



The Meaning of

Linkfiles

In

Forensic

Examinations

A look at the practical value to forensic examinations of dates and times, and object identifiers in Windows shortcut files.

Introduction

I have always considered that there should be some forensically useful conclusions that could be drawn from the different dates and times associated with Windows Shortcut Files (referred to here as link files). A common request to an examiner might be “can you tell whether the suspect has viewed this file after it has been downloaded”; the aim of this paper is to answer that question and at the same time provide other related information that will be of practical value in computer examinations.

Each link file has its own Created, Modified and Accessed dates and within each link file there are Created, Modified and Accessed dates which belong to the target file. In addition, if the target file still exists on the media, that file has its own three dates.

The purpose of this paper is to explore how these nine dates relate to each other and what conclusions can be drawn from the relationships that exist. The reader is invited to participate in the development of this paper by carrying out their own experiments and reporting the results back to the authorⁱ.

The only link files referred to in this paper are those which link to recent files (as opposed to Internet Shortcuts). These link files are used on the “My Recent Documents” list in XP, and “Recent Items” in Vista. They are to be found in the following locations,

Windows XP

\Documents and Settings*UserName*\Recent and
\Documents and Settings*UserName*\Application Data\Microsoft\Office\Recent

Windows Vista

\Users*UserName*\AppData\Roaming\Microsoft\Windows\Recent and
\Users*UserName*\AppData\Roaming\Microsoft\Office\Recent

In addition to the dates within link files there may be Link Tracking information embedded in a link file which can provide information about the origins, history and movement of the target file.

The file system observed has been NTFS.

Note on Experiments

I would be interested to receive comments on this paper and the results of any experiments that you have conducted to test the following observations.

Any experiment will require you to capture 1) the file metadata for the target file prior to it being accessed, followed by 2) the content of the link file itself after the access, together with the link file’s metadata, and finally 3) the metadata of the target file after it has been accessed.

You should explain how you have ensured the experiment is forensically sound and the version of the Windows OS.

Note on File Dates and Times

In forensic terms Accessed Dates are often of little value as a change does not necessarily mean the file has been accessed by the user.

The NTFS file system delays updates to the last access time for a file by up to 1 hour after the last access(1). Updates to the last access time can actually be switched off using the command line tool *fsutil* (2) or by manually setting the value of the registry entry for `NtfsDisableLastAccessUpdate` to 1. In Vista the default value is set to 1 (Last Accessed disabled) whereas in XP default is 0 (enabled). According to Microsoft (1) "Timestamps are updated at various times and for various reasons. The only guarantee about a file timestamp is that the file time is correctly reflected when the handle that makes the change is closed." It is not clear what `NtfsDisableLastAccessUpdate` actually means, according to TechNet in one instance the value of 0 is defined to mean,

"updates the last-accessed timestamp of a file whenever that file is opened, "

and in another,

"when listing directories, NTFS updates the last-access timestamp on each directory it detects, and it records each time change in the NTFS log."

In reality it appears to be a combination of the two.

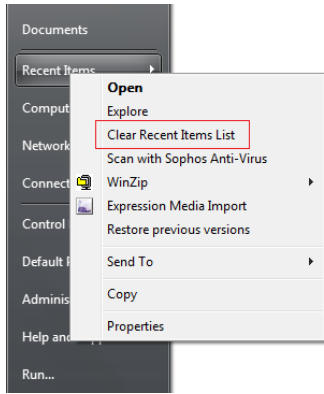
I have seen numerous references in forensic papers and presentations to the effect of this registry change in Vista, including one comment that "`NtfsDisableLastAccessUpdate` is now 1, which means no last access timestamp will be written at all".

Disabling last access update **does not** mean that the Accessed Date on files does not get updated *at all*; it means that it does not get updated on directory listing or file opening, but last accessed can sometimes be updated when a file is modified and is updated when a file is moved between volumes. The exact conditions under which a file's Accessed Date is updated on modification are unclear; it appears to vary depending upon file type and/or the application used to modify the file.

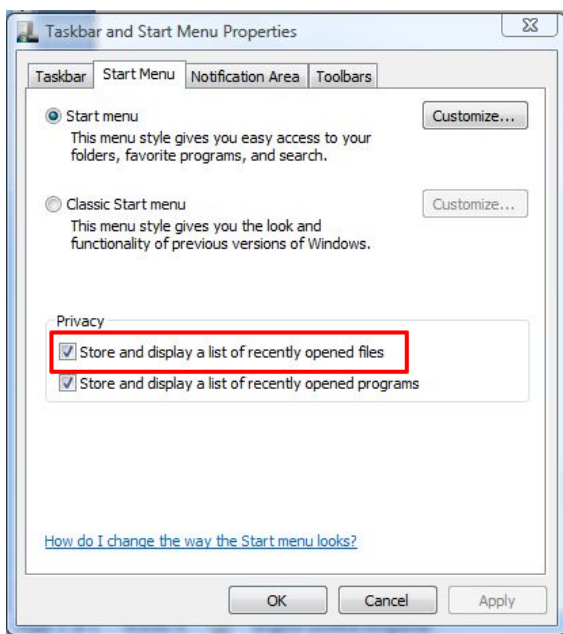
When references are made to XP or Vista below it is in the context that in XP `NtfsDisableLastAccessUpdate` is disabled and in Vista it is enabled.

Some general notes on Link Files

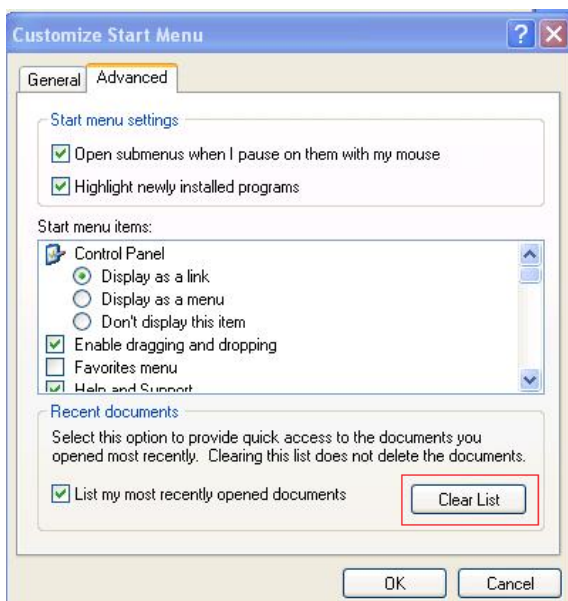
In Vista the Recent List of documents on the Windows Start Menu can be cleared by right-clicking Recent Items and selecting Clear Recent Items.



The option to store and display a list of recently opened files is in the Start Menu properties under Privacy.



In XP this is accessed via the properties of the Start Menu.



The above settings can be examined in the windows registry at
HKEY_CURRENT_USER\Software\Microsoft\Windows\CurrentVersion\Explorer\Advanced

In XP

Start_ShowRecentDocs = 0 indicates the option to list Recent Documents is unchecked

Start_ShowRecentDocs = 2 indicates the option is checked




In Vista

Start_ShowRecentDocs = 0 and Start_TrackDocs = 0 indicates the option is unchecked

Start_ShowRecentDocs = 2 and Start_TrackDocs = 1 indicates the option is checked

Recent items are created when an application calls the Windows Shell Function SHAddToRecentDocs() (3). Windows itself calls this function in certain cases, when activating a file in explorer, and when using the file open and save dialog. Any application can call this function and add a file to the Recent Items list. It is possible to clear the Recent Items list by calling SHAddToRecentDocs() with the *pv* parameter set to NULL.

