CYBERDEFNERD

Krzysztof Gajewski's blog

2022-10-20 BY KRZYSZTOF GAJEWSKI

The $MFT flag that you have never considered before – OneDrive not synchronized files.

This article, shows how you can use $MFT flags to find “not synchronized” OneDrive files – files which actually do not exist on the system. And now you may ask yourself a question: “So in $MFT there are entries for files that actually do not exist on the disk?”. Well… yes, and it’s not something new, there are for example orphaned entries that do not necessary point to the files that are still present on the disk. But today I will speak about **OneDrive** files, which in $MFT have entries flagged as “**offline**“.

As you could observe, I said that these files do not exist on the system. And believe me, I spent some time on that part, to prove that indeed it’s a valid theory.

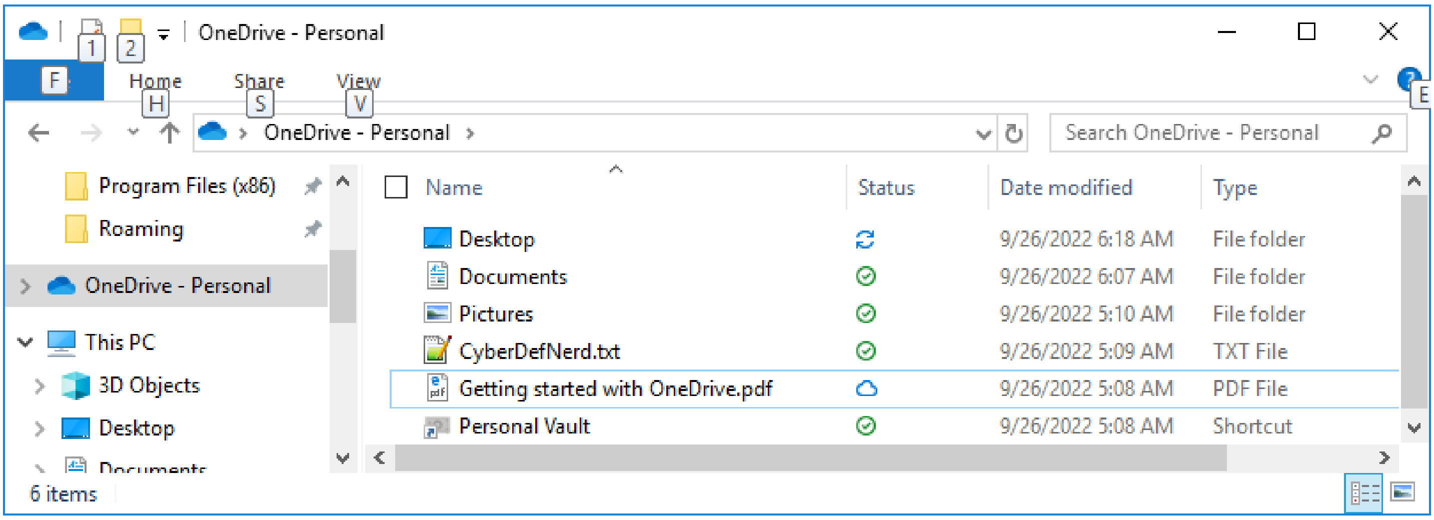
But before we will go further, I want to make sure that everybody knows what I mean when I am saying “no synchronized files”. Therefore, first take a look at a small introduction. If you are not interested in that part, feel free to jump to another “yellow header”.

