Lab Report

Name: Inor Wang

Title: Microsoft Link File and Jumplist Analysis

Case: 25-T102

Date: 09/19/2025

Table of Contents

Document Revision History **4**

Executive Summary **5**

Synopsis **7**

Evidence Analyzed **10**

Tools Used **11**

Workstation 11

Software 11

Analysis Findings **12**

Overview of Examination Procedures 12

Evidence Reviewed 12

Key Findings 13

Section 1 – Link Files 13

--- Case 1 --- 13

1. As an investigator, you are interested in a file named “HW02File01.docx” – the original (“target”) file. Report the MAC d/t stamps for this file. 13

2. Where would you normally go to find the link file corresponding to HW02File01.docx? 14

3. This time, the link file has been provided for you. Report the MAC d/t stamps for this link file. 15

4. Report the embedded MAC d/t stamps of the target file within the link file. 16

5. The user downloaded the file but says they never opened it. Does the evidence support or refute this claim? Explain. 18

--- Case 2 --- 20

1. This time, you are interested in a file named “HW02File02.docx”. Report the MAC d/t stamps for this file. 20

2. Report the MAC d/t stamps for the corresponding link file. 21

3. Report the embedded MAC d/t stamps of the target file within the link file. 22

4. For each of the following activities, report if the activity occurred. If so, report when it occurred, and how you know. (If it cannot be determined, indicate as much.): a. Target File created – b. Target File last opened – c. Target File last saved – d. Target File last closed 24

5. Why is the target file’s modified time, the link file’s modified time, and the embedded modified time of the target file within the link file inconsistent (different from each other)? 26

--- General Questions --- 28

1. During your investigation, you find a target file whose creation time comes after the creation time of its associated link file. Explain what happened (assume the user did not deliberately changed the d/t stamps). 28

2. When a target file is moved to another directory, what happens to the associated link file? The target file is then opened from its new directory and immediately closed; what happens to the link file, now? Report the steps you took or the references you consulted to answer these questions. 30

Section 2 – Jumplist 33

--- Preparation --- 33

1. Create a word document with the filename “Evidence01.docx” and some content, and save it anywhere you wish. 33

2. Take a screenshot of the recently opened jumplist shown for Word in the Start Menu. 33

3. Delete the file you just made. If the file was sent to the Recycle Bin, go to the Recycle Bin and delete it permanently. 34

--- Analysis of Jumplists --- 35

1. Take a screenshot of the recently opened jumplist shown for Word in the Start Menu again. Is Evidence01.docx still there? 35

2. Find the jumplist files in the AutomaticDestinations folder using either CMD or File Explorer. Take a screenshot of these files. 36

3. To parse these jumplists, we'll first use NirSoft's JumpListsView tool. After starting the application, find Evidence01.docx in the list. What information do we learn about the file from this Tool? Take note of the corresponding Application ID – we will need it later. 37
4. Close JumpListsView. Next, we'll be using Eric Zimmerman's JumpLists Explorer tool. If you are having trouble running it, follow the "Requirements and Troubleshooting" instructions at the bottom of the webpage where you downloaded the tool. (You may need to install Microsoft .NET Framework 4.6) 38
5. Click File > Load jump lists. Navigate to the AutomaticDestinations folder and select the jumplist file with the AppID you noted in Step 3 of this section. You will see a list of entries – each corresponds to a different Word document recently opened on this machine. Find the entry for Evidence01.docx and double-click it. A new window opens with detailed information regarding the file. 39
6. Browse the information available in this window; there are multiple tabs. There is one artifact in particular that this tool provides us (and NirSoft's tool does not) which could prove highly useful in a forensic investigation of the Evidence01.docx file. Take a screenshot of this information and explain why it is so potentially useful. 40

Conclusion 41

References 42

Document Revision History

Name	Revision Date	Version	Description
Inor Wang	09/19/2025	0.1	Draft

Executive Summary

On September 19, 2025, the examiner, Inor Wang, conducted a forensic analysis of Microsoft Windows shortcut artifacts, specifically Link files (.lnk) and Jump Lists (.automaticDestinations-ms), as part of Case 25-T102. The examination was performed against the provided virtual disk image (link-file-capture.vmdk) to replicate industry-standard practices for artifact preservation, extraction, and interpretation. The **primary objective was to analyze these artifacts to reconstruct user activity, validate claims regarding file usage, and identify system-specific metadata capable of supporting attribution**.

The analysis followed forensic best practices. Evidentiary integrity was validated with MD5, SHA1, and SHA256 cryptographic hashing. Tools employed included FTK Imager (for mounting and hex review), NirSoft's JumpListsView (for basic Jump List parsing), and Eric Zimmerman's tools LECmd.exe (for Link file analysis) and JumpList Explorer (for advanced Jump List parsing). These utilities enabled the examiner to correlate metadata across files, validate timeline consistency, and uncover unique identifiers critical to system attribution.