A registry setting can be used to prevent members of a file class being added to the Recent Items list. An EditFlags value 0x00100000 can be set for the file association ProgID key in HKEY_CLASSES_ROOT. For example, it is likely that you will find that the ProgID key for the .msc file at HKEY_CLASSES_ROOT\mscfile looks like this,

Name	Type	Data
 (Default)	REG_SZ	Microsoft Common Console Document
 EditFlags	REG_DWORD	0x00100000 (1048576)
 FriendlyTypeName	REG_EXPAND_SZ	@%SystemRoot%\system32\mmcbase.dll,-130

So if you were to open services.msc for example, this setting prevents it appearing in the Recent Items list. Change the value to zero and services.msc will appear on the Recent Items List when it is opened.

Observation One

When a target file is opened and a link file is created, the created date of the link file remains the date that target file was first accessed during the lifetime of that link file. If the target file is opened subsequently, the Created Date of the link file remains the same.

The following example illustrates this observation.

Here are the properties of a link file, note the Created Date 13/07/2008 17:38:30

General	
Name	squad_selector_66x66.gif.lnk
File Class	Regular file
File Size	555
Date Accessed	13/07/2008 17:38:30
Date Created	13/07/2008 17:38:30
Date Modified	13/07/2008 17:38:30
Encrypted	<input type="checkbox"/>
Compressed	<input type="checkbox"/>

These are the properties of the same link file after the target file has been opened several times. The Created Date is unchanged.

General	
Name	squad_selector_66x66.gif.lnk
File Class	Regular file
File Size	639
Date Accessed	13/07/2008 17:54:47
Date Created	13/07/2008 17:38:30
Date Modified	13/07/2008 17:54:47
Encrypted	<input type="checkbox"/>
Compressed	<input type="checkbox"/>

Observation Two

Once a link file has been created for a target file with a given filename, during the lifetime of that link file, if another target file of the same name is accessed from a different location, the original link file for that given filename is updated. *Observation One* applies and the Created Date for the link file remains the same.

The link file name and the Created Date are the only artefacts from the original link file, all the internal detail of the link file is overwritten, and it appears from observation that the same MFT file record is used.

Observation Three

At the time a target file is opened the Created, Accessed and Modified dates of the target file are read and stored within the associated link file at offset 0x1C. Each date is recorded in the FILETIME data type (1) in 8 bytes. The dates are in the order Created, Accessed, Modified.

The following shows the Created Date embedded in the link file.

00228806656	4C 00 00 00 01 14 02 00 00 00 00 00 C0 00 00 00	L	À
00228806672	00 00 00 46 9B 00 00 00 20 00 00 00 D0 1F 9A C3	F	Ë
00228806688	CB E5 C8 01 80 29 13 7A CC E5 C8 01 80 2C D0 B7	Ë	z
00228806704	CB E5 C8 01 42 1F 00 00 00 00 00 00 01 00 00 00	Ë	Ë

Data Interpreter

FILETIME: 14/07/2008 16:07:50

Observation Four

The Modified Date of the link file represents the time when the related target file was last opened (as opposed to when it was closed).

Where the Created, Accessed and Modified dates of the link file are the same, this indicates that the target file has not been opened since that time.

In Vista the Modified and Accessed times will always be the same as when the link file is modified the Accessed time always gets updated. In XP the Accessed time could be different to the Modified time.

Observation Five

Where a new file has been created in an application and then saved from it, and a link file has been created, the link file will not contain any embedded dates relating to the target file.

The following shows the start of such a link file with no embedded dates.

000	4c 00 00 00 01 14 02 00-00 00 00 00 c0 00 00 00	L.....Ä...
010	00 00 00 46 9b 00 28 00-00 00 00 00 00 00 00 00	...F..(.....
020	00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00
030	00 00 00 00 00 00 00 00-00 00 00 00 01 00 00 00
040	00 00 00 00 00 00 00 00-00 00 00 00 90 00 14 00
050	1f 44 47 1a 03 59 72 3f-a7 44 89 c5 55 95 fe 6b	.DG..Yr?SD.ÄU.pk
060	30 ee 20 00 00 00 1a 00-ee bb fe 23 00 00 10 00	0i.....i»p#.....
070	30 81 e2 33 1e 4e 76 46-83 5a 98 39 5c 3b c3 bb	0-â3.NvF.Z.9\;Ä»
080	00 00 5a 00 32 00 00 00-00 00 00 00 00 00 80 00	..Z.2.....
090	76 67 61 30 32 73 6d 2e-6a 70 67 00 40 00 07 00	vga02sm.jpg-@...

The same generally applies where a picture, for example, is saved from a web page by right click, Save Picture As.

It has been observed that while this appears to be the general case there have been instances where a file has been saved in both ways described above and the link file has contained embedded dates.

This certainly happens when a new file is saved with an existing filename overwriting the original file. In these circumstances the link file embedded dates will be taken from the existing file that is overwritten.

Where a picture has been saved and there are no embedded dates in the link file, if it is subsequently opened the link file will thereafter contain embedded dates.

The definite exception to the Observation Five is in the case of files saved by Microsoft Office applications (2003 & 2007). When a file is created in an Office application and it is first saved, a link file is created in both the user's Recent folder and Office Recent folder. The link file in the Office Recent folder appears from observation to always contain embedded dates when it is first created but the one in the Recent folder contains no embedded dates.

The conclusion to be drawn from Observation Five is that if a link file does not contain embedded dates that target file has not been opened since the link file was created. The converse is not necessarily true but the content of the embedded dates will be an indicator, i.e. if they are almost contemporaneous with the link file Created, Accessed and Modified then the target file has not been opened since the link file was created.

Some Practical Examples

Example One

This on a Vista Operating System.

This file was a picture on a web page, it was viewed in the web page, the user has right clicked and Save Picture As..., and then saved to My Pictures. The chronology is explained as follows.

Temp Int File – 6666[1].jpg			Target File – 6666.jpg			Link File – 6666.jpg.lnk		
❶	Created	13/09/2008 10:17:00	❸	Created	13/09/2008 12:42:29	❷	Created	13/09/2008 12:42:29
	Modified	13/09/2008 10:17:00	❹	Modified	13/09/2008 10:17:00		Modified	13/09/2008 12:42:29
	Accessed	13/09/2008 10:17:00		Accessed	13/09/2008 12:42:29		Accessed	13/09/2008 12:42:29

❸ The internal dates of the link file are zero bytes.

```

000 4c 00 00 00 01 14 02 00-00 00 00 00 c0 00 00 00 L.....À...
010 00 00 00 46 9b 00 28 00-00 00 00 00 00 00 00 00 ..F..(.....
020 00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00 .....
030 00 00 00 00 00 00 00 00-00 00 00 00 01 00 00 00 .....
040 00 00 00 00 00 00 00 00-00 00 00 00 88 00 14 00 .....
050 1f 44 47 1a 03 59 72 3f-a7 44 89 c5 55 95 fe 6b -DG-Yr?SD·AU·pk
060 30 ee 20 00 00 00 1a 00-ee bb fe 23 00 00 10 00 0i .....i»p#....
070 30 81 e2 33 1e 4e 76 46-83 5a 98 39 5c 3b c3 bb 0-â3·NvF·Z·9\;Ã»
080 00 00 52 00 32 00 00 00-00 00 00 00 00 00 80 00 ..R·2.....
090 36 36 36 36 2e 6a 70 67-00 00 3a 00 07 00 04 00 6666.jpg...:.....
  
```

❶	The picture has been downloaded as part of the web page and the three file dates are the same.
❷	The Target Modified Date shows the time it was last modified on this computer when it was saved in the cache on initial download.
❸	The Target Created Date shows the date the file was saved in My Pictures, the gap in time from the Modified Date is the difference in time from when it was displayed initially in the web page to when it was saved in My Pictures.
❹	The Link File three dates are when the Target File was saved in My Pictures.
❺	The internal dates of the link file are all zero bytes and this, together with ❷, indicate the file has not been opened since saving.

Example Two

The following is found when the picture is subsequently opened in a viewer application; this might be a common scenario in an investigation.

