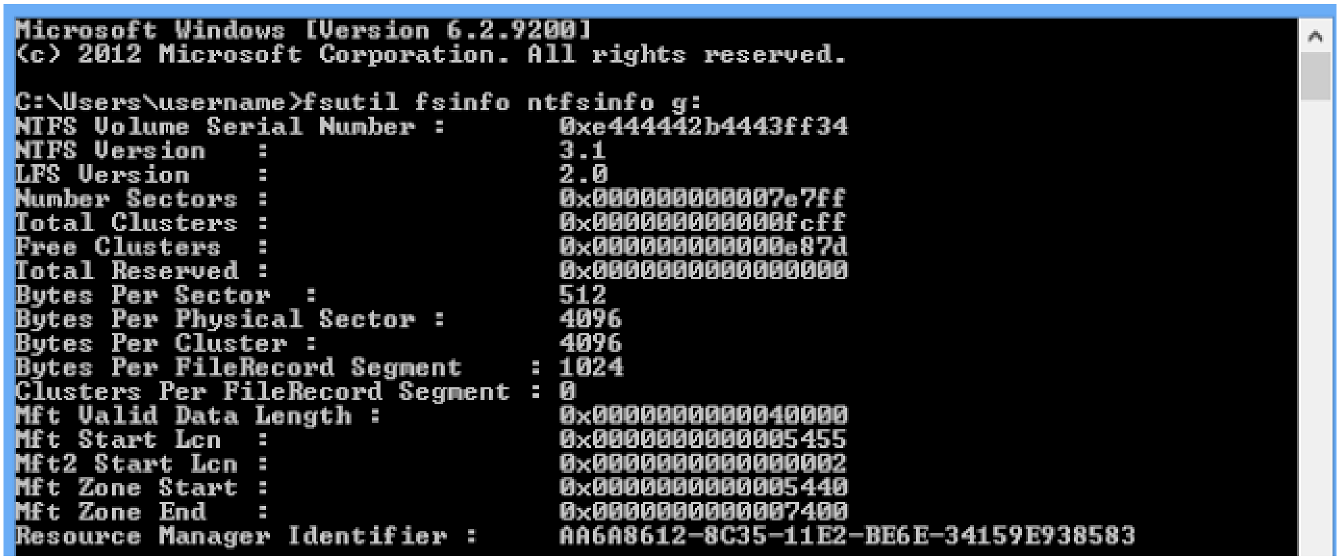
The virtual hard disk file is allocated to its maximum size when the virtual hard disk is created.

○ Dynamically expanding (Recommended)

The virtual hard disk file grows to its maximum size as data is written to the virtual hard disk.

[http://1.bp.blogspot.com/--1vu33IlmOc/UUEsPmfBPXI/AAAAAAAAACU/BhNwFtuWCQ/s1600/Screen+Shot+2013-03-13+at+8.47.13+PM.png]

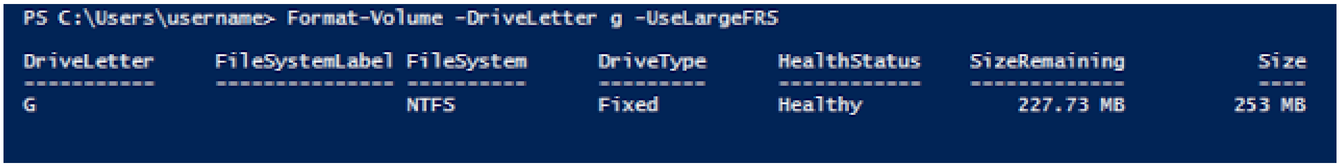
2. I initialized the disk and formatted with NTFS using the default options in the Disk Management MMC (Note: this drive was mounted as G: on my system). Running fsutil showed the following properties of this drive:



```
Microsoft Windows [Version 6.2.9200]
(c) 2012 Microsoft Corporation. All rights reserved.

C:\Users\username>fsutil fsinfo ntfsinfo g:
NTFS Volume Serial Number :       0xe444442b4443ff34
NTFS Version   :                  3.1
LFS Version    :                  2.0
Number Sectors :                  0x000000000007e7ff
Total Clusters :                  0x000000000000fcff
Free Clusters  :                  0x000000000000e87d
Total Reserved :                  0x0000000000000000
Bytes Per Sector  :               512
Bytes Per Physical Sector :       4096
Bytes Per Cluster :               4096
Bytes Per FileRecord Segment    : 1024
Clusters Per FileRecord Segment : 0
Mft Valid Data Length :           0x0000000000040000
Mft Start Lcn  :                  0x0000000000005455
Mft2 Start Lcn :                  0x0000000000000002
Mft Zone Start :                  0x0000000000005440
Mft Zone End   :                  0x0000000000007400
Resource Manager Identifier :     AA6A8612-8C35-11E2-BE6E-34159E938583
```