Key findings from the examination are as follows:

- **Link File MAC Times**: Extracted Modified, Accessed, and Created (MAC) timestamps from target files and their associated .lnk shortcuts. These values provided evidence of when files were created, opened, and saved.
- **File Recreation Evidence**: Recreated a case where target file creation times postdated shortcut creation times, indicating deletion and later recreation of the target file, a normal artifact of operating system behavior rather than timestamp tampering.
- **User Activity Verification**: Demonstrated that the existence of a .lnk file provides strong evidence that a file was opened, refuting user claims of "downloaded but never opened."
- **Embedded Metadata**: Confirmed that .lnk files contain cached target timestamps that may not update with subsequent file modifications, explaining inconsistencies across target, link, and embedded metadata.
- **Jump List Persistence**: Established that Jump List artifacts retain references to files even after those files have been deleted from the file system, underscoring their value for reconstructing past user activity.

- **Enhanced Attribution**: Identified the network interface card MAC address within Jump List Explorer, a unique artifact not available in NirSoft's tool. This globally unique identifier directly ties file access activity to a specific system.
- **Corroboration Across Tools**: By cross-referencing Link file timestamps, embedded target metadata, and Jump List records, the examiner successfully reconstructed a reliable narrative of file access activity within the test environment.

In conclusion, **link files and Jump Lists are highly reliable forensic artifacts**. They provide evidence of user activity, support detailed timeline reconstruction, and enable attribution to a specific host system, even in cases where the original files have been deleted or recreated. This lab was valuable for building digital forensics skills, giving the examiner hands-on experience with Link files and Jump Lists. It highlighted how these artifacts preserve user activity, support attribution, and remain useful even after file deletion, key lessons for a student learning practical forensic investigation.

Synopsis

A set of Windows shortcut artifacts (Link files and Jumplists) was provided for offline analysis to answer defined investigative questions about a seized virtual machine environment. These artifacts are key sources of evidence regarding historical user activity on the system, including which documents and applications were executed, when they were accessed, and in some cases, how they were used. The client requested a step-by-step, reproducible workflow with annotated screenshots and explicit timestamps supporting each finding.

Client Questions:

Section 1 – Link Files

- Case 1
    1. As an investigator, you are interested in a file named "HW02File01.docx" – the original ("target") file. Report the MAC d/t stamps for this file.
    2. Where would you normally go to find the link file corresponding to HW02File01.docx?
    3. This time, the link file has been provided for you. Report the MAC d/t stamps for this link file.
    4. Report the embedded MAC d/t stamps of the target file within the link file.
    5. The user downloaded the file but says they never opened it. Does the evidence support or refute this claim? Explain.
- Case 2
    1. This time, you are interested in a file named "HW02File02.docx". Report the MAC d/t stamps for this file.
    2. Report the MAC d/t stamps for the corresponding link file.
    3. Report the embedded MAC d/t stamps of the target file within the link file.
    4. For each of the following activites, report if the activity occurred. If so, report when it occurred, and how you know. (If it cannot be determined, indicate as much.)
        a. Target File created.
        b. Target File last opened.

c. Target File last saved.

d. Target File last closed.

5. Why is the target file's modified time, the link file's modified time, and the embedded modified time of the target file within the link file inconsistent (different from each other)?

- General Questions

1. During your investigation, you find a target file whose creation time comes after the creation time of its associated link file. Explain what happened (assume the user did not deliberately change the d/t stamps).

2. When a target file is moved to another directory, what happens to the associated link file? The target file is then opened from its new directory and immediately closed; what happens to the link file, now? Report the steps you took or the references you consulted to answer these questions.

Section 2 – Jumplist

- Preparation

1. Create a word document with the filename "Evidence01.docx" and some content, and save it anywhere you wish.

2. Take a screenshot of the recently opened jumplist shown for Word in the Start Menu.

3. Delete the file you just made. If the file was sent to the Recycle Bin, go to the Recycle Bin and delete it permanently.

- Analysis of Jumplists

1. Take a screenshot of the recently opened jumplist shown for Word in the Start Menu again. Is Evidence01.docx still there?

2. Find the jumplist files in the Automatic Destinations folder using either CMD or File Explorer. Take a screenshot of these files.

3. To parse these jumplists, we'll first use NirSoft's JumpListsView tool. After starting the application, find Evidence01.docx in the list. What information do we learn about the file from this tool? Take note of the corresponding Application ID – we will need it later.

4. Close JumpListsView. Next, we'll be using Eric Zimmerman's JumpLists Explorer tool. If you are having trouble running it, follow the "Requirements and Troubleshooting" instructions at the bottom of the webpage where you downloaded the tool. (You may need to install Microsoft .NET Framework 4.6)

5. Click File > Load jump lists. Navigate to the AutomaticDestinations folder and select the jumplist file with the AppID you noted in Step 3 of this section. You

will see a list of entries – each corresponds to a different Word document recently opened on this machine. Find the entry for Evidence01.docx and double-click it. A new window opens with detailed information regarding the file.

6. Browse the information available in this window; there are multiple tabs. There is one artifact in particular that this tool provides us (and NirSoft's tool does not) which could prove highly useful in a forensic investigation of the Evidence01.docx file. Take a screenshot of this information and explain why it is so potentially useful.

Scope of Work:

- Acquisition of the forensic disk image ("link-file-capture.vmdk.zip") provided through course materials.
- Verification of evidentiary integrity using MD5, SHA1, and SHA256 cryptographic hashes.