Internal Link File Dates			Created 13/09/2008 12:42:29	Accessed 13/09/2008 12:42:29	Modified 13/09/2008 10:17:00
6	Created	13/09/2008 12:42:29	<pre> 000 44 00 00 00 01 14 02 00 30 00 00 00 c0 00 00 00 A... 010 00 00 00 46 9b 00 28 00 20 00 00 00 d0 da 2e ...F...EMU 020 9e 15 c9 01 00 f8 e6 2e 9e 15 c9 01 d0 5b 0b dc -E-ae.-E-B[U 030 89 15 c9 01 41 0a 00 00-00 00 00 01 00 00 00 -E A..... 040 00 00 00 00 00 00 00-00 00 00 00 88 00 14 00 050 1f 44 47 1a 03 59 72 3f-a7 44 89 c5 55 95 fe 6b -DG-Yr?SDAU jk 060 30 ee 20 00 00 00 1a 00-ee bb fe 23 00 00 10 00 0Ii>#... 070 30 81 e2 33 1e 4e 76 46-83 5a 98 39 5c 3b c3 bb 0-83.NvF.Z-9\A 080 00 00 52 00 32 00 41 0a-00 00 24 39 21 52 20 00 -R.Z.A.--9!R - 090 36 36 36 36 2e 6a 70 67-00 00 3a 00 07 00 04 00 6666.jpg...:..... </pre>		
	Modified	13/09/2008 10:17:00			
	Accessed	13/09/2008 12:42:29			

Temp Int File – 6666[1].jpg			Target File – 6666.jpg			Link File - 6666.jpg.lnk		
1	Created	13/09/2008 10:17:00	3	Created	13/09/2008 12:42:29	4	Created	13/09/2008 12:42:29
	Modified	13/09/2008 10:17:00	2	Modified	13/09/2008 10:17:00	5	Modified	19/09/2008 19:57:39
	Accessed	13/09/2008 10:17:00		Accessed	13/09/2008 12:42:29		Accessed	19/09/2008 19:57:39

1	13/09/2008 10:17:00	The picture has been downloaded as part of the web page and the three file dates are the same.
2	13/09/2008 10:17:00	The Target Modified Date shows the time it was last modified on this computer when it was saved in the cache on initial download.
3	13/09/2008 12:42:29	The Target Created Date shows the date the file was saved in My Pictures, the gap in time from the Modified Date is the difference in time from when it was displayed initially in the web page to when it was saved in My Pictures.
4	13/09/2008 12:42:29	The Link File Created date is when the Target File was first accessed in My Pictures.
5	19/09/2008 19:57:39	The Modified and Accessed dates of the link file reflect the last time that the Target File was opened.
6		The Internal dates in the Link File are the same as the Target File dates and this shows that it has not been modified the last time it was accessed.

Example Three

In the following case the Target File has been edited.

Internal Link File Dates		
	Created	13/09/2008 12:42:29
4	Modified	20/09/2008 19:55:23
	Accessed	20/09/2008 19:55:23

```

000 4c 00 00 00 01 14 02 00-00 00 00 00 c0 00 00 00 L.....À...
010 00 00 00 46 9b 00 28 00-20 00 00 00 d0 4d da 2e ...F..( ...EMÓ.
020 9e 15 c9 01 b1 5e c5 d1-5a 1b c9 01 21 52 cb d1 ..É±ÃÑZÉ!REÑ
030 5a 1b c9 01 69 2d 00 00-00 00 00 00 01 00 00 00 Z.É-i-.....
040 00 00 00 00 00 00 00-00 00 00 00 00 00 88 00 14 00 .....
050 1f 44 47 1a 03 59 72 3f-a7 44 89 c5 55 95 fe 6b .DG..Yr?SD.ÀU·pk
060 30 ee 20 00 00 00 1a 00-ee bb fe 23 00 00 10 00 01 .....iþ#.....
070 30 81 e2 33 1e 4e 76 46-83 5a 98 39 5c 3b c3 bb 0.ã3·NvF·Z·9\;Ã»
080 00 00 52 00 32 00 69 2d-00 00 34 39 ec 9e 20 00 .R·2·i-...49i...
090 36 36 36 36 2e 6a 70 67-00 00 3a 00 07 00 04 00 6666.jpg:.....
  
```

Temp Int File – 6666[1].jpg			Target File – 6666.jpg			Link File - 6666.jpg.lnk		
1	Created	13/09/2008 10:17:00	2	Created	13/09/2008 12:42:29	3	Created	13/09/2008 12:42:29
	Modified	13/09/2008 10:17:00	6	Modified	20/09/2008 20:01:59	5	Modified	20/09/2008 20:01:31
	Accessed	13/09/2008 10:17:00		Accessed	20/09/2008 20:01:59		Accessed	20/09/2008 20:01:31

1	13/09/2008 10:17:00	The picture has been downloaded as part of the web page and the three file dates are the same.
2	13/09/2008 12:42:29	The Target Created Date shows the date the file was saved in My Pictures
3	13/09/2008 12:42:29	The Link File Created date is when the Target File was first accessed.
4	20/09/2008 19:55:23	The Link File internal Modified and Accessed Dates reflect the Target File properties at the time the Target File was last opened and show it was modified at 19:55:23.
5	20/09/2008 20:01:31	The Modified and Accessed dates of the link file reflect the last time that the Target File was opened.
6	20/09/2008 20:01:59	The Target File Modified Date shows that the file was modified after it was last opened.

Link Tracking and ObjectIDs

(4) The Distributed Link Tracking Service maintains its link to an object by using an object identifier (ObjectID). An ObjectID is an optional attribute that uniquely identifies a file or directory on a volume. An index of all ObjectIDs is stored on the volume. Rename, backup, and restore operations preserve object IDs. However, copy operations do not preserve ObjectIDs, because that would violate their uniqueness. ObjectIDs are not maintained on FAT systems or removable media.

A FileLocation(4) describes the location of a file at some point in time, it is made up of a VolumeID and an ObjectID, and both are 16 bytes in length. The system keeps track of files that are moved and so a PreviousFileLocation(4) in the same format can be stored. The low order bit of a VolumeID must be zero, thus you should always find that the VolumeID is an even number.

On NTFS when a file is opened and a Link File is created, two FileLocations are embedded at the end of the Link File. The last four bytes of the Link File are zero and the 64 preceding bytes contain two FileLocations. The two are the same unless a file has been moved between two NTFS volumes.

This is a typical example,

```

130 | 00 a0 58 00 00 00 00 00-00 00 66 65 72 67 75 73 | . X . . . . . fergus
140 | 6d 61 63 6c 65 69 64 65-00 00 3a 6c 1a 42 40 27 | macleide...:l·B@'
150 | 96 45 aa bd 16 21 72 94-df 12 92 1a 55 50 81 88 | .E*¼·!r·B...UP..
160 | dd 11 b8 b3 00 1d 09 16-60 c9 3a 6c 1a 42 40 27 | Ý·,³...·É:l·B@'
170 | 96 45 aa bd 16 21 72 94-df 12 92 1a 55 50 81 88 | .E*¼·!r·B...UP..
180 | dd 11 b8 b3 00 1d 09 16-60 c9 00 00 00 00 | Ý·,³...·É....

```

The 16 bytes starting 3a 6c 1a is the VolumeID of the volume where the file resides. The next 16 bytes starting 92 1a 55 are the ObjectID of the file.