[http://1.bp.blogspot.com/-sZ_OwnPoQPg/UUEtNQbrcRI/AAAAAAAAACc/IaaPfpmkFbs/s1600/Screen+Shot+2013-03-13+at+8.51.03+PM.png]

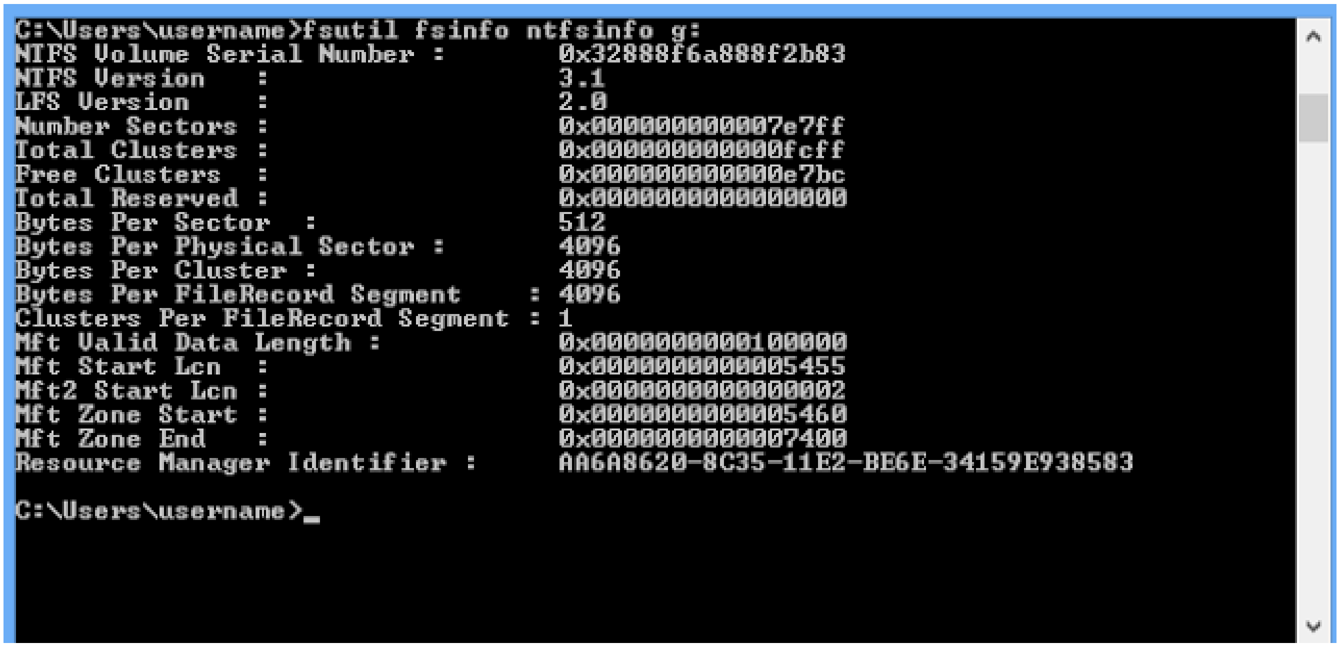
3. I used powershell to convert this disk to an advanced format disk using the "UseLargeFRS" option.



```
PS C:\Users\username> Format-Volume -DriveLetter g -UseLargeFRS

DriveLetter      FileSystemLabel FileSystem       DriveType        HealthStatus     SizeRemaining             Size
-----------      --------------- ----------       ---------        ------------     -------------             ----
G                                NTFS             Fixed            Healthy               227.73 MB          253 MB
```

