An Investigation into Timestamps in NTFS

E&CE | Department of Electronic and Computer Engineering

Nathaniel Patrick Teskey

BSc. Cybersecurity and IT Forensics

Introduction

NTFS has been the foundation of Windows since the launch of the outcomes of operations Windows NT. I carried out experiments into common filetypes such as .txt, .docx, .pptx, .pdf, and .png, by recording such as file creation, access, modify, rename, copy, move and delete on the timestamp attributes. Following a strict experimental setup was essential to retain the integrity and accuracy of my results, to ensure this I calculated cryptographic hashes of my evidence to show that it was not inadvertently modified.

I then took a dive into the world of “timestomping”, which is the anti-forensic technique of artificially modifying timestamps. I used both basic and more advanced methods to investigate such as NewFileTime and setMACE.

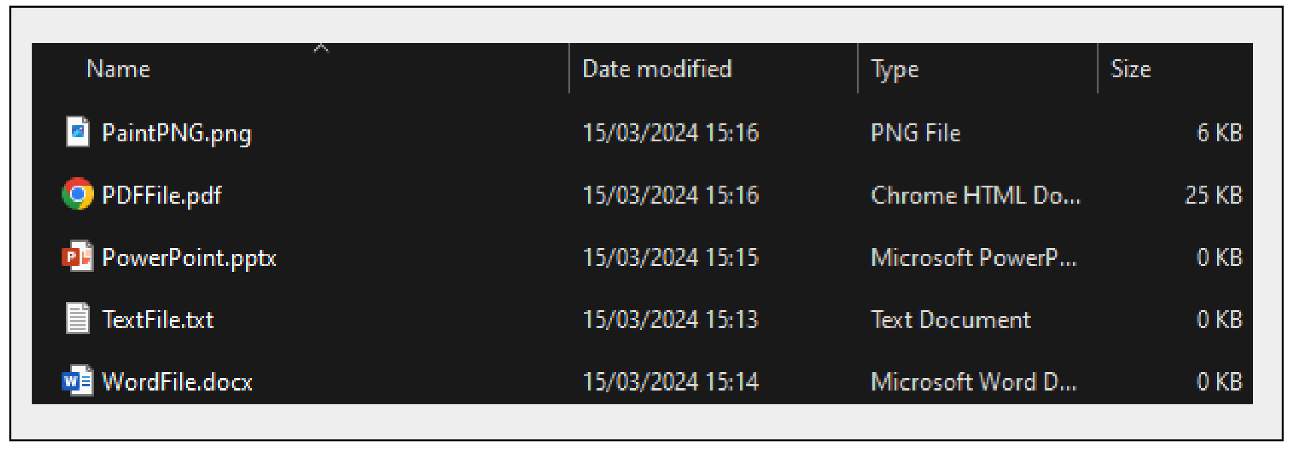
Aims

- Develop a solid understanding of timestamps within the NTFS filesystem.
- Investigate the behaviour of timestamps in relation to various file operations and filetypes.
- Utilize forensic tools to extract and analyse the evidence, adhering to best forensic practices.
- Examine the impact of both basic and advanced anti-forensic “timestomping” tools and evaluate their efficacy.
- Provide insight, based on my experiments, outlining the rules that impact timestamp behaviour on Windows 11.

Method

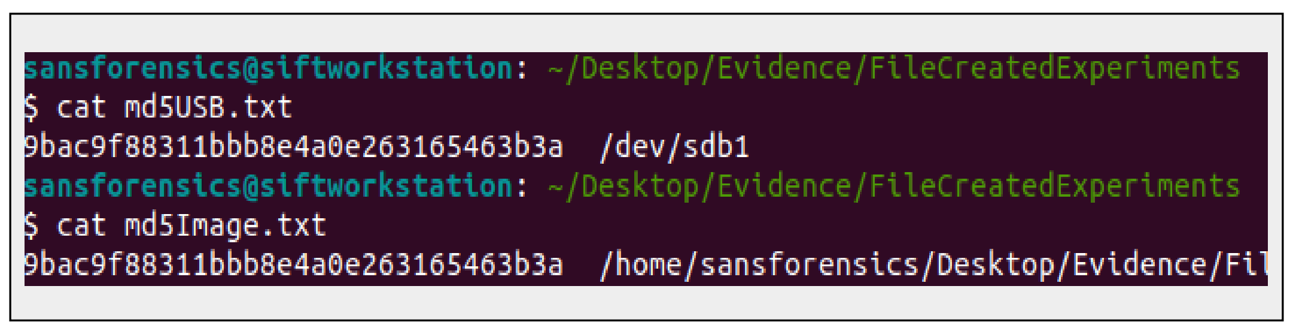
Experiments were set up on a USB drive inserted to a laptop running version 10.0.22631 of Windows 11, while analysis was carried out through a Linux Virtual Machine running SIFT Workstation.