The ObjectID of the volume can be found in the \$Volume file, in the OBJECT_ID attribute 40 00 00 00. The attribute, shown below, is 0x28 in length, the data stream is 0x10 in length and starts at attribute offset 0x18.

```

| 06 00 00 00 00 00 00 00 07 03 24 00 56 00 6F 00 | | $ V o
| 6C 00 75 00 6D 00 65 00 40 00 00 00 28 00 00 00 | l u m e @ (
| 00 00 00 00 00 00 06 00 10 00 00 00 18 00 00 00 |
| 54 DE 5C BC CE 6F 0E 40 A0 9E 83 AA 95 78 91 B6 | TË\¼Ío @ ||@|x'¶

```

The ObjectID of the file is found in the same way in the MFT record for the file (shown in the following illustration).

```

49 4C 45 30 00 03 00 CD 00 0B 59 02 00 00 00 FILE0 I Y
5A 00 02 00 38 00 01 00 00 02 00 00 00 04 00 00 Z 8
00 00 00 00 00 00 00 00 05 00 00 00 8C E4 00 00 I
02 00 47 11 00 00 00 00 10 00 00 00 60 00 00 00 G
00 00 00 00 00 00 00 00 48 00 00 00 18 00 00 00 H
F1 77 99 23 59 1B C9 01 FD E8 43 61 BB 19 C9 01 ŕw!#Y É ý@Ca> É
7D 60 EA 97 AA 1C C9 01 5D 12 EA 97 AA 1C C9 01 }`é!³ É ] é!³ É
20 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00 00 00 00 EB 02 00 00 00 00 00 00 00 00 00 00 00
88 D7 D0 5A 00 00 00 00 30 00 00 00 78 00 00 00 IxBZ 0 x
00 00 00 00 00 00 03 00 5A 00 00 00 18 00 01 00 Z
FF 23 00 00 00 00 06 00 5D 12 EA 97 AA 1C C9 01 ý# ] é!³ É
5D 12 EA 97 AA 1C C9 01 5D 12 EA 97 AA 1C C9 01 ] é!³ É ] é!³ É
5D 12 EA 97 AA 1C C9 01 00 20 00 00 00 00 00 00 ] é!³ É
00 00 00 00 00 00 00 00 20 00 00 00 00 00 00 00
0C 02 4C 00 4F 00 47 00 4F 00 2D 00 57 00 7E 00 L O G O - W ~
31 00 2E 00 47 00 49 00 46 00 66 00 00 00 00 00 00 1 . G I F f
30 00 00 00 78 00 00 00 00 00 00 00 00 00 02 00 0 x
5C 00 00 00 18 00 01 00 FF 23 00 00 00 00 06 00 \ ý#
5D 12 EA 97 AA 1C C9 01 5D 12 EA 97 AA 1C C9 01 ] é!³ É ] é!³ É
5D 12 EA 97 AA 1C C9 01 5D 12 EA 97 AA 1C C9 01 ] é!³ É ] é!³ É
00 20 00 00 00 00 00 00 00 00 00 00 00 00 00 00
20 00 00 00 00 00 00 00 0D 01 6C 00 6F 00 67 00 l o g
6F 00 2D 00 77 00 69 00 64 00 65 00 2E 00 67 00 o - w i d e . g
69 00 66 00 00 00 00 00 40 00 00 00 28 00 00 00 i f @ (
00 00 00 00 00 00 04 00 10 00 00 00 18 00 00 00
92 1A 55 50 81 88 DD 11 B8 B3 00 1D 09 16 60 C9 ' U P ! Y , ' ` É
80 00 00 00 48 00 00 00 01 00 00 00 00 00 01 00 I H
00 00 00 00 00 00 00 00 01 00 00 00 00 00 00 00
40 00 00 00 00 00 00 00 00 20 00 00 00 00 00 00 @
46 16 00 00 00 00 00 00 46 16 00 00 00 00 00 00 F F
31 02 14 9B 04 00 00 00 FF FF FF FF 82 79 02 00 l I ýýýý!y

```

On a live computer the ObjectIDs of a file can be queried using the command line tool fsutil –

```
fsutil objectid query logo-wide.gif
```

```

Object ID : 921a55508188dd11b8b3001d091660c9
BirthVolume ID : 3a6c1a4240279645aabd16217294df12
BirthObjectid ID : 921a55508188dd11b8b3001d091660c9
Domain ID : 00000000000000000000000000000000

```

The Domain ID will always be zero as it is currently unused(2).

In the following example of a Link File the original Target File above has been moved from one NTFS volume to another on the same computer. The File ObjectID is retained but the Link File shows the VolumeID of the volume to which the file has been moved.

```

200 00 66 65 72 67 75 73 6d-61 63 6c 65 69 64 65 00 .fergusmacleide.
210 00 54 de 5c bc ce 6f 0e-40 a0 9e 83 aa 95 78 91 .TB\Wlo@ . . . x.
220 b6 92 1a 55 50 81 88 dd-11 b8 b3 00 1d 09 16 60 ' . . U P . . Y . , ' . . . .
230 c9 3a 6c 1a 42 40 27 96-45 aa bd 16 21 72 94 df É:l·B@'·E²¼!r·B
240 12 92 1a 55 50 81 88 dd-11 b8 b3 00 1d 09 16 60 . . . U P . . Y . , ' . . . .
250 c9 00 00 00 00 É . . .

```

Annotations in the image:

- New VolumeID**: Points to the hex value `b6` at the start of line 220.
- Birth File ObjectID**: Points to the hex value `c9` at the start of line 230.
- Birth VolumeID**: Points to the hex value `12` at the start of line 240.

If fsutil is used to query the file now the following result is seen –

```

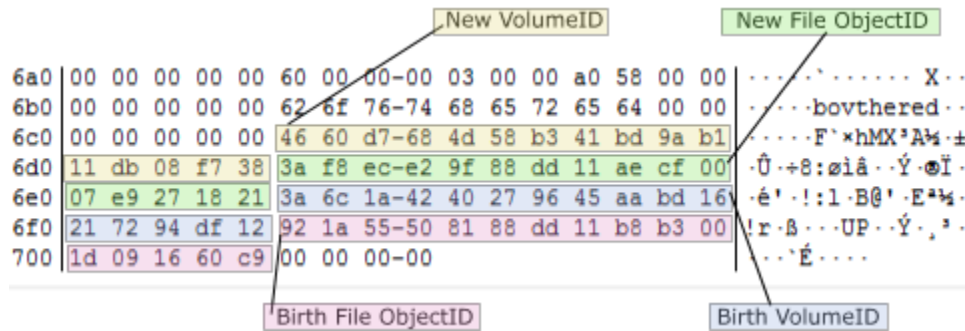
Object ID : 921a55508188dd11b8b3001d091660c9
BirthVolume ID : 3b6c1a4240279645aabd16217294df12
BirthObjectid ID : 921a55508188dd11b8b3001d091660c9
Domain ID : 00000000000000000000000000000000

```

It does not report the ObjectID of the current volume on which the file resides but it can be seen that the BirthVolumeID now starts 3b where it was previously 3a. As mentioned earlier the low order bit of a VolumeID must be zero (3a=111010); this bit is used to indicate whether the file has been

moved. The low order bit represents the CrossVolumeMoveFlag which has a value of 1 if at any time the file has been moved across volumes as it has in this case (3b=111011). Note that the CrossVolumeMoveFlag is not replicated in the VolumeID embedded in the link file.

In the following example of a Link File the original Target File above has been moved from one NTFS volume to another NTFS volume on a different computer. In this case the file is given a new file ObjectID because it has been moved to a different computer.



If fsutil is used to query the file now the following result is seen –
 Object ID : 3af8ece29f88dd11aecf0007e9271821
 BirthVolume ID : 3b6c1a4240279645aabd16217294df12
 BirthObjectID ID : 921a55508188dd11b8b3001d091660c9
 Domain ID : 00000000000000000000000000000000

The new file ObjectID can be seen and the original VolumeID and ObjectID are retained as is the CrossVolumeMoveFlag.

The above examples all relate to files that have been **moved** between volumes; where a file has been copied the file ObjectID will be a new one and the Birth VolumeID will be the volume that the file was copied to.