Evidence Analyzed

This section provides details of the digital evidence collected

Evidence ID	**E001**
Name	link-file-capture.vmdk.zip
Type	Zip archive data, at least v2.0 to extract, compression method=deflate
Size	219,188 bytes (0.21 MB)
MD5	0469F7919C13A4DC91473A24722E0B4D
SHA1	7B3A04DEBED7AD8F86F0177A6D7B63628EF87748
SHA256	82D35989E641367DC13FE7A11B6E0F7C521A79123330EBDC13CEC24AD50EF1E5

Tools Used

Workstation

Hostname	Operating System	Build	Physical / Virtual	Built
IS-4523-001-WINDOWS	Windows 11	2021	Virtual	09/06/2025

Software

Name	Version	Release	Purpose
FTK Imager (AccessData)	4.5.0.3	Sep 2020	Mounts the .VMDK file, enabling extraction and analysis of link files
LECmd.exe (Eric Zimmerman)	1.5.1.0	Jan 2025	Analyzes link files
JumpListView (NirSoft)	1.16	2018	Parses jumplist files to reveal recent document access history
JumpLists Explorer (Eric Zimmerman)	2.1.0	Jun 2025	Provides detailed jumplist metadata, including unique forensic artifacts

Analysis Findings

Overview of Examination Procedures

The forensic analysis was conducted on the provided virtual disk image (*link-file-capture.vmdk*), which contained Windows link files (.lnk) and Jump List artifacts that were central to the examination. This disk image served as a controlled environment to simulate typical user activity, such as the creation, modification, and access of Microsoft Word documents, along with the automatic generation of associated shortcut and Jump List entries by the operating system. The evidence was supplied by Professor Stauffer in a compressed archive. Upon receipt, the examiner extracted the contents and immediately verified the integrity of the disk image using cryptographic hashing algorithms (MD5, SHA1, and SHA256). This ensured that the acquired evidence matched its original source and had not been altered during transfer. The use of multiple hashing methods strengthened the validation process by providing redundancy against potential algorithmic collisions. After validation, the examiner prepared the environment by mounting the virtual disk image in forensic tools to allow for read-only analysis. This step was critical to maintain forensic soundness and prevent accidental modification of the evidence. From there, targeted analysis was carried out on the link files and Jump List data, which provided insight into file access patterns, timestamps, and unique identifiers embedded within the artifacts. These procedures established a solid foundation for the deeper analysis that followed.

Additional targeted analysis was performed using:

- **FTK Imager (AccessData)** → mounted the .vmdk file, allowing the examiner to extract and analyze link files in their native environment.
- **LECmd.exe (Eric Zimmerman)** → parsed individual link (.lnk) files to extract embedded timestamps, file paths, and metadata.
- **JumpListView (NirSoft)** → parses jumplist files to reveal recent document access history.
- **JumpListExplorer (Eric Zimmerman)** → provides detailed jumplist metadata, including unique forensic artifacts.

Throughout the process, all findings were documented, and cryptographic hash values were maintained for validation.

Evidence Reviewed

1. **link-file-capture.vmdk.zip (E001)**: Link file capture – virtual disk

Key Findings

Section 1 – Link Files

--- Case 1 ---

1. As an investigator, you are interested in a file named "HW02File01.docx" – the original ("target") file. Report the MAC d/t stamps for this file.

- **Analysis Performed:**
  - The link-file-capture.vmdk was analyzed through the AccessData FTK Imager application.
  - The examiner navigated to "HW02File01.docx", the original ("target") file to report the MAC d/t stamps for the file which is shown in Figure 1.
  - Path: *link-file-capture.vmdk/Partition 1 [2045MB]/LinkFileAnalysis [NTFS]/[root]/Case 1/HW02File01.docx*
- **Answer:**
  The date that the file named "HW02File01.docx" was modified was **1/31/2019 7:51:29 PM UTC**. The date that the file named "HW02File01.docx" was accessed was **1/31/2019 7:51:29 PM UTC.** The date that the file named "HW02File01.docx" was created was **1/31/2019 7:51:29 PM UTC.**