- **Create test files** on USB drive using Windows file explorer or other application.



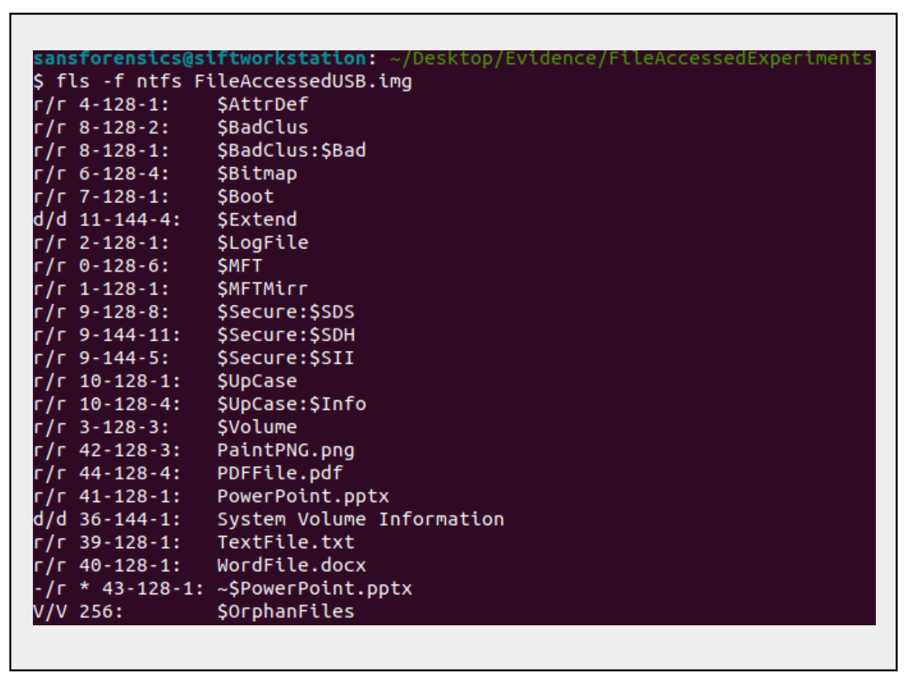
Name	Date modified	Type	Size
PaintPNG.png	15/03/2024 15:16	PNG File	6 KB
PDFFile.pdf	15/03/2024 15:16	Chrome HTML Do...	25 KB
PowerPoint.pptx	15/03/2024 15:15	Microsoft PowerP...	0 KB
TextFile.txt	15/03/2024 15:13	Text Document	0 KB
WordFile.docx	15/03/2024 15:14	Microsoft Word D...	0 KB

- Perform a permitted **file operation** or **timestomping** on one test file at a time.
- Launch **SIFT Workstation** virtual machine and **mount** USB drive.
- Calculate **MD5 hash of USB** drive.
- Take **raw image** of USB drive for analysis.
- Calculate **MD5 hash of raw image.**
- **Compare hashes** ensuring they are **identical**, proving evidence has not been modified during acquisition.



```
sansforensics@siftworkstation: ~/Desktop/Evidence/FileCreatedExperiments
$ cat md5USB.txt
9bac9f88311bbb8e4a0e263165463b3a  /dev/sdb1
sansforensics@siftworkstation: ~/Desktop/Evidence/FileCreatedExperiments
$ cat md5Image.txt
9bac9f88311bbb8e4a0e263165463b3a  /home/sansforensics/Desktop/Evidence/Fil
```