The ObjectID Structure

The ObjectIDs described above follow the Universally Unique Identifier (UUID) specification found in RFC 4122(5), arguably defined in more detail in ITU-T Rec. X.667(6).

The ObjectIDs for the files can reveal forensically useful information which will be discussed in detail later.

The file ObjectID is a time-based version which means it is created using a system time. The time is a 60 bit time value, a count of 100 nanosecond intervals of UTC since midnight at the start of 15th October 1582. Using an example ObjectID the following is a breakdown according to the specifications.

ObjectID	01DBD30F61FCDD11B5DF0003FF2E1821
Lower order time	01DBD30F
Middle order time	61FC
High order time and version	DD 11
Variant and sequence	B5DF
Node - MAC address	0003FF2E1821

The time values are in little endian order and taking out the **version** number the time value above is

01DBD30F61FCDD**01**

This can be converted back to the system time by taking away the offset between 15th October 1582 and the FILETIME epoch of 1st January 1601, and then using a time and date decoder for FILETIME. The offset is 17 Days in Oct + 30 (Nov) + 31 (Dec) + 18 years and 5 leap days (6), in 100 nanosecond units this is decimal 5748192000000000.

Doing this manually with a calculator –

01DBD30F61FCDD01 (Little Endian)

01DDFC610FD3DB01 (Big Endian)

Convert to decimal

134541057698552577

Subtract 5748192000000000

=

128792865698552577

Convert back to hex

01C9906DD1911B01 (Big Endian)

and convert using a date decoder

=

2009-02-16 19:36:09

Doing the same for the Sequence –

(The Variant bits (2 bits) followed by the high-order bits of the Clock Sequence (6 bits) and the low-order bits of the Clock Sequence (8 bits))

Sequence = B5DF

Binary = **10**11010111011111

The most significant two bits are the Variant (not of any significance)

And the remainder is the Sequence number

11010111011111

= 13791 decimal

The sequence value used in the generation of the ObjectID is taken from the 14 lowest order bits of the registry value –

HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Rpc\UuidSequenceNumber

The MAC address used in the ObjectID is the address of the primary network card in the computer. For systems with no MAC address a random number may be used. In the event that the value in the ObjectID appears to be something other than a MAC address it may be necessary to refer to the

detail of the specifications as several alternative values to the MAC address may be used for this part of the ObjectID.

I have discussed just a few of the lesser well known parts of the link file but there is a lot more information contained within every link file; Microsoft has made the structure of the link file available as part of the MSDN Open Specifications.(7)

How can all this information help me in the examination of a computer?

The first consideration is how to extract the information from the link files. It may be manageable to manually decode a small number of link files but if you want to recover information from all the link files on a computer then an automated process will be required. Your forensic software will probably be able to extract the main details such as file path, volume name and volume serial number of the target file, but to get maximum value from the link files you may need more.

Sanderson Forensics' LinkAlyzer program (8) will carve link files from Disk/Volume/Folder/Encase Image and parse out the details contained within the link file (times and dates, paths, etc) and also decode ObjectIDs presenting the MAC address, time and sequence number.

There is an Enscript available from the Guidance Enscript Resource Centre (9) which extracts ObjectIDs from the MFT and reports the date, sequence, MAC address, associated filename and MFT record of each ObjectID. A sample output from this script is shown below –

```
-----Case: MyTestCase-----
28/Sep/09 07:03:29 Seq: 1944 Mac: 00-50-56-C0-00-08 (Entry: MyTestCase\MyDriveImage\ComputerTriage.doc FileID: 38)
28/Sep/09 07:03:29 Seq: 1944 Mac: 00-50-56-C0-00-08 (Entry: MyTestCase\MyDriveImage\ComputerTriage.docx FileID: 39)
28/Sep/09 07:03:29 Seq: 1944 Mac: 00-50-56-C0-00-08 (Entry: MyTestCase\MyDriveImage\12345.bmp FileID: 40)
28/Sep/09 07:03:29 Seq: 1944 Mac: 00-50-56-C0-00-08 (Entry: MyTestCase\MyDriveImage\123456.bmp FileID: 41)
28/Sep/09 07:03:29 Seq: 1944 Mac: 00-50-56-C0-00-08 (Entry: MyTestCase\MyDriveImage\1234567.bmp FileID: 35)
28/Sep/09 07:03:29 Seq: 1944 Mac: 00-50-56-C0-00-08 (Entry: MyTestCase\MyDriveImage\Scan 004.jpg FileID: 36)
28/Sep/09 07:03:29 Seq: 1944 Mac: 00-50-56-C0-00-08 (Entry: MyTestCase\MyDriveImage\screenshot.bmp FileID: 37)
-----
```

Perhaps you have a loose hard drive and you want to identify which computer it was installed in.

The presence of MAC addresses in ObjectIDs can be a useful forensic artefact in cases where it is necessary to identify the originating computer of loose hard drives, or identify periods when a hard drive has been moved between computers.

If the file system is NTFS then the ObjectID in the link files will contain a MAC address from which you may be able to identify the computer that the hard drive was connected to.

If you suspect the hard drive was moved between computers, again you might be able to identify the periods during which it was in different computers from the MAC addresses in the ObjectIDs.

It is interesting to note that the MAC address in the ObjectID will generally be that of the primary network card; however other MAC addresses have been observed. For example when a mobile phone (Windows Mobile OS) is connected to the host computer via Active Sync the MAC address of the phone has been observed to be used in the creation of the ObjectIDs rather than that of the

host's network adaptor. In the case of a Windows Mobile device the MAC address is one provided by the Active Sync software and it will commonly be 80:00:60:0F:E8:00 . The MAC address of a VMware Virtual Ethernet Adapter has also been found in ObjectIDs e.g. 00:50:56:C0:00:08 (00:50:56 is one of several Organizationally Unique Identifiers registered to VMware which can be found in the public listings(10)). So finding a MAC address other than that of the host computer does not necessarily place the hard drive in another computer.

Perhaps you suspect the computer clock has been tampered with.

In pre-Vista Windows OS the sequence numbers within the ObjectIDs will enable you to put the link files in chronological order and if the dates and times are anomalous it should become apparent.

In Windows XP the UuidSequenceNumber is incremented at system boot so in a case where the system clock has been altered and moved backwards and forwards the sequence value might reveal that, and indicate the order in which some files were originally accessed.

The following screenshot shows part of the information carved from a number of link files. It shows the RemainingPathName, FileObjectID, and the Date and Sequence decoded from the ObjectID. It can be seen that the sequence numbers are in corresponding order with the dates.