Synchronized and not synchronized OneDrive files

If you use OneDrive client on your machine, it will automatically try to synchronize files from your system with the cloud storage and vice versa. By default OneDrive client synchronizes files from three folders:
– Pictures
– Desktop
– Documents

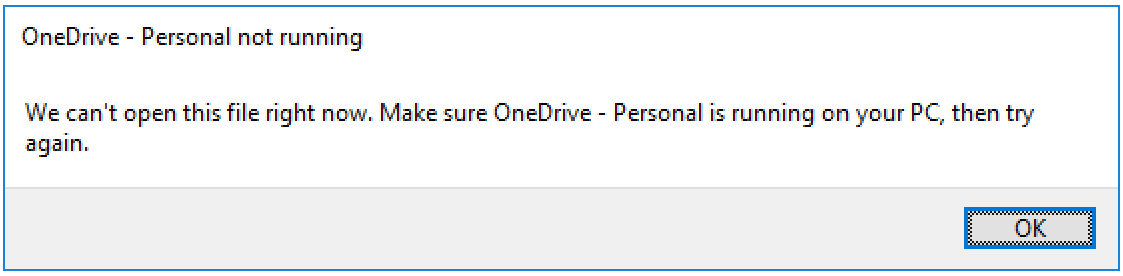
It is be possible, that some files are available in the cloud and were not synchronized with your device (yet). In that case, you may still see that files on your system, but if you try to open it you will get an error.

The GUI access to the device, allows you to quickly find out such files. Just start EXPLORER.EXE and jump to OneDrive directory. Files with the green check mark were synchronized properly, but files with a small blue cloud were not – and still only sit in the cloud (screenshot below).



Name	Status	Date modified	Type
Desktop	⟳	9/26/2022 6:18 AM	File folder
Documents	✓	9/26/2022 6:07 AM	File folder
Pictures	✓	9/26/2022 5:10 AM	File folder
CyberDefNerd.txt	✓	9/26/2022 5:09 AM	TXT File
Getting started with OneDrive.pdf	☁	9/26/2022 5:08 AM	PDF File
Personal Vault	✓	9/26/2022 5:08 AM	Shortcut

If you tried to open a "not synchronized" file, the system would automatically try to sync and get that file for you. But what will happen if there is no way to establish a connection with OneDrive storage? To simulate that scenario, I just paused OneDrive and then tried to open the "not synchronized" file (the one showed in the screenshot above – 'Getting started with OneDrive.pdf'). In results I got an error, below you can find a screenshot showing the message:



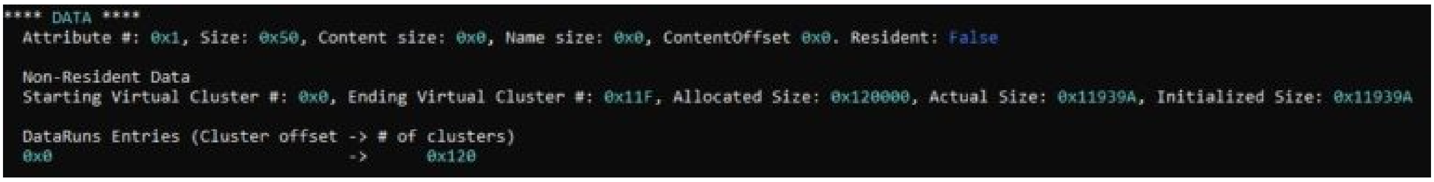
OneDrive - Personal not running

We can't open this file right now. Make sure OneDrive - Personal is running on your PC, then try again.

There was not way to open the file.

Are "not synchronized" OneDrive files present on the system or not?

In the first section, I said that "not synchronized" OneDrive files do not exist on the system. And it is true, indeed the space for them may be allocated, but there is no content. To prove it, I imaged a disk for the testing VM, and did not find any entry for the "not synchronized" OneDrive file. Of course in $MFT there is an entry for that file, but looking at $DATA, you will not find any clusters storing the content (screenshot below).



```
**** DATA ****
  Attribute #: 0x1, Size: 0x50, Content size: 0x0, Name size: 0x0, ContentOffset 0x0. Resident: False

  Non-Resident Data
  Starting Virtual Cluster #: 0x0, Ending Virtual Cluster #: 0x11F, Allocated Size: 0x120000, Actual Size: 0x11939A, Initialized Size: 0x11939A

  DataRuns Entries (Cluster offset -> # of clusters)
  0x0                              ->      0x120
```