- **Supporting Evidence:**

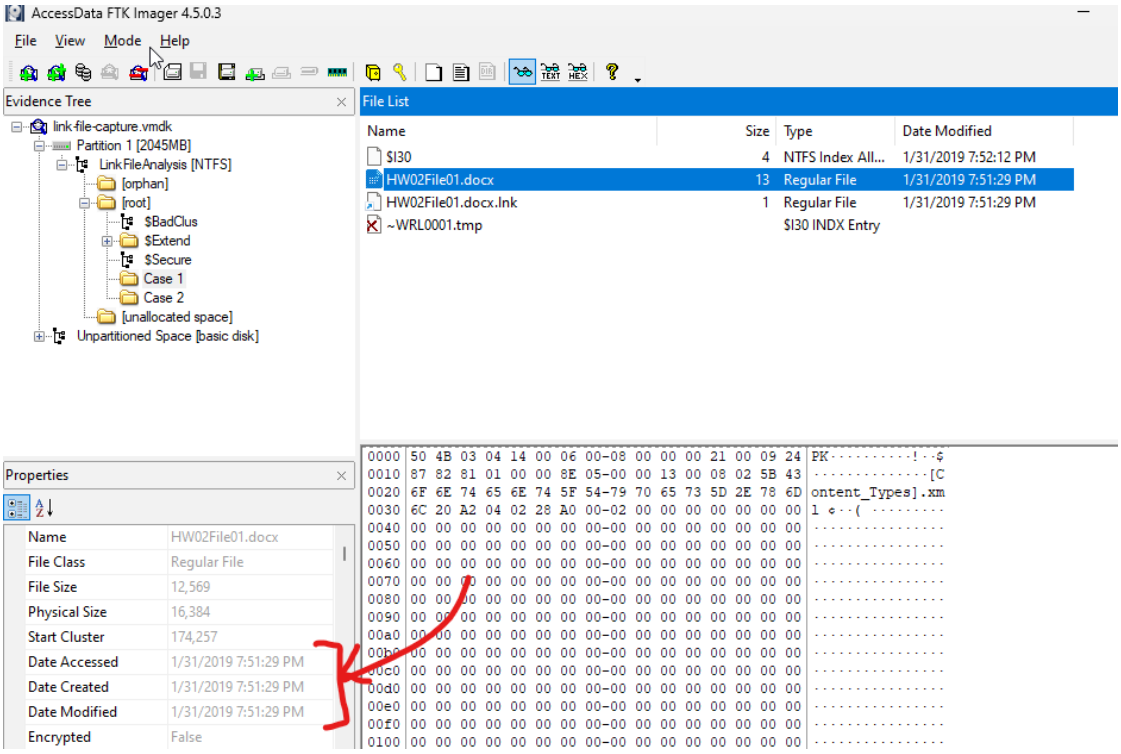

Figure 1. The MAC d/t stamps of "HW02File01.docx"

2. Where would you normally go to find the link file corresponding to HW02File01.docx?

- **Analysis Performed:**
  - The examiner used OSINT in order to find which directory the link files are in, in a Windows 11 machine.
  - As you can see in Figure 2, it shows the directory containing multiple link files.
  - Path: *C:\Users\inorw\AppData\Roaming\Microsoft\Windows\Recent*
- **Answer:**
  The absolute path of the directory that contains the link files, specifically where HW02File01.docx.lnk would normally be, is "***C:\Users\(yourname)\AppData\Roaming\Microsoft\Windows\Recent***".

- **Supporting Evidence:**

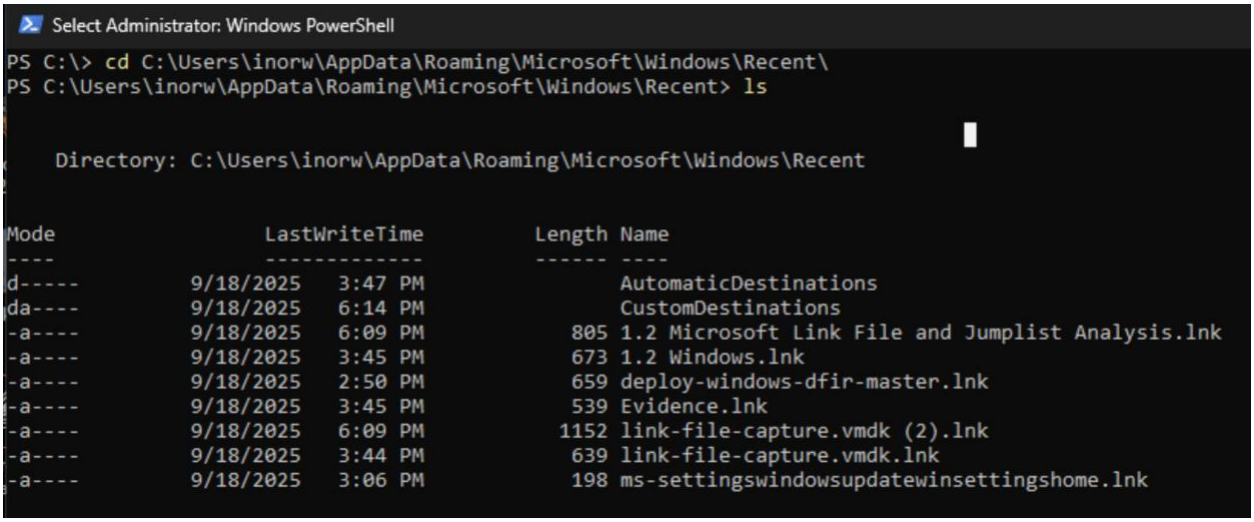

Figure 2. The directory that contains the link files for the Windows 11 system

3. This time, the link file has been provided for you. Report the MAC d/t stamps for this link file.

- **Analysis Performed:**
  - The link-file-capture.vmdk was analyzed through the AccessData FTK Imager application.
  - The examiner navigated to "HW02File01.docx.lnk", which was provided, to report the MAC d/t stamps for the file which is shown in Figure 3.
  - Path: *link-file-capture.vmdk/Partition 1 [2045MB]/LinkFileAnalysis [NTFS]/[root]/Case 1/HW02File01.docx.lnk*
- **Answer:**
  The date that the file named "HW02File01.docx.lnk" was modified was **1/31/2019 7:51:29 PM UTC**. The date that the file named "HW02File01.docx.lnk" was accessed was **1/31/2019 7:51:29 PM UTC.** The date that the file named "HW02File01.docx.lnk" was created was **1/31/2019 7:51:29 PM UTC.**