RemainingPathName	FileObjectID	Date	MAC	Sequence
Program Files\Sophos\Sophos Anti-Virus\Sophos Anti-Virus.URL	5098FA1931BADC11B5AA0003FF2E1821	03/01/2008 19:21	00:03:FF:2E:18:21	13738
Program Files\Sophos\Sophos Anti-Virus\SavMain.exe	5198FA1931BADC11B5AA0003FF2E1821	03/01/2008 19:21	00:03:FF:2E:18:21	13738
Program Files\Sophos\AutoUpdate\ALMon.exe	5298FA1931BADC11B5AA0003FF2E1821	03/01/2008 19:21	00:03:FF:2E:18:21	13738
Program Files\MSN Messenger\msnmsgr.exe	717E47F742BADC11B5AB0003FF2E1821	03/01/2008 21:29	00:03:FF:2E:18:21	13739
Program Files\Yahoo!\Messenger\logs\network_user_log	D1DC28C924FCDC11B5C30003FF2E1821	27/03/2008 17:39	00:03:FF:2E:18:21	13763
splashw.bmp	B0FC07638847DD11B5C90003FF2E1821	01/07/2008 16:11	00:03:FF:2E:18:21	13769
NAS_Configuration.html	B1FC07638847DD11B5C90003FF2E1821	01/07/2008 16:11	00:03:FF:2E:18:21	13769
Program Files\TrueCrypt\TrueCrypt.exe	300B1627DB4DDD11B5CB0003FF2E1821	09/07/2008 17:19	00:03:FF:2E:18:21	13771
Program Files\TrueCrypt\TrueCrypt Setup.exe	310B1627DB4DDD11B5CB0003FF2E1821	09/07/2008 17:19	00:03:FF:2E:18:21	13771
Documents and Settings\user\Application Data\TrueCrypt\Configuration.xml	340B1627DB4DDD11B5CB0003FF2E1821	09/07/2008 17:19	00:03:FF:2E:18:21	13771
Documents and Settings\user\Desktop\FTK Imager\FTK Imager.exe	C0178F90BE51DD11B5CD0003FF2E1821	14/07/2008 16:04	00:03:FF:2E:18:21	13773
Documents and Settings\user\My Documents\My Pictures\110.jpg	C2178F90BE51DD11B5CD0003FF2E1821	14/07/2008 16:04	00:03:FF:2E:18:21	13773
Documents and Settings\user\Desktop\FTK Imager\FTK Imager.exe	C0178F90BE51DD11B5CD0003FF2E1821	14/07/2008 16:04	00:03:FF:2E:18:21	13773
Documents and Settings\user\Desktop\Contents Table Recent3.txt	844A06F6627FDD11B5D30003FF2E1821	10/09/2008 18:04	00:03:FF:2E:18:21	13779

The decoded date represents the system date and time when the computer was booted at the start of that session in which the ObjectID was created, that date will remain the same in all ObjectIDs created during that boot session. In the same way the sequence value will remain the same in that session.

If the computer clock had at some time been changed to an earlier date the sequence value would increment when next rebooted but the date embedded in any ObjectID created would be out of synchronisation. This is shown in the example below.

RemainingPathName	FileObjectID	Date	MAC	Sequence
Program Files\Sophos\Sophos Anti-Virus\Sophos Anti-Virus.URL	5098FA1931BADC11B5AA0003FF2E1821	03/01/2008 19:21	00:03:FF:2E:18:21	13738
Program Files\Sophos\Sophos Anti-Virus\SavMain.exe	5198FA1931BADC11B5AA0003FF2E1821	03/01/2008 19:21	00:03:FF:2E:18:21	13738
Program Files\Sophos\AutoUpdate\ALMon.exe	5298FA1931BADC11B5AA0003FF2E1821	03/01/2008 19:21	00:03:FF:2E:18:21	13738
Program Files\MSN Messenger\msnmsgr.exe	717E47F742BADC11B5AB0003FF2E1821	03/01/2008 21:29	00:03:FF:2E:18:21	13739
Program Files\Yahoo!\Messenger\logs\network_user_log	D1DC28C924FCDC11B5C30003FF2E1821	27/03/2008 17:39	00:03:FF:2E:18:21	13763
splashw.bmp	B0FC07638847DD11B5C90003FF2E1821	01/07/2008 16:11	00:03:FF:2E:18:21	13769
NAS_Configuration.html	B1FC07638847DD11B5C90003FF2E1821	01/07/2008 16:11	00:03:FF:2E:18:21	13769
Program Files\TrueCrypt\TrueCrypt.exe	300B1627DB4DDD11B5CB0003FF2E1821	09/02/2008 17:19	00:03:FF:2E:18:21	13771
Program Files\TrueCrypt\TrueCrypt Setup.exe	310B1627DB4DDD11B5CB0003FF2E1821	09/02/2008 17:19	00:03:FF:2E:18:21	13771
Documents and Settings\user\Application Data\TrueCrypt\Configuration.xml	340B1627DB4DDD11B5CB0003FF2E1821	09/02/2008 17:19	00:03:FF:2E:18:21	13771
Documents and Settings\user\Desktop\FTK Imager\FTK Imager.exe	C0178F90BE51DD11B5CD0003FF2E1821	14/07/2008 16:04	00:03:FF:2E:18:21	13773
Documents and Settings\user\My Documents\My Pictures\110.jpg	C2178F90BE51DD11B5CD0003FF2E1821	14/07/2008 16:04	00:03:FF:2E:18:21	13773
Documents and Settings\user\Desktop\FTK Imager\FTK Imager.exe	C0178F90BE51DD11B5CD0003FF2E1821	14/07/2008 16:04	00:03:FF:2E:18:21	13773
Documents and Settings\user\Desktop\Contents Table Recent3.txt	844A06F6627FDD11B5D30003FF2E1821	10/09/2008 18:04	00:03:FF:2E:18:21	13779

It has been noted that in XP the sequence numbers often appear to be incremented by two. In relation to the Clock Sequence “The Remote Procedure Call” specification(11) states “The clock sequence value must be changed whenever the UUID generator detects that the local value of UTC has gone backward; this may be due to normal functioning of the DCE Time Service.” It has been observed that the action of the Windows Time Service setting the clock backwards to the correct time has caused the UuidSequenceNumber in the registry to be incremented by one. At the same time this change to the sequence number was not reflected in the ObjectIDs subsequently generated in that same boot session. When the computer was rebooted the UuidSequenceNumber was incremented again and then any ObjectIDs generated reflected the increment of two in the sequence value.

From observation in Vista systems the sequence numbers are randomised and so the same principle does not apply. This has been confirmed by Microsoft (12) “In Vista and later versions of the OS, due to some un-intended call pattern we are unable to read the UuidSequenceNumber that is stored in the registry. Thus on every boot we will create a new one. This is why the UuidSequenceNumber is not incremented on boot as happens on pre-Vista releases of the OS.”

Perhaps you want to try and work out the order in which some files were first accessed during a particular boot session.

An ObjectID is generated for a file when it is first opened in an application via the File/Open dialog or by double-clicking in Windows Explorer. Where an application is open and a file is created and then saved from the application an ObjectID is generally not generated. However this seems to be application specific, creating a txt in Notepad, a jpg and bmp from Paint, and a pdf from Word (2007) did not cause an ObjectID to be generated for the file. Creating a Word document in Word (2007) and an xlsx in Excel (2007) and a text document in Open Office (v2) caused an ObjectID to be generated.

In the example below the column headed “NewDate” shows the time and date decoded from the ObjectID to its left, the column headed “NewSequence” shows the Sequence value from the ObjectID. The fact these two values are the same indicates that the ObjectIDs were generated during the same boot session. Every new ObjectID generated is incremented by one from the previous value; this can be seen in the least significant bytes which are the first two bytes reading left to right. In order to make the incrementing of the ObjectID easier to recognise I have converted the two least significant bytes to decimal in the column headed “LastTwoBytes”.

RemainingPathName	LastTwoBytes	NewFileOID	NewDate	NewSequenc
Research\LinkFiles\test.jpg	48335	CFBC857E8BA6DE11B59E005056C00008	21/09/2009 08:48	13726
Research\LinkFiles\OID.txt	48336	D0BC857E8BA6DE11B59E005056C00008	21/09/2009 08:48	13726
Documents\NTFS.pdf	48337	D1BC857E8BA6DE11B59E005056C00008	21/09/2009 08:48	13726
Research\LinkFiles\SequenceNumbersInOrder.JPG	48496	70BD857E8BA6DE11B59E005056C00008	21/09/2009 08:48	13726
[MS-SHLLINK](2).pdf	48539	9BBD857E8BA6DE11B59E005056C00008	21/09/2009 08:48	13726
Research\LinkFiles\FileSize.PNG	48574	BE8D857E8BA6DE11B59E005056C00008	21/09/2009 08:48	13726
Research\LinkFiles\MyRecent.xls	48589	CDBD857E8BA6DE11B59E005056C00008	21/09/2009 08:48	13726
Research\LinkFiles\One.txt	48618	EABD857E8BA6DE11B59E005056C00008	21/09/2009 08:48	13726
Research\LinkFiles\Two.txt	48619	EB8D857E8BA6DE11B59E005056C00008	21/09/2009 08:48	13726
Research\LinkFiles\Three.txt	48620	ECBD857E8BA6DE11B59E005056C00008	21/09/2009 08:48	13726
Research\LinkFiles\Four.txt	48621	ED8D857E8BA6DE11B59E005056C00008	21/09/2009 08:48	13726

The last four files listed were all created in Notepad, saved, and then subsequently opened to generate the ObjectIDs. They were opened in the order indicated by their names and it can be seen that the ObjectIDs are all consecutive in the correct order.