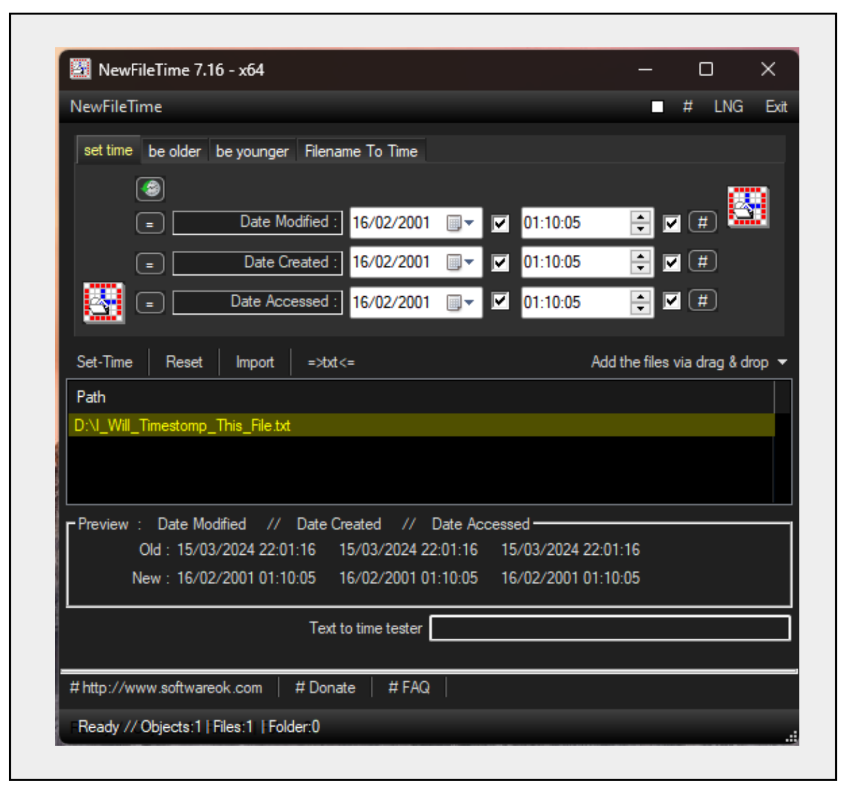
- Use `fls` command on raw image to obtain the **$MFT Entry address** of the test file.
- Using the `istat` command, specify the $MFT Entry address to **view** the test files **$MFT Entry**.
- Record the **changes** in the **timestamps** within the $STANDARD_INFORMATION and $FILE_NAME attributes.



```
sansforensics@siftworkstation: ~/Desktop/Evidence/FileAccessedExperiments
$ fls -f ntfs FileAccessedUSB.img
r/r 4-128-1:    $AttrDef
r/r 8-128-2:    $BadClus
r/r 8-128-1:    $BadClus:$Bad
r/r 6-128-4:    $Bitmap
r/r 7-128-1:    $Boot
d/d 11-144-4:   $Extend
r/r 2-128-1:    $LogFile
r/r 0-128-6:    $MFT
r/r 1-128-1:    $MFTMirr
r/r 9-128-8:    $Secure:$SDS
r/r 9-144-11:   $Secure:$SDH
r/r 9-144-5:    $Secure:$SII
r/r 10-128-1:   $UpCase
r/r 10-128-4:   $UpCase:$Info
r/r 3-128-3:    $Volume
r/r 42-128-3:   PaintPNG.png
r/r 44-128-4:   PDFFile.pdf
r/r 41-128-1:   PowerPoint.pptx
d/d 36-144-1:   System Volume Information
r/r 39-128-1:   TextFile.txt
r/r 40-128-1:   WordFile.docx
-/r * 43-128-1: ~$PowerPoint.pptx
V/V 256:        $OrphanFiles
```

Output of `fls` command showing TextFile.txt at $MFT Entry address 39.

Graphical user interface of a popular but basic anti-forensic timestomping tool, known as NewFileTime.



Date Modified : 16/02/2001 ☑ 01:10:05 ☑ #

Date Created : 16/02/2001 ☑ 01:10:05 ☑ #

Date Accessed : 16/02/2001 ☑ 01:10:05 ☑ #

Path
D:\I_Will_Timestomp_This_File.txt

Preview : Date Modified // Date Created // Date Accessed
Old : 15/03/2024 22:01:16 15/03/2024 22:01:16 15/03/2024 22:01:16
New : 16/02/2001 01:10:05 16/02/2001 01:10:05 16/02/2001 01:10:05