[http://4.bp.blogspot.com/-QXyMhNyTg3Q/UUEtbjSc4GI/AAAAAAAAACk/tt7IRX91ayc/s1600/Screen+Shot+2013-03-13+at+8.52.25+PM.png]

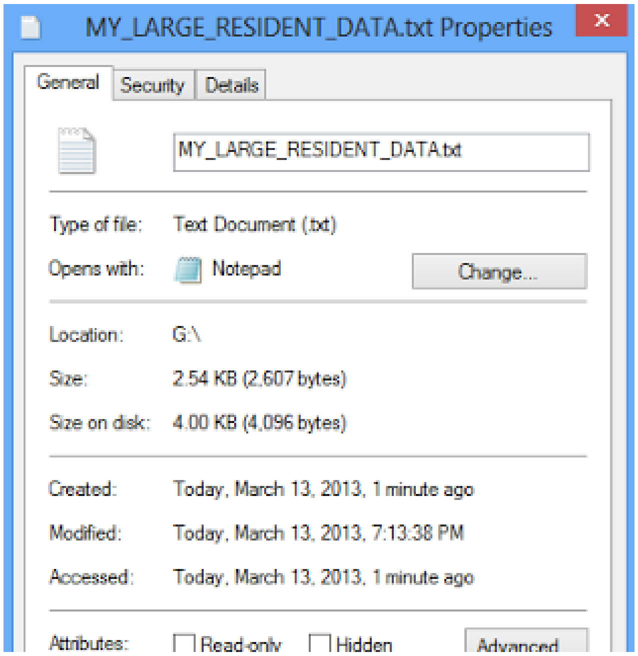
4. I re-ran fsutil. *Notice that the Bytes Per FileRecord Segment value has changed from **1024 bytes to 4096.***



```
C:\Users\username>fsutil fsinfo ntfsinfo g:
NTFS Volume Serial Number :       0x32888f6a888f2b83
NTFS Version   :                  3.1
LFS Version    :                  2.0
Number Sectors :                  0x000000000007e7ff
Total Clusters :                  0x000000000000fcff
Free Clusters  :                  0x000000000000e7bc
Total Reserved :                  0x0000000000000000
Bytes Per Sector  :               512
Bytes Per Physical Sector :       4096
Bytes Per Cluster :               4096
Bytes Per FileRecord Segment    : 4096
Clusters Per FileRecord Segment : 1
Mft Valid Data Length :           0x0000000000100000
Mft Start Lcn  :                  0x0000000000005455
Mft2 Start Lcn :                  0x0000000000000002
Mft Zone Start :                  0x0000000000005460
Mft Zone End   :                  0x0000000000007400
Resource Manager Identifier :     AA6A8620-8C35-11E2-BE6E-34159E938583

C:\Users\username>_
```

[http://2.bp.blogspot.com/-8ebj_ZeD1_A/UUEtxxf7n0I/AAAAAAAAACs/p6iQCKaE_Kk/s1600/Screen+Shot+2013-03-13+at+8.53.49+PM.png]

5. I created a text file called MY_LARGE_RESIDENT_DATA.txt around 2.54KB in size on the G volume containing the string "This file is about 3KB in size." repeated a bunch of times.