Synchronized and not synchronized OneDrive files in $MFT

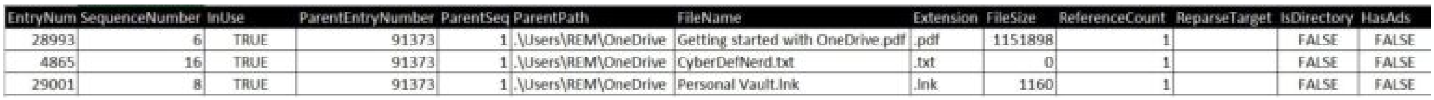
Okay, but what if you only have a forensics collection and there is no way to connect to the system ? In that situation, probably you would take a look at $MFT, and assume that all files listed here, are or at least were (during the forensics collection) present on the system. But… with OneDrive files it may not be true!

I used three tools to parse $MFT:

- **mft2.exe** (Harlan Carvey)
- **MFTDump_V.1.3.0** (Michael G. Spohn, the tool used to be available on http://www.malware-hunters.net long time ago)
- **MFTECmd** version 1.2.2.0 (Eric Zimmerman)

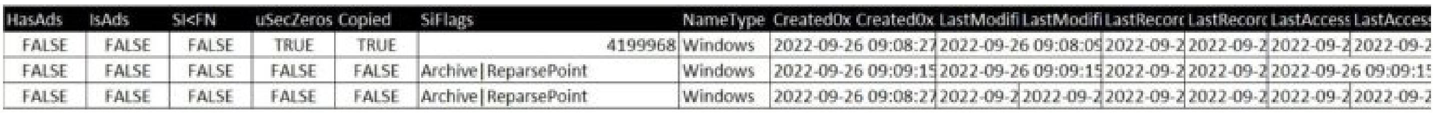
The order in which I listed them, is not accidental. I always use Harlan's tool first, because it gives me the output in MACB and TLN format. Then if I need, I use **MFTDump_V.1.3.0** to get the MFT entry number, which in turn I take and provide to Eric's tool to print all data for a specific file (if I want to get ADS or resident data). But for the purpose of that article, I will present my findings in a reversed order, because only Harlan's tool gave me the output that allowed me to understand how to distinguish synchronized files, from these that still sit in the cloud – so I will leave it at the end.

So first I parsed the $MFT using the Eric Zimmerman tool called **MFTECmd** and searched for a file "Getting started with OneDrive.pdf" – this is the file which was not synchronized on my VM. I found the entry for that file in $MFT, and at the first glance I could not find anything abnormal, so I compared the entry for that file with other entries for files that were present on the system. Below you can find two screenshots demonstrating that:



EntryNum	SequenceNumber	InUse	ParentEntryNumber	ParentSeq	ParentPath	FileName	Extension	FileSize	ReferenceCount	ReparseTarget	IsDirectory	HasAds
28993	6	TRUE	91373	1	.\Users\REM\OneDrive	Getting started with OneDrive.pdf	.pdf	1151898	1		FALSE	FALSE
4865	16	TRUE	91373	1	.\Users\REM\OneDrive	CyberDefNerd.txt	.txt	0	1		FALSE	FALSE
29001	8	TRUE	91373	1	.\Users\REM\OneDrive	Personal Vault.lnk	.lnk	1160	1		FALSE	FALSE

(click to zoom in)



HasAds	IsAds	SI<FN	uSecZeros	Copied	SiFlags	NameType	Created0x	Created0x	LastModifi	LastModifi	LastRecorc	LastRecorc	LastAccess	LastAccess
FALSE	FALSE	FALSE	TRUE	TRUE	4199968	Windows	2022-09-26 09:08:27		2022-09-26 09:08:09		2022-09-2	2022-09-2	2022-09-2	2022-09-2
FALSE	FALSE	FALSE	FALSE	FALSE	Archive|ReparsePoint	Windows	2022-09-26 09:09:15		2022-09-26 09:09:15		2022-09-2	2022-09-2	2022-09-26 09:09:15	
FALSE	FALSE	FALSE	FALSE	FALSE	Archive|ReparsePoint	Windows	2022-09-26 09:08:27		2022-09-2	2022-09-2	2022-09-2	2022-09-2	2022-09-2	2022-09-2

(click to zoom in)

Looking at it, I could observe different values in columns **uSecZeros**, **Copied** and **SiFlags**.

First two are quite clear (if you understand the intention of that columns):
– **uSecZeros**: True if STANDARD_INFO created, modified, or last access has 0s for sub-second

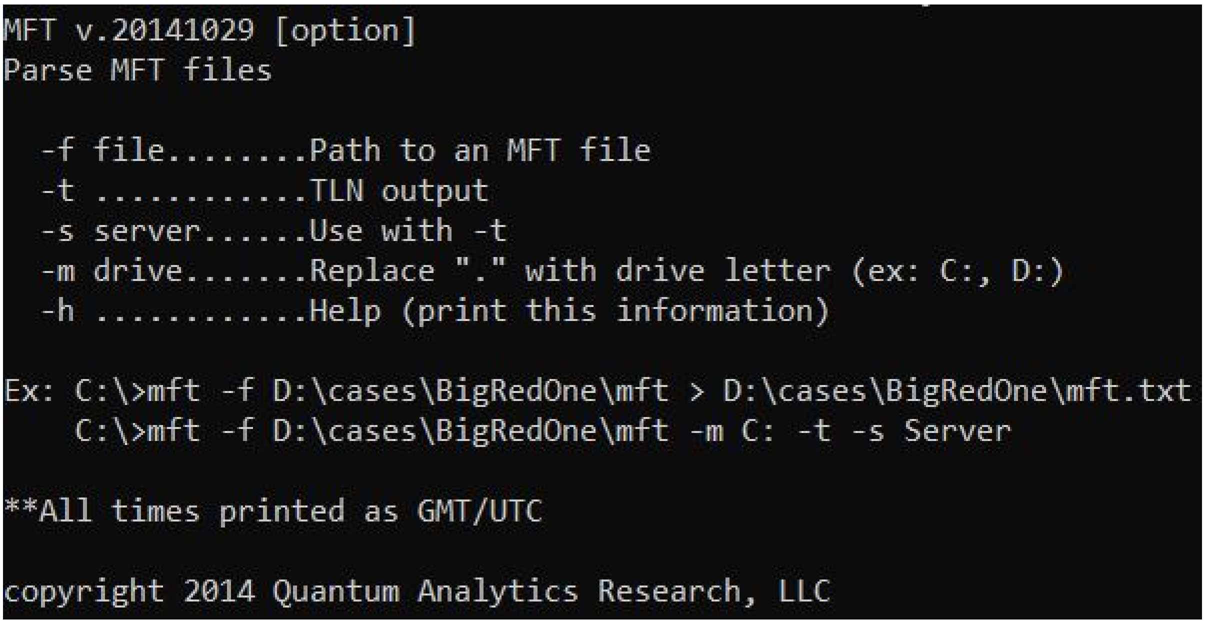
precision
– **Copied**: True if STANDARD_INFO modified < STANDARD_INFO created time

None of them can be used to clearly indicate that a OneDrive file was not properly synchronized (more information about columns you can find here).

But we still have the third column, named **SiFlags**. Here we should have all flags (I will speak about them later) set for that $MFT entry, but for the file in question we do not have listed them in a nice readable way – just "some number". I think that you can agree with me, that the most of us would just ignore that difference (I think I would…).