- **Supporting Evidence:**

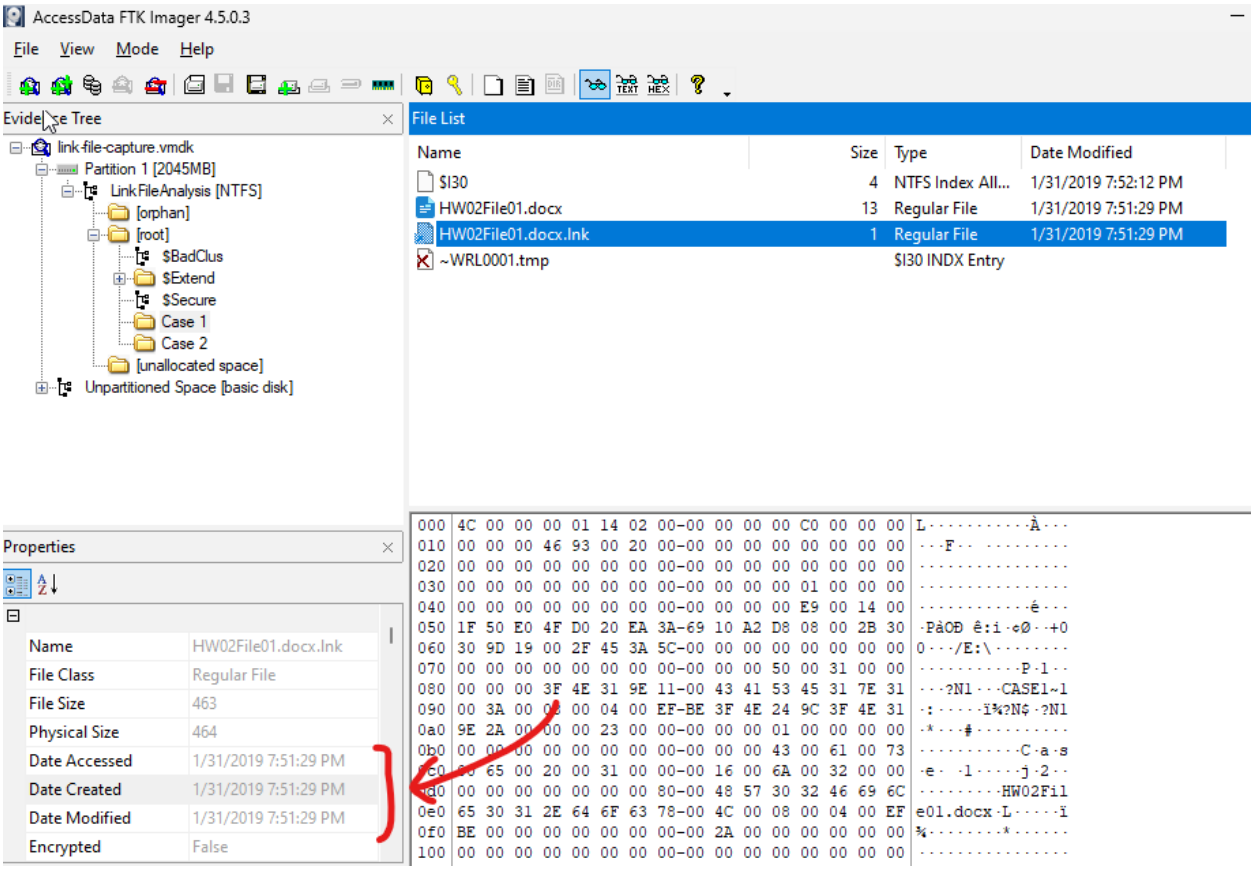

Figure 3. The MAC d/t stamps of "HW02File01.docx.lnk"

4. Report the embedded MAC d/t stamps of the target file within the link file.

- **Analysis Performed:**
  - The link-file-capture.vmdk was analyzed through the AccessData FTK Imager application.
  - The examiner navigated to "HW02File01.docx.lnk", which was provided, to report the embedded MAC d/t stamps for the target file within the link file which is shown in Figure 4.
  - The link file data structure states that starting from offset 28, 8B long is the embedded file creation time; from offset 36, 8B long is the embedded file access time; from offset 44, 8B long is the embedded file modified time.
  - The examiner mounted image file with READ-ONLY with FTK Imager to use Eric Zimmerman's LECmd.exe tool to learn more about the link file. After running the command on the .lnk file, it shows that the target MAC d/t stamps is "null" as shown in Figure 5.
  - Path: *link-file-capture.vmdk/Partition 1 [2045MB]/LinkFileAnalysis [NTFS]/[root]/Case 1/HW02File01.docx.lnk*
- **Answer:**
  The embedded MAC d/t stamps of the target file (HW02File01.docx) within the link file (HW02File01.docx.lnk) is **NULL since there are no values from offset 28 to 51** as shown in Figure 4, which is where the embedded MAC d/t stamps are located due to the link file data structure. This answer is later strengthened by using Eric Zimmerman's LECmd.exe tool, it shows that the target MAC d/t stamps is "**null**" as shown in Figure 5.