MY_LARGE_RESIDENT_DATA.txt

Type of file: Text Document (.txt)

Opens with: Notepad

Location: G:\

Size: 2.54 KB (2,607 bytes)

Size on disk: 4.00 KB (4,096 bytes)

Created: Today, March 13, 2013, 1 minute ago

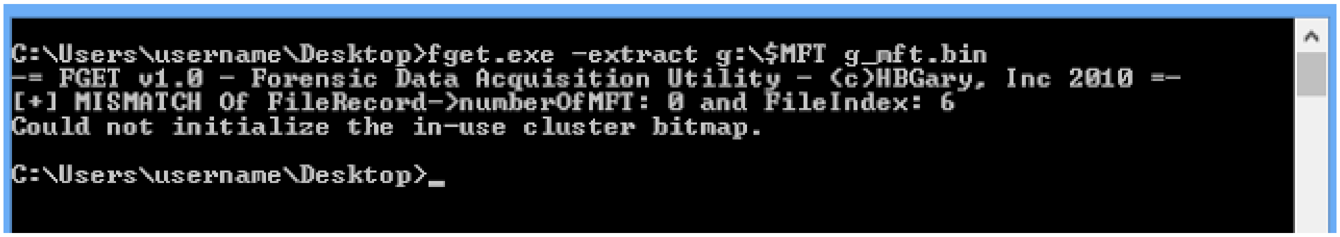
Modified: Today, March 13, 2013, 7:13:38 PM

Accessed: Today, March 13, 2013, 1 minute ago

Attributes: □ Read-only □ Hidden

[http://1.bp.blogspot.com/-W4HSAGiKGhQ/UUE_FUmCebI/AAAAAAAAADk/B-8fpPion88/s1600/my.png]

6. Attempted to extract the $MFT using fget.exe for analysis.



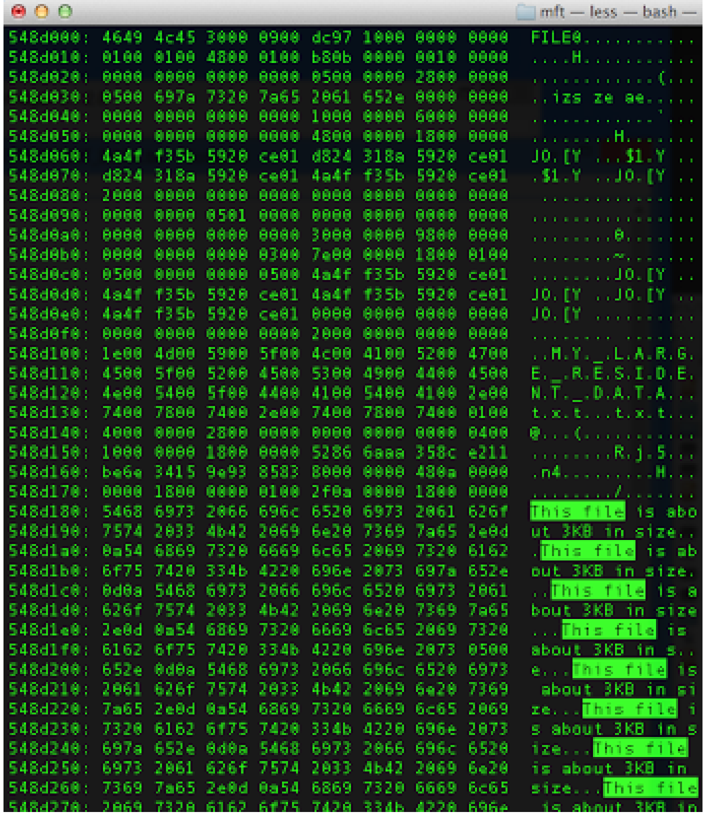
```
C:\Users\username\Desktop>fget.exe -extract g:\$MFT g_mft.bin
-= FGET v1.0 - Forensic Data Acquisition Utility - (c)HBGary, Inc 2010 =-
[+] MISMATCH Of FileRecord->numberOfMFT: 0 and FileIndex: 6
Could not initialize the in-use cluster bitmap.

C:\Users\username\Desktop>_
```

[http://3.bp.blogspot.com/-z1ZFOWeZdpM/UUEuV4gU_-I/AAAAAAAAAC0/kcKDWavf6Ag/s1600/Screen+Shot+2013-03-13+at+8.56.17+PM.png]

**Hmm, this doesn't work.   I wonder how many other tools will "break" due to this change?*

7. Since I could not extract the $MFT directly using fget.exe I decided to create a physical disk image of my G volume using FTK Imager.   I saved this file as g_drive.dd.001 and moved it out of the Windows 8 virtual machine to my Mac for further analysis.