As the number representing flags, did not allow me to understand if the files were synchronized or not, I used another tool to parse $MFT, this time it was **MFTDump_V.1.3.0**. The tool supports two output formats, the **standard** one and the **long** one. I used the standard one only, and it did not list flags at all. It looks like there are separate columns for Hidden and System flag, but nothing else

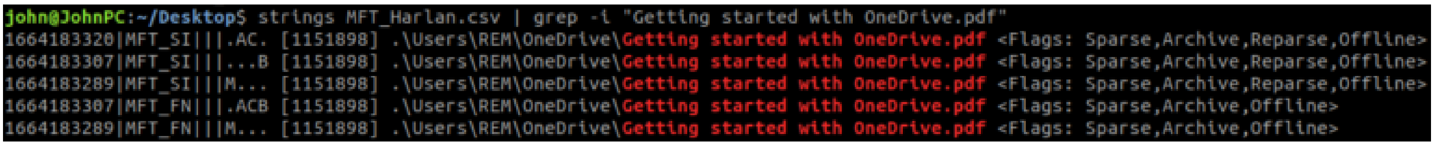
And then I tested the last tool, namely **mft2.exe**:



```
MFT v.20141029 [option]
Parse MFT files

  -f file........Path to an MFT file
  -t ............TLN output
  -s server......Use with -t
  -m drive.......Replace "." with drive letter (ex: C:, D:)
  -h ............Help (print this information)

Ex: C:\>mft -f D:\cases\BigRedOne\mft > D:\cases\BigRedOne\mft.txt
    C:\>mft -f D:\cases\BigRedOne\mft -m C: -t -s Server

**All times printed as GMT/UTC

copyright 2014 Quantum Analytics Research, LLC
```

(mft2.exe v.20141029)

Once I parsed the $MFT, I did a search for a file in question:



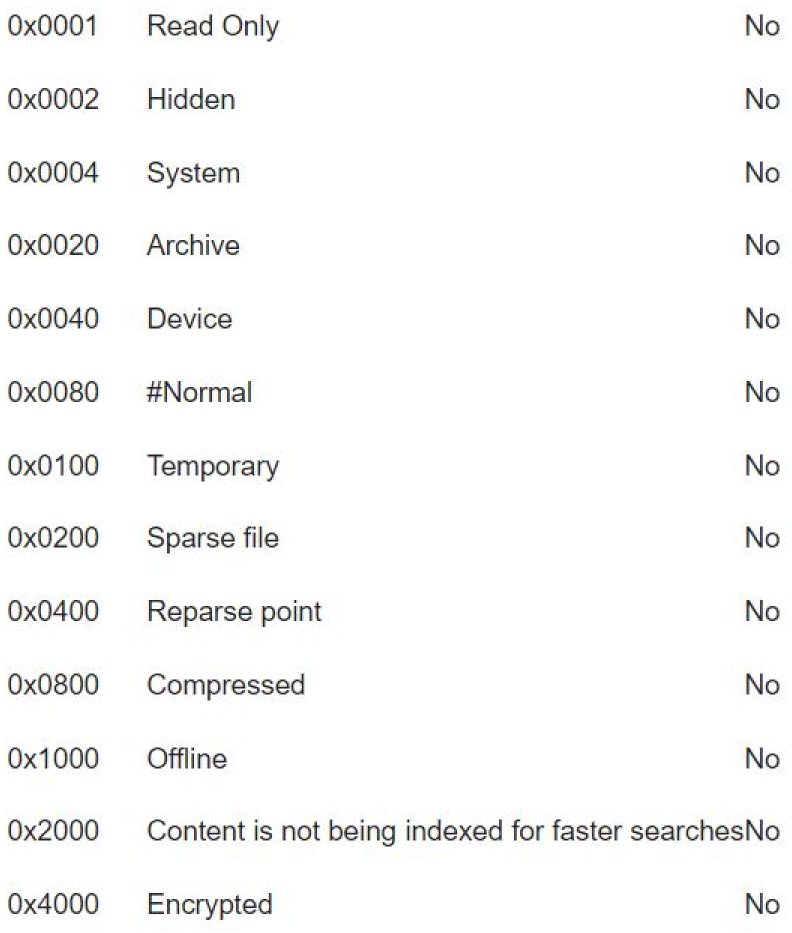
```
john@JohnPC:~/Desktop$ strings MFT_Harlan.csv | grep -i "Getting started with OneDrive.pdf"
1664183320|MFT_SI|||.AC. [1151898] .\Users\REM\OneDrive\Getting started with OneDrive.pdf <Flags: Sparse,Archive,Reparse,Offline>
1664183307|MFT_SI|||...B [1151898] .\Users\REM\OneDrive\Getting started with OneDrive.pdf <Flags: Sparse,Archive,Reparse,Offline>
1664183289|MFT_SI|||M... [1151898] .\Users\REM\OneDrive\Getting started with OneDrive.pdf <Flags: Sparse,Archive,Reparse,Offline>
1664183307|MFT_FN|||.ACB [1151898] .\Users\REM\OneDrive\Getting started with OneDrive.pdf <Flags: Sparse,Archive,Offline>
1664183289|MFT_FN|||M... [1151898] .\Users\REM\OneDrive\Getting started with OneDrive.pdf <Flags: Sparse,Archive,Offline>
```

(search for "Getting started with OneDrive.pdf")

Now I immediately got a new flag, that I think I have never seen before (or I did not pay attention to it), namely a flag called **OFFLINE**. That flag seems to be used to mark files, that do not have any real content on the disk. Going further, you can use that flag to find all "not synchronized" OneDrive files, which in my opinion is SUPER handy.

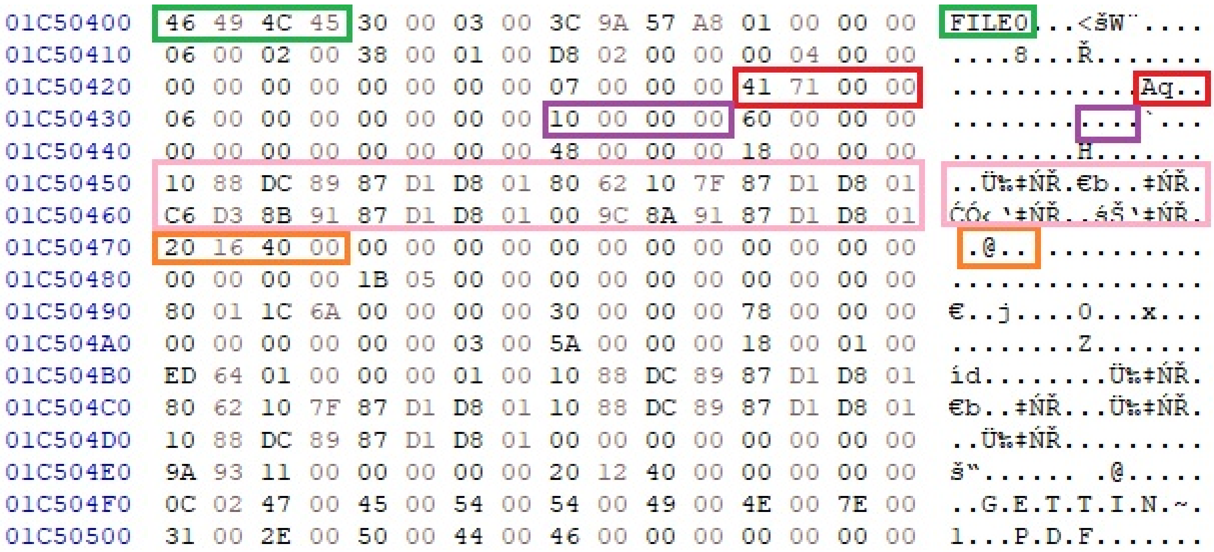
$STANDARD_INFORMATION flags

If you are not aware of $MFT flags, you have to jump back to your $MFT documentation (books, training materials etc.) and check where that flags can be found. For a quick reference you can use this link. It says, that **$STANDARD_INFORMATION** attributes may have several different flags. One of them is **Offline**. But keep in mind, that **$FILE_NAME** has them as well.