- **Supporting Evidence:**

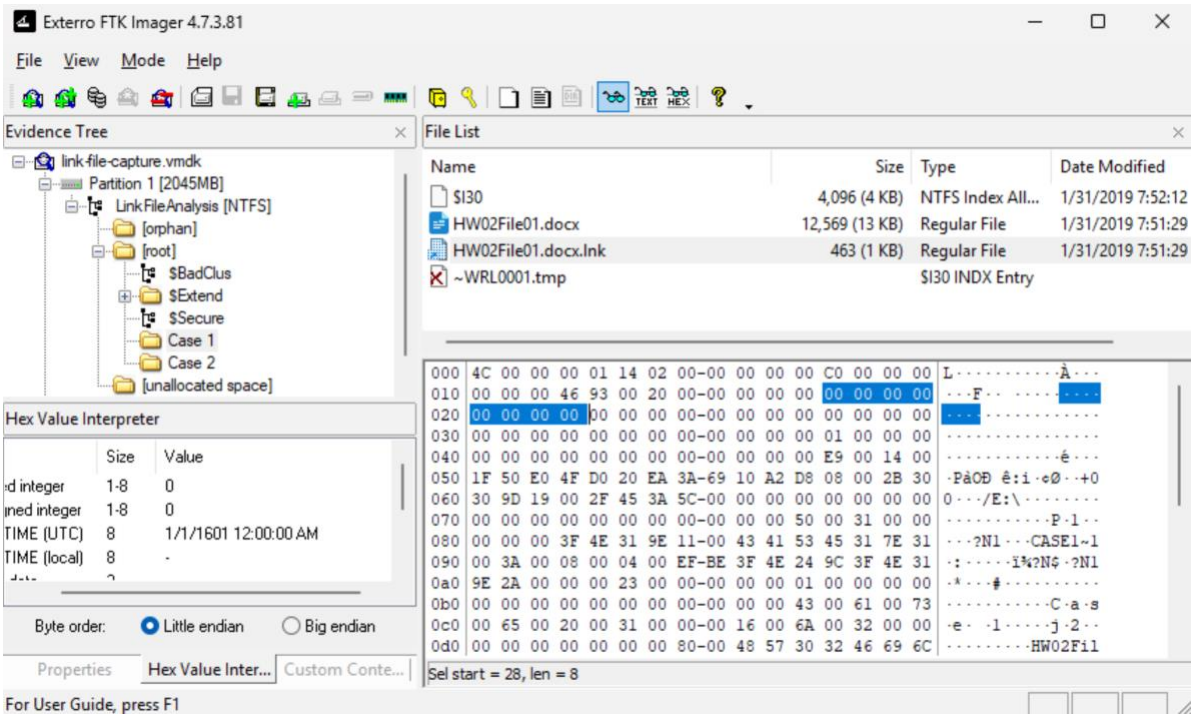

Figure 4. The FTK Imager showing the embedded MAC d/t of the target file within the link file