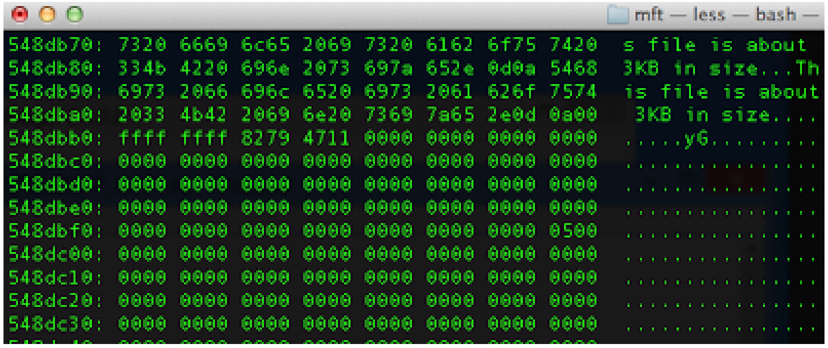
As many digital forensic practitioners would do, I started with a hex editor to review the disk image. Using xxd I searched for the string "This file" to locate the file contents within the forensic image.   Sure enough, we locate our file and can see the NTFS FILE0 magic bytes at offset 0x548d000 and the $MFT structure. *Note the $DATA attribute of the $MFT record starts at offset 0x548d168 and the resident flag offset 9 bytes from the start of the $Data attribute is 0x00 indicating that the data is resident to the $MFT.*



```
548d000: 4649 4c45 3000 0900 dc97 1000 0000 0000  FILE0...........
548d010: 0100 0100 4800 0100 b80b 0000 0010 0000  ....H...........
548d020: 0000 0000 0000 0000 0500 0000 2800 0000  ............(...
548d030: 0500 697a 7320 7a65 2061 652e 0000 0000  ..izs ze ae.....
548d040: 0000 0000 0000 0000 1000 0000 6000 0000  ............`...
548d050: 0000 0000 0000 0000 4800 0000 1800 0000  ........H.......
548d060: 4a4f f35b 5920 ce01 d824 318a 5920 ce01  JO.[Y ...$1.Y ..
548d070: d824 318a 5920 ce01 4a4f f35b 5920 ce01  .$1.Y ..JO.[Y ..
548d080: 2000 0000 0000 0000 0000 0000 0000 0000   ...............
548d090: 0000 0000 0501 0000 0000 0000 0000 0000  ................
548d0a0: 0000 0000 0000 0000 3000 0000 9800 0000  ........0.......
548d0b0: 0000 0000 0000 0300 7e00 0000 1800 0100  ........~.......
548d0c0: 0500 0000 0000 0500 4a4f f35b 5920 ce01  ........JO.[Y ..
548d0d0: 4a4f f35b 5920 ce01 4a4f f35b 5920 ce01  JO.[Y ..JO.[Y ..
548d0e0: 4a4f f35b 5920 ce01 0000 0000 0000 0000  JO.[Y ..........
548d0f0: 0000 0000 0000 0000 2000 0000 0000 0000  ........ .......
548d100: 1e00 4d00 5900 5f00 4c00 4100 5200 4700  ..M.Y._.L.A.R.G.
548d110: 4500 5f00 5200 4500 5300 4900 4400 4500  E._.R.E.S.I.D.E.
548d120: 4e00 5400 5f00 4400 4100 5400 4100 2e00  N.T._.D.A.T.A...
548d130: 7400 7800 7400 2e00 7400 7800 7400 0100  t.x.t...t.x.t...
548d140: 4000 0000 2800 0000 0000 0000 0000 0400  @...(...........
548d150: 1000 0000 1800 0000 5286 6aaa 358c e211  ........R.j.5...
548d160: be6e 3415 9e93 8583 8000 0000 480a 0000  .n4.........H...
548d170: 0000 1800 0000 0100 2f0a 0000 1800 0000  ......../.......
548d180: 5468 6973 2066 696c 6520 6973 2061 626f  This file is abo
548d190: 7574 2033 4b42 2069 6e20 7369 7a65 2e0d  ut 3KB in size..
548d1a0: 0a54 6869 7320 6669 6c65 2069 7320 6162  .This file is ab
548d1b0: 6f75 7420 334b 4220 696e 2073 697a 652e  out 3KB in size.
548d1c0: 0d0a 5468 6973 2066 696c 6520 6973 2061  ..This file is a
548d1d0: 626f 7574 2033 4b42 2069 6e20 7369 7a65  bout 3KB in size
548d1e0: 2e0d 0a54 6869 7320 6669 6c65 2069 7320  ...This file is 
548d1f0: 6162 6f75 7420 334b 4220 696e 2073 0500  about 3KB in s..
548d200: 652e 0d0a 5468 6973 2066 696c 6520 6973  e...This file is
548d210: 2061 626f 7574 2033 4b42 2069 6e20 7369   about 3KB in si
548d220: 7a65 2e0d 0a54 6869 7320 6669 6c65 2069  ze...This file i
548d230: 7320 6162 6f75 7420 334b 4220 696e 2073  s about 3KB in s
548d240: 697a 652e 0d0a 5468 6973 2066 696c 6520  ize...This file 
548d250: 6973 2061 626f 7574 2033 4b42 2069 6e20  is about 3KB in 
548d260: 7369 7a65 2e0d 0a54 6869 7320 6669 6c65  size...This file
548d270: 2069 7320 6162 6f75 7420 334b 4220 696e   is about 3KB in
```