You could identify when and how often the computer has been started up.

Carve all the link files from the machine and then decode the ObjectIDs. Identify those link files that have been created by the user on that machine. To do that examine the MAC addresses and the date and time decoded from the ObjectIDs; some of the link files will probably be program shortcuts that have been created by the software developer and will be nothing to do with the current user. Sort the list by the decoded date order and these are the times and dates the computer has been started. Bear in mind that a link file has not necessarily been created every time the computer has been used, so you will not be able to use the information to say a computer had not been started rather when it *had been* started.

A more comprehensive picture might be obtained by carving all the ObjectIDs from the MFT and decoding those.

You can identify if files have been moved across volumes

If a file is moved from one NTFS volume to another and then opened after it has been moved, the link file will contain both the ObjectID of the Birth Volume and ObjectID of the current Volume.

RemainingPathName	NewVolID	BirthVolID
Research\LinkFiles\test.jpg	54DE5CBCCE6F0E40A09E83AA957891B6	54DE5CBCCE6F0E40A09E83AA957891B6
Research\LinkFiles\testodt.odt	54DE5CBCCE6F0E40A09E83AA957891B6	8E8F6C5353B51B4EA302D1267AA36654
U3LauncherLogParser.zip	54DE5CBCCE6F0E40A09E83AA957891B6	54DE5CBCCE6F0E40A09E83AA957891B6

The above illustration shows the file testodt.odt has been moved from a volume with the ObjectID starting 8E8F.

Perhaps you have a link file and want to know the size of the target file

The file size is found in 4 bytes directly after the Created Accessed and Modified times. The value represents the file size at the time the file was last opened.

Type	Size	Value
signed integer	1-8	714,872
unsigned integer	1-8	714,872

000	4c 00 00 00 01 14 02 00-00 00 00 00 c0 00 00 00	I.....À...
010	00 00 00 46 9b 00 28 00-20 00 00 00 08 c6 02 71	...F..(.....E-g
020	9f 28 c9 01 60 0b 97 67-ee 3a ca 01 b0 c7 ad 67	.(É..`gi:É.°Ç-g
030	ee 3a ca 01 78 e8 0a 00-00 00 00 00 01 00 00 00	î:É:xE.....
040	00 00 00 00 00 00 00 00-00 00 00 00 a4 01 14 00x...

Just note here you might occasionally find a file size that is anomalous as the file size is stored as an unsigned integer in 4 bytes and where the file size is over 0xFFFFFFFF the 32 least significant bits of the file size are recorded.

For example a file of a size 15,841,593,856 bytes = 03 B0 3B 8A 00 will be recorded as B0 3B 8A 00 = 2,956,691,968 (which will be seen little endian in the link file as 00 8A 3B B0).

Perhaps you want to know if a file has been renamed and what the previous filename was

Where a file has been created and subsequently opened such that a link file has been created, if the file is then renamed and opened again with the new name a new link file will be created. However the target file itself remains the same file so its ObjectID which is embedded in both link files will be the same. By carving link files and looking for matching ObjectIDs it may be possible to find renamed files and identify the different names. By examining the dates and times of the link files and internal dates it may be possible to identify the order in which the file names existed.

The following is an example of two links files created in the way described above. The two different file names can be seen and both contain the same ObjectIDs.

Offset	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F	
00000000	4C	00	00	00	01	14	02	00	00	00	00	00	C0	00	00	00	L
00000010	00	00	00	46	9B	00	28	00	20	00	00	00	79	A7	6E	02	F
00000020	0C	25	CB	01	03	D9	7C	02	0C	25	CB	01	27	27	7D	02	(
00000030	0C	25	CB	01	04	00	00	00	00	00	00	00	01	00	00	00	{
00000040	00	00	00	00	00	00	00	00	00	00	00	00	6C	00	6A	00	%E
00000050	32	00	04	00	00	00	F0	3C	4C	8B	20	00	46	49	52	53	l
00000060	54	46	7E	31	2E	54	58	54	00	00	4E	00	07	00	04	00	j
00000070	EF	BE	F0	3C	4C	8B	F0	3C	4C	8B	26	00	00	00	7E	0B	<
00000080	03	00	00	00	60	00	00	00	00	00	00	00	00	00	00	00	L
00000090	46	00	69	00	72	00	73	00	74	00	20	00	46	00	69	00	
000000A0	6C	00	65	00	6E	00	61	00	6D	00	65	00	2E	00	74	00	F
000000B0	78	00	74	00	00	00	1C	00	00	00	5E	00	00	00	1C	00	i
000000C0	00	00	01	00	00	00	1C	00	00	00	33	00	00	00	00	00	%<
000000D0	00	00	5D	00	00	00	17	00	00	00	03	00	00	00	DC	6F	L&
000000E0	33	2A	10	00	00	00	46	65	72	67	75	73	00	43	3A	5C	~
000000F0	55	73	65	72	73	5C	61	69	74	63	68	5C	44	65	73	6B	^
00000100	74	6F	70	5C	46	69	72	73	74	20	46	69	6C	65	6E	61	F
00000110	6D	65	2E	74	78	74	00	00	29	00	2E	00	2E	00	5C	00	l
00000120	2E	00	2E	00	5C	00	2E	00	2E	00	5C	00	2E	00	2E	00	.
00000130	5C	00	2E	00	2E	00	5C	00	44	00	65	00	73	00	6B	00	.
00000140	74	00	6F	00	70	00	5C	00	46	00	69	00	72	00	73	00	.
00000150	74	00	20	00	46	00	69	00	6C	00	65	00	6E	00	61	00	.
00000160	6D	00	65	00	2E	00	74	00	78	00	74	00	16	00	43	00	.
00000170	3A	00	5C	00	55	00	73	00	65	00	72	00	73	00	5C	00	.
00000180	61	00	69	00	74	00	63	00	68	00	5C	00	44	00	65	00	.
00000190	73	00	6B	00	74	00	6F	00	70	00	60	00	00	00	03	00	.
000001A0	00	A0	58	00	00	00	00	00	00	00	66	65	72	67	75	73	.
000001B0	6D	61	63	6C	65	69	64	65	00	00	70	DA	1A	D4	B3	33	3
000001C0	AE	4D	80	1A	C2	91	D1	E3	A3	2B	2A	DF	0B	95	F2	90	0
000001D0	DF	11	8B	63	00	50	56	C0	00	08	70	DA	1A	D4	B3	33	0
000001E0	AE	4D	80	1A	C2	91	D1	E3	A3	2B	2A	DF	0B	95	F2	90	0
000001F0	DF	11	8B	63	00	50	56	C0	00	08	00	00	00	00	00	00	0