Text to time tester

Ready // Objects:1 | Files:1 | Folder:0

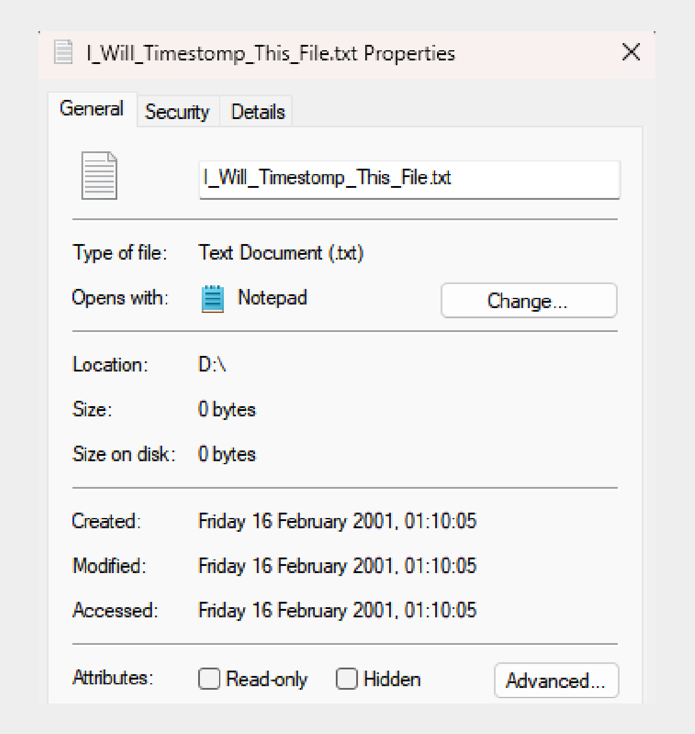
Results

Here I will explain one of my experiments results that I found to be quite interesting – see report for analysis of all results.

When carrying out experiments regarding timestomping with NewFileTime, the documentation provided with the application stated that it was only able to modify the $STANDARD_INFORMATION timestamp. This is in line with information available online, stating that the $FILE_NAME timestamp cannot be modified by anti-forensic tools as it is updated only by the Windows kernel. As you will see in my report – I have found this to be untrue, as it is possible to update the $FILE_NAME timestamps even using a basic method like NewFileTime.

Following the method outlined previously, I ran NewFileTime. It worked as expected and updated the $STANDARD_INFORMATION timestamps which are shown in Windows’ Properties view.

Windows 11 Properties view of the file that had its timestamps modified. It is highly suspicious as Created, Modified and Accessed timestamps all display the same time of 01:10:05 on 16/02/01.



I_Will_Timestomp_This_File.txt

Type of file: Text Document (.txt)

Opens with: Notepad

Location: D:\

Size: 0 bytes

Size on disk: 0 bytes

Created: Friday 16 February 2001, 01:10:05

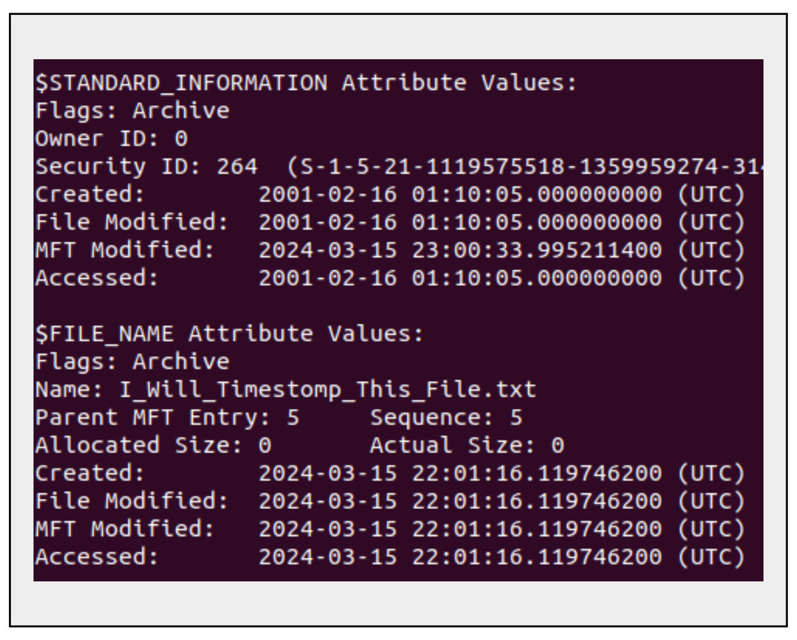
Modified: Friday 16 February 2001, 01:10:05

Accessed: Friday 16 February 2001, 01:10:05

Attributes: □ Read-only □ Hidden

A deeper look at $STANDARD_INFORMATION and $FILE_NAME timestamps using SIFT Workstation.

It is obvious that timestomping has occurred as $SI Created, File Modified and Accessed are all set to .000000000 ns. This is a clear sign of timestomping.