0x0001	Read Only	No
0x0002	Hidden	No
0x0004	System	No
0x0020	Archive	No
0x0040	Device	No
0x0080	#Normal	No
0x0100	Temporary	No
0x0200	Sparse file	No
0x0400	Reparse point	No
0x0800	Compressed	No
0x1000	Offline	No
0x2000	Content is not being indexed for faster searches	No
0x4000	Encrypted	No

To make sure that Harlan's tool found all flags properly, I checked $MFT manually (that allowed me to better understand the structure of $MFT). To find that flags, first I had to find the $MFT entry number for that file, which was **28993**. But how to use that number to find a correct place in the $MFT? I converted that number to a hexadecimal value, which gave me **7141**. Having that, I had to reverse the order of bytes and add **00 00** at the end. In results I got **41 71 00 00**. Now by looking for that byte sequence in Hex Editor (CTRL + F), I found an entry for my file. Moving further, and finding bytes for all "fields", I found that at offset **01C50470** there are four bytes reserved for flags – in that case it was **20 16 40 00**.



```
01C50400  46 49 4C 45 30 00 03 00 3C 9A 57 A8 01 00 00 00  FILE0...<šW¨....
01C50410  06 00 02 00 38 00 01 00 D8 02 00 00 00 04 00 00  ....8...Ř.......
01C50420  00 00 00 00 00 00 00 00 07 00 00 00 41 71 00 00  ............Aq..
01C50430  06 00 00 00 00 00 00 00 10 00 00 00 60 00 00 00  ............`...
01C50440  00 00 00 00 00 00 00 00 48 00 00 00 18 00 00 00  ........H.......
01C50450  10 88 DC 89 87 D1 D8 01 80 62 10 7F 87 D1 D8 01  ..Ü‰‡ŃŘ.€b..‡ŃŘ.
01C50460  C6 D3 8B 91 87 D1 D8 01 00 9C 8A 91 87 D1 D8 01  ĆÓ‹‘‡ŃŘ..śŠ‘‡ŃŘ.
01C50470  20 16 40 00 00 00 00 00 00 00 00 00 00 00 00 00   .@.............
01C50480  00 00 00 00 1B 05 00 00 00 00 00 00 00 00 00 00  ................
01C50490  80 01 1C 6A 00 00 00 00 30 00 00 00 78 00 00 00  €..j....0...x...
01C504A0  00 00 00 00 00 00 03 00 5A 00 00 00 18 00 01 00  ........Z.......
01C504B0  ED 64 01 00 00 00 01 00 10 88 DC 89 87 D1 D8 01  íd........Ü‰‡ŃŘ.
01C504C0  80 62 10 7F 87 D1 D8 01 10 88 DC 89 87 D1 D8 01  €b..‡ŃŘ...Ü‰‡ŃŘ.
01C504D0  10 88 DC 89 87 D1 D8 01 00 00 00 00 00 00 00 00  ..Ü‰‡ŃŘ.........
01C504E0  9A 93 11 00 00 00 00 00 20 12 40 00 00 00 00 00  š“...... .@.....
01C504F0  0C 02 47 00 45 00 54 00 54 00 49 00 4E 00 7E 00  ..G.E.T.T.I.N.~.
01C50500  31 00 2E 00 50 00 44 00 46 00 00 00 00 00 00 00  1...P.D.F.......
```

20 00 00 00 – stands for **Archive**

00 10 00 00 – stands for **Offline**

00 02 00 00 – stands for **Sparse File**

00 04 00 00 – stands for **Reparse Point**

In total it gives **20 16**, but I do not know what remaining two bytes **40 00** mean. If you know it, please explain that in the comments.

So everything looks okay! I checked if there are other files with that flag, and I found other not synchronized OneDrive files.

Summary

As you can see, you may use $MFT to find files on the disk, which do not keep any real content. It may be very useful during your investigations. For example, you may be asked to find out what files were exfiltrated during the time of the breach. Let's say that you know what folder was accessed by the attacker, or maybe what folder was archived and sent out. Having that information, probably you would use $MFT to see what files were available under that path in question. But what if some files sitting there were not synchronized? If you attached them in the report for the client, you would basically provide him **incorrect** information! That information in turn could impact client's reputation or even led to some financial penalties. OneDrive synchronization may be stopped due to SEVERAL different reasons, not only because there is not network connection.

FORENSICS