Offset	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F	
00000000	4C	00	00	00	01	14	02	00	00	00	00	00	C0	00	00	00	L
00000010	00	00	00	46	9B	00	28	00	20	00	00	00	79	A7	6E	02	F
00000020	0C	25	CB	01	03	D9	7C	02	0C	25	CB	01	27	27	7D	02	(
00000030	0C	25	CB	01	04	00	00	00	00	00	00	00	01	00	00	00	{
00000040	00	00	00	00	00	00	00	00	00	00	00	00	6C	00	6A	00	%E
00000050	32	00	04	00	00	00	F0	3C	4C	8B	20	00	53	45	43	4F	n
00000060	4E	44	7E	31	2E	54	58	54	00	00	50	00	07	00	04	00	l
00000070	EF	BE	F0	3C	4C	8B	F0	3C	4C	8B	26	00	00	00	7E	0B	<
00000080	03	00	00	00	60	00	00	00	00	00	00	00	00	00	00	00	L
00000090	53	00	65	00	63	00	6F	00	6E	00	64	00	20	00	46	00	
000000A0	69	00	6C	00	65	00	6E	00	61	00	6D	00	65	00	2E	00	F
000000B0	74	00	78	00	74	00	00	00	1C	00	00	00	5F	00	00	00	i
000000C0	1C	00	00	00	01	00	00	00	1C	00	00	00	33	00	00	00	%<
000000D0	00	00	00	00	5E	00	00	00	17	00	00	00	03	00	00	00	^
000000E0	DC	6F	33	2A	10	00	00	00	46	65	72	67	75	73	00	43	F
000000F0	3A	5C	55	73	65	72	73	5C	61	69	74	63	68	5C	44	65	.
00000100	73	6B	74	6F	70	5C	53	65	63	6F	6E	64	20	46	69	6C	.
00000110	65	6E	61	6D	65	2E	74	78	74	00	00	2A	00	2E	00	2E	*
00000120	00	5C	00	2E	00	2E	00	5C	00	2E	00	2E	00	5C	00	2E	.
00000130	00	2E	00	5C	00	2E	00	2E	00	5C	00	44	00	65	00	73	.
00000140	00	6B	00	74	00	6F	00	70	00	5C	00	53	00	65	00	63	.
00000150	00	6F	00	6E	00	64	00	20	00	46	00	69	00	6C	00	65	.
00000160	00	6E	00	61	00	6D	00	65	00	2E	00	74	00	78	00	74	.
00000170	00	16	00	43	00	3A	00	5C	00	55	00	73	00	65	00	72	.
00000180	00	73	00	5C	00	61	00	69	00	74	00	63	00	68	00	5C	.
00000190	00	44	00	65	00	73	00	6B	00	74	00	6F	00	70	00	60	.
000001A0	00	00	00	03	00	00	A0	58	00	00	00	00	00	00	00	00	6
000001B0	65	72	67	75	73	6D	61	63	6C	65	69	64	65	00	00	70	f
000001C0	DA	1A	D4	B3	33	AE	4D	80	1A	C2	91	D1	E3	A3	2B	2A	0
000001D0	DF	0B	95	F2	90	DF	11	8B	63	00	50	56	C0	00	08	70	0
000001E0	DA	1A	D4	B3	33	AE	4D	80	1A	C2	91	D1	E3	A3	2B	2A	0
000001F0	DF	0B	95	F2	90	DF	11	8B	63	00	50	56	C0	00	08	00	0
00000200	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	0

The details of the two link files above are as follows –

LinkFile Created	16/07/2010 17:50:01	16/07/2010 17:50:22
LinkFile Modified	16/07/2010 17:50:01	16/07/2010 17:50:22
LinkFile Accessed	16/07/2010 17:50:01	16/07/2010 17:50:22
Embedded Created	16/07/2010 17:26:22	16/07/2010 17:26:22
Embedded Modified	16/07/2010 17:26:22	16/07/2010 17:26:22
Embedded Accessed	16/07/2010 17:26:22	16/07/2010 17:26:22
FileLength	4	4
RelativePath	..\..\..\..\Desktop\First Filename.txt	..\..\..\..\Desktop\Second Filename.txt
Volume ObjectID	70DA1AD4B333AE4D801AC291D1E3A32B	70DA1AD4B333AE4D801AC291D1E3A32B
File ObjectID	2ADF0B95F290DF118B63005056C00008	2ADF0B95F290DF118B63005056C00008
GUID Date	16/07/2010 15:55:40	16/07/2010 15:55:40
GUID MAC	00:50:56:C0:00:08	00:50:56:C0:00:08
GUID Sequence	2915	2915

The link file Created, Modified and Accessed dates indicate which file name came first. The embedded dates are the same indicating the target file itself has not been modified.

It can be seen that the File ObjectIDs are identical (the Volume ObjectID would be the same for all files on that volume).

It may be possible to find MFT artefacts for the files by searching for the File ObjectID and so identify further information about the file.

The Short Version

So in summary -

If you have a Target File and a Link file you potentially have the following history, subject to the details discussed above,

The Created Date of the Link File – the date the Target File was first accessed.

The Link File internal dates which are a snapshot of the Target File’s three dates before it was last opened.

The Modified Date of the Link File which is the time the Target File was last opened.

The Modified Date of the Target File which is when the Target File was last changed.

Link Files for Target Files on NTFS volumes can contain the VolumeID of the volume and ObjectID of the Target File, and these can show if files have been moved between NTFS volumes. The ObjectIDs can also show the order of ObjectID creation and the MAC address of the host computer.

Finally, the observations described in this paper will only be certain in the context and at the time in which they were made. Experience shows that the dynamic nature of computer technology means the findings can only be a guide and that the observations must be tested in the environment under examination. A wise investigator will never just accept what he reads no matter from what source; the comment mentioned earlier that inaccurately states “NtfsDisableLastAccessUpdate is now 1,

which means no last access timestamp will be written at all” comes from the author of a book on computer forensics. I hope you will find this paper useful in providing a starting point for your own testing and observations.

If the information in this paper helps you in an investigation please let me know, I would be pleased to learn my efforts have been put to some practical use, as that was the purpose of writing it.

I can be contacted at digitalforensics@ my domain.

Bibliography

1. **Microsoft.** Win32 and Com Development Windows System Information - File Times. *MSDN Library*. [Online] <http://msdn.microsoft.com/en-us/library/ms724290.aspx>.
2. —. Windows 2008 Command Reference - fsutil. *Technet Library*. [Online] <http://technet.microsoft.com/en-us/library/cc753059.aspx>.
3. —. Windows Shell Reference. *MSDN Library*. [Online] [http://msdn.microsoft.com/en-us/library/bb762105\(VS.85\).aspx](http://msdn.microsoft.com/en-us/library/bb762105(VS.85).aspx).
4. —. *[MS-DLTM] Distributed Link Tracking: Central Manager Protocol Specification*.
5. **The Internet Engineering Task Force.** RFC 4122. [Online] <http://rfc.net/rfc4122.html>.
6. **International Telecommunication Union.** Information technology - Open Systems Interconnection - Procedures for the operation of OSI Registration Authorities: Generation and registration of Universally Unique Identifiers (UUIDs) and their use as ASN.1 Object Identifier components. [Online] <http://www.itu.int/ITU-T/studygroups/com17/oid.html>.
7. **Microsoft.** [MS-SHLLINK]: Shell Link (.LNK) Binary File Format. *MSDN*. [Online] [Cited: 21 Sept 2009.] <http://msdn.microsoft.com/en-us/library/dd871305%28PROT.10%29.aspx>.
8. **Sanderson Forensics.** Products/LinkAlyzer. [Online] [Cited: 21 Sept 2009.] <http://www.sandersonforensics.com>.
9. **Guidance Software.** Enscript Resource Centre. *Guidance Support Portal*. [Online] <https://support.guidancesoftware.com/forum/downloads.php?do=file&id=426>.
10. **Institute of Electrical and Electronics Engineers.** Organizationally Unique Identifier Assignments. [Online] [Cited: 28 Sept 2009.] <http://standards.ieee.org/regauth/oui/index.shtml>.
11. **The Open group.** CDE 1.1: Remote Procedure Call . [Online] [Cited: 28 September 2009.] <http://www.opengroup.org/onlinepubs/9629399/toc.htm>.
12. **Engineer, Microsoft.** *Personal email communication with Harry Parsonage*.