[http://1.bp.blogspot.com/-2uSpH3ZGnX4/UUE0uWgbbHI/AAAAAAAAADE/SRZ6XbwHDQU/s1600/Screen+Shot+2013-03-13+at+9.22.16+PM.png]

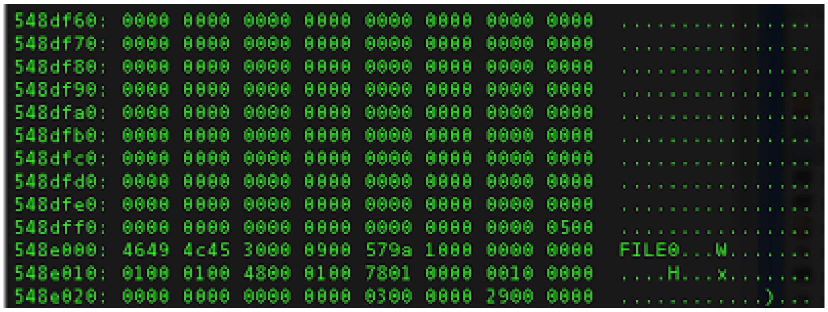
Scrolling down I found that the file contents end at offset 0x548dbad and that our $MFT record is padded with null bytes up until the next FILE0 marker indicating the start of the next $MFT record.



```
548db70: 7320 6669 6c65 2069 7320 6162 6f75 7420  s file is about
548db80: 334b 4220 696e 2073 697a 652e 0d0a 5468  3KB in size...Th
548db90: 6973 2066 696c 6520 6973 2061 626f 7574  is file is about
548dba0: 2033 4b42 2069 6e20 7369 7a65 2e0d 0a00   3KB in size....
548dbb0: ffff ffff 8279 4711 0000 0000 0000 0000  .....yG.........
548dbc0: 0000 0000 0000 0000 0000 0000 0000 0000  ................
548dbd0: 0000 0000 0000 0000 0000 0000 0000 0000  ................
548dbe0: 0000 0000 0000 0000 0000 0000 0000 0000  ................
548dbf0: 0000 0000 0000 0000 0000 0000 0000 0500  ................
548dc00: 0000 0000 0000 0000 0000 0000 0000 0000  ................
548dc10: 0000 0000 0000 0000 0000 0000 0000 0000  ................
548dc20: 0000 0000 0000 0000 0000 0000 0000 0000  ................
548dc30: 0000 0000 0000 0000 0000 0000 0000 0000  ................
```