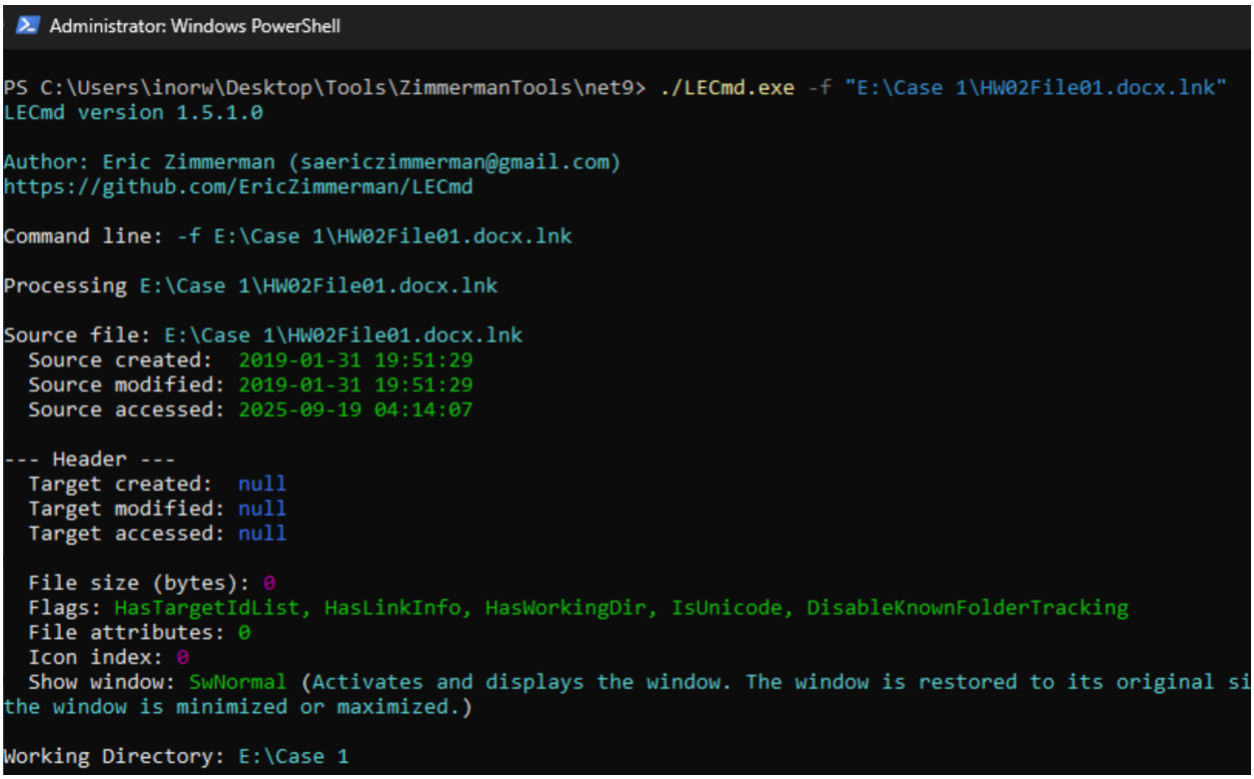

Figure 5. LECmd.exe command and output of HW02File01.docx.lnk showing the embedded MAC d/t of the target file

5. The user downloaded the file but says they never opened it. Does the evidence support or refute this claim? Explain.

- **Analysis Performed:**
  - The link-file-capture.vmdk was analyzed through the AccessData FTK Imager application.
  - The examiner used OSINT resources, more specifically an article by Mandiant, to understand more about why the embedded MAC d/t of the target file within the link file may be null.
  - Additionally, the examiner used OSINT resources to understand how .lnk files are created in the Windows system.
  - Path: *link-file-capture.vmdk/Partition 1 [2045MB]/LinkFileAnalysis [NTFS]/[root]/Case 1/HW02File01.docx*
  - Path: *link-file-capture.vmdk/Partition 1 [2045MB]/LinkFileAnalysis [NTFS]/[root]/Case 1/HW02File01.docx.lnk*
- **Answer:**
  A shortcut (HW02File01.docx.lnk) exists. **Windows creates/updates link files when a file is accessed from Explorer, so the presence of the .lnk file indicates that the document was <u>opened</u>**. This is also additionally confirmed by the examiner as he went to the VM's .lnk directory (Recent) and noticed that multiple directories had .lnk files as shown in Figure 9. The directories were not created by the examiner but only accessed, further corroborating that the **user opened the document.** Although the .lnk file's embedded target MAC fields are NULL, some .lnk variants especially Explorer "user search .lnks" legitimately omit the target timestamps. Therefore, the null embedded fields reflect missing metdata, not lack of access. Combined with the batching MAC times between the target file and the .lnk file, the evidence **<u>refuted the claim that the file was never opened.</u>**

- **Supporting Evidence:**

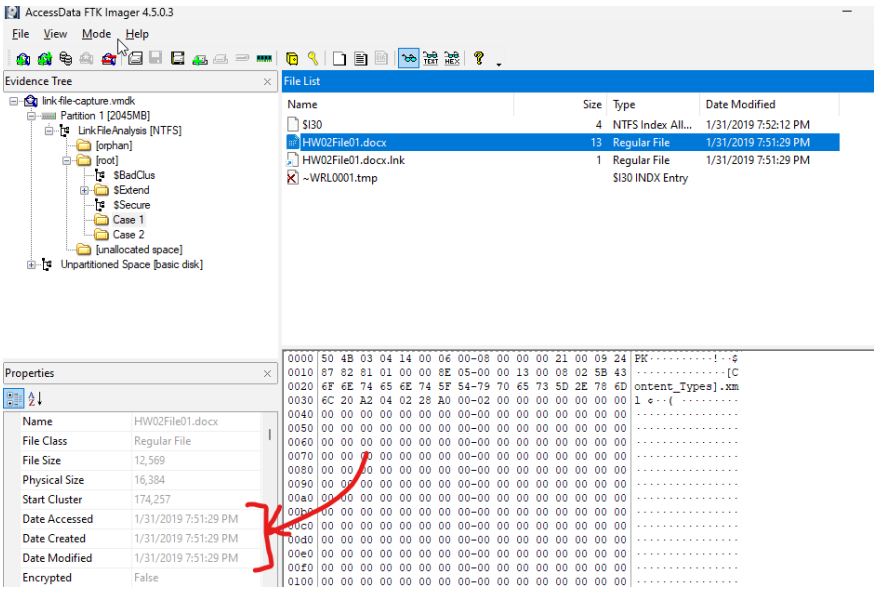

Figure 6. FTK Imager of the HW01File01.docx

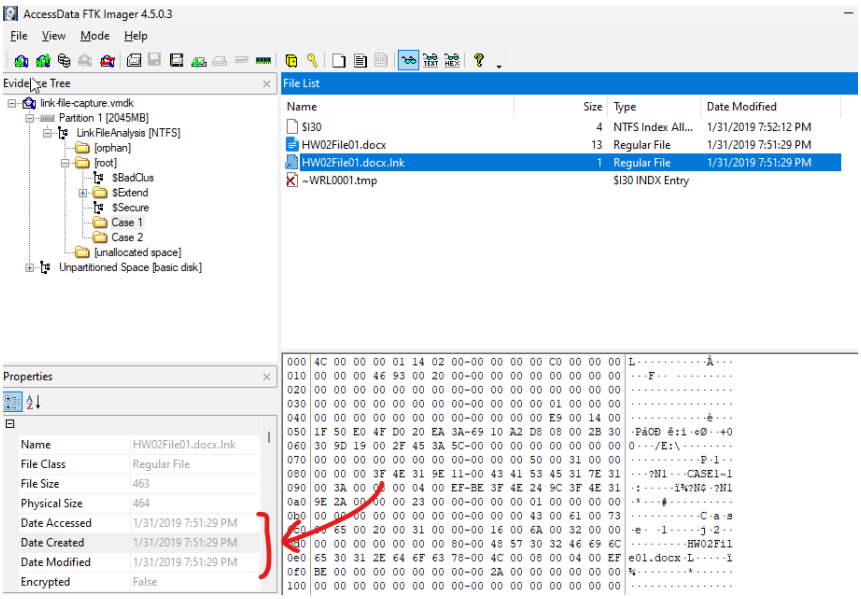

Figure 7. FTK Imager of HW01File01.docx.lnk

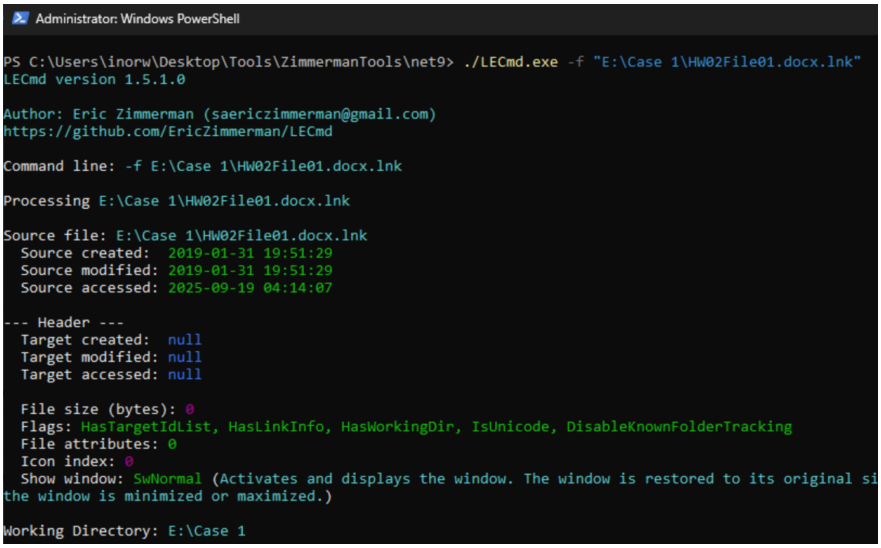

Figure 8. LECmd.exe command and output of HW01File01.docx.lnk showing the embedded MAC d/t stamps

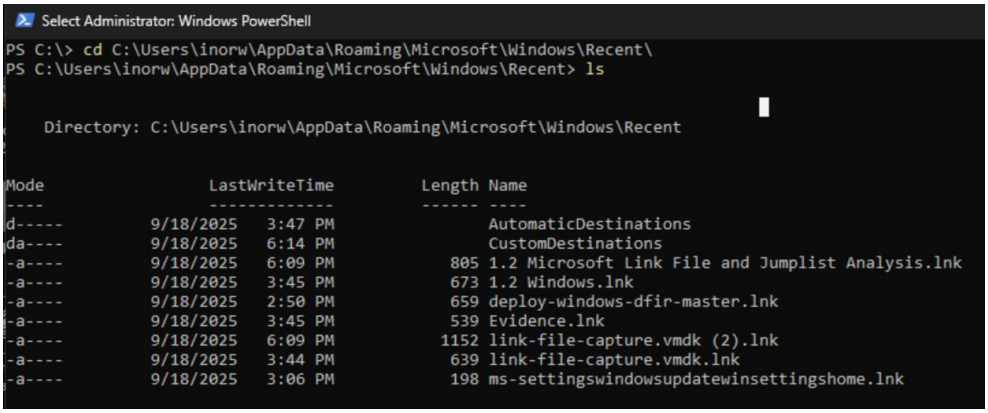

Figure 9. The directory containing the VM's .lnk files

--- Case 2 ---

1. This time, you are interested in a file named "HW02File02.docx". Report the MAC d/t stamps for this file.

- **Analysis Performed:**
  - The link-file-capture.vmdk was analyzed through the AccessData FTK Imager application.
  - The examiner navigated to "HW02File02.docx", the original ("target") file to report the MAC d/t stamps for the file which is shown in Figure 10.
  - Path: *link-file-capture.vmdk/Partition 1 [2045MB]/LinkFileAnalysis [NTFS]/[root]/Case 2/HW02File02.docx*
- **Answer:**
  The date that the file named "HW02File02.docx" was modified was **1/31/2019 8:09:23 PM UTC**. The date that the file named "HW02File02.docx" was accessed was **1/31/2019 8:09:23 PM UTC.** The date that the file named "HW02File01.docx.lnk" was created was **1/31/2019 8:03:10 PM UTC.**

- **Supporting Evidence:**