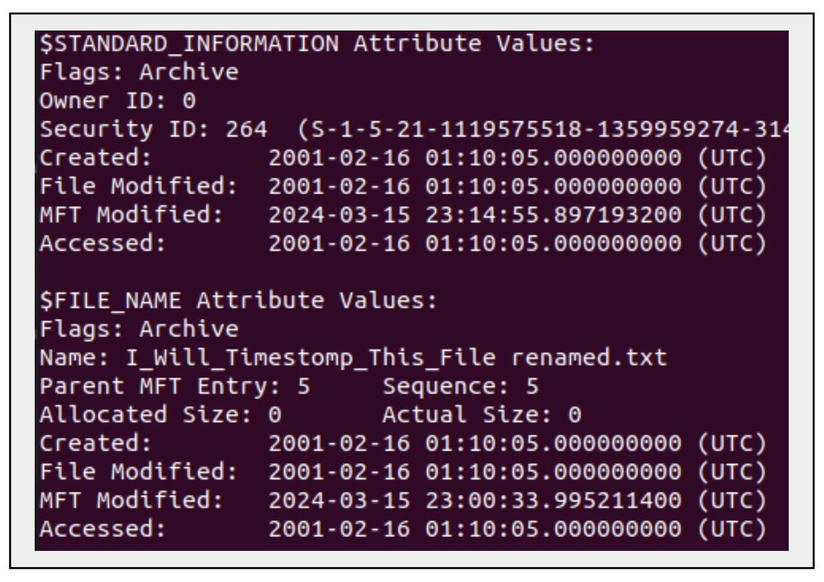
```
$STANDARD_INFORMATION Attribute Values:
Flags: Archive
Owner ID: 0
Security ID: 264  (S-1-5-21-1119575518-1359959274-31
Created:        2001-02-16 01:10:05.000000000 (UTC)
File Modified:  2001-02-16 01:10:05.000000000 (UTC)
MFT Modified:   2024-03-15 23:00:33.995211400 (UTC)
Accessed:       2001-02-16 01:10:05.000000000 (UTC)

$FILE_NAME Attribute Values:
Flags: Archive
Name: I_Will_Timestomp_This_File.txt
Parent MFT Entry: 5     Sequence: 5
Allocated Size: 0       Actual Size: 0
Created:        2024-03-15 22:01:16.119746200 (UTC)
File Modified:  2024-03-15 22:01:16.119746200 (UTC)
MFT Modified:   2024-03-15 22:01:16.119746200 (UTC)
Accessed:       2024-03-15 22:01:16.119746200 (UTC)
```

Notice how $SI MFT Modified was set to the time that the timestomping process was carried out – however as this is not displayed in Windows properties view it does not matter to end users.

Interestingly, using this basic timestomping method first, then renaming the file, I managed to disprove the myth that anti – forensic tools cannot alter the $FN timestamps.

Renaming the file caused the replication of the timestomped $SI timestamps to carry over to the $FN timestamps. However, $FN MFT Modified now displays the original file creation time.



```
$STANDARD_INFORMATION Attribute Values:
Flags: Archive
Owner ID: 0
Security ID: 264  (S-1-5-21-1119575518-1359959274-31
Created:        2001-02-16 01:10:05.000000000 (UTC)
File Modified:  2001-02-16 01:10:05.000000000 (UTC)
MFT Modified:   2024-03-15 23:14:55.897193200 (UTC)
Accessed:       2001-02-16 01:10:05.000000000 (UTC)

$FILE_NAME Attribute Values:
Flags: Archive
Name: I_Will_Timestomp_This_File renamed.txt
Parent MFT Entry: 5     Sequence: 5
Allocated Size: 0       Actual Size: 0
Created:        2001-02-16 01:10:05.000000000 (UTC)
File Modified:  2001-02-16 01:10:05.000000000 (UTC)
MFT Modified:   2024-03-15 23:00:33.995211400 (UTC)
Accessed:       2001-02-16 01:10:05.000000000 (UTC)
```

Conclusion and Reflection

I really enjoyed undertaking this investigation into NTFS timestamps as it was a challenging topic that I knew very little about this time last year! I had to do background research into NTFS, then learn how to use tools such as The Sleuth Kit and SIFT Workstation and create a plan of file operations, file types, anti – forensic tools and correct forensic procedures to undertake insightful experiments.

In hindsight, I would have used a physical write blocker to further preserve the integrity of the evidence on the USB drive – however it is not absolutely necessary as the MD5 hashes were always a match.

In my report I have explained the results of all the experiments that I undertook. I also gave my own opinion on the reasons for timestamps being updated in a specific way on Windows.

Acknowledgements

I want to extend my sincere thanks to my supervisor Dr. Jacqueline Walker for her valuable advice and encouragement. I was delighted to see that this project had been provided as an FYP option and for that I am extremely grateful.

UNIVERSITY OF LIMERICK
OLLSCOIL LUIMNIGH