[http://2.bp.blogspot.com/-XbO-2c-YXc4/UUE0w6bX12I/AAAAAAAAADM/p4ax6GKBRZQ/s1600/Screen+Shot+2013-03-13+at+9.22.54+PM.png]

Start of the next $MFT record at offset 0x548e000.



```
548df60: 0000 0000 0000 0000 0000 0000 0000 0000  ................
548df70: 0000 0000 0000 0000 0000 0000 0000 0000  ................
548df80: 0000 0000 0000 0000 0000 0000 0000 0000  ................
548df90: 0000 0000 0000 0000 0000 0000 0000 0000  ................
548dfa0: 0000 0000 0000 0000 0000 0000 0000 0000  ................
548dfb0: 0000 0000 0000 0000 0000 0000 0000 0000  ................
548dfc0: 0000 0000 0000 0000 0000 0000 0000 0000  ................
548dfd0: 0000 0000 0000 0000 0000 0000 0000 0000  ................
548dfe0: 0000 0000 0000 0000 0000 0000 0000 0000  ................
548dff0: 0000 0000 0000 0000 0000 0000 0000 0500  ................
548e000: 4649 4c45 3000 0900 579a 1000 0000 0000  FILE0...W.......
548e010: 0100 0100 4800 0100 7801 0000 0010 0000  ....H...x.......
548e020: 0000 0000 0000 0000 0300 0000 2900 0000  ............)...
```

[http://2.bp.blogspot.com/-VmOKaXieEOc/UUE0w7bNmOI/AAAAAAAAADQ/1xrjuiGYyfU/s1600/Screen+Shot+2013-03-13+at+9.23.06+PM.png]

By subtracting 0x548d000 from from 0x548e000 we can confirm the fsutil output indicting the $MFT record size as 4096 (0x1000) to be true.

What does this mean?

In the end it is unclear if this is something that will cause DRIR practitioners problems.   It depends how many real world scenarios responders will run across where this formatting scheme is in use.   As of now, I'm unsure how many new Windows 8 systems are being shipped with this type for formatting enabled by default or if Windows 8 will use this formatting schema when installed to a fresh 4K advanced format drive. Via Google I found several articles where questions are being asked about systems with the "UseLargeFRS " option enabled so evidently some people are working with this type of drive.

If we start to see $MFT records of 4K in size as standard practice we can expect that we may start seeing resident data upwards of 3.7KB in size (and $MFT files 4 times the size of a a "standard" $MFT).   The good news is you collect the $MFT for triage forensics you have a better chance that bad guy activity will be resident in the $MFT without having to return to collect non-resident files.

Also, as this article showed some tools may break when they expect the $MFT to be at a certain offset into the volume or if they expect the $MFT record to be 1024 bytes in length instead of 4096 bytes. The good news is, I did test several popular tools including analyzeMFT.py, mft.pl and log2timeline on the $MFT file from my test scenario. Each of these tools were able to parse the extracted $MFT file properly.

Posted 20th March 2013 by Kyle Oetken

Labels: dfir, forensics, mft

4 View comments

Keydet89 March 22, 2013 at 9:08 AM

Thanks for sharing this...when you posted this, I was teaching our Windows Forensic Analysis course, and the question of "if 4K records are used, will there be more data in the resident files?" came up. The MFT parser I use can easily parse the larger records, but I hadn't seen a system yet where 4K MFT records were used.

Reply

Keydet89 March 22, 2013 at 10:09 AM

I hope to see more blog posts like this in the future...

Reply

Mike March 22, 2013 at 10:30 AM

I have done some testing of 4k sector drives. In my case Windows 7 changed $MFT record size to match sector size. Blog post and test image available here: http://www.writeblocked.org/index.php/blog/19-4096-byte-sector-drives-ntfs-and-forensic-tools.html

Reply

Eric Zimmerman June 13, 2013 at 9:01 AM

X-Ways Forensics is NOT broken by 4k record size. I tested 17.2 today but it seems XWF has supported 4k since at least 16.1.

Thanks for the test image Mike!

Reply

Enter your comment...

Comment as: Google Accou ▼

Publish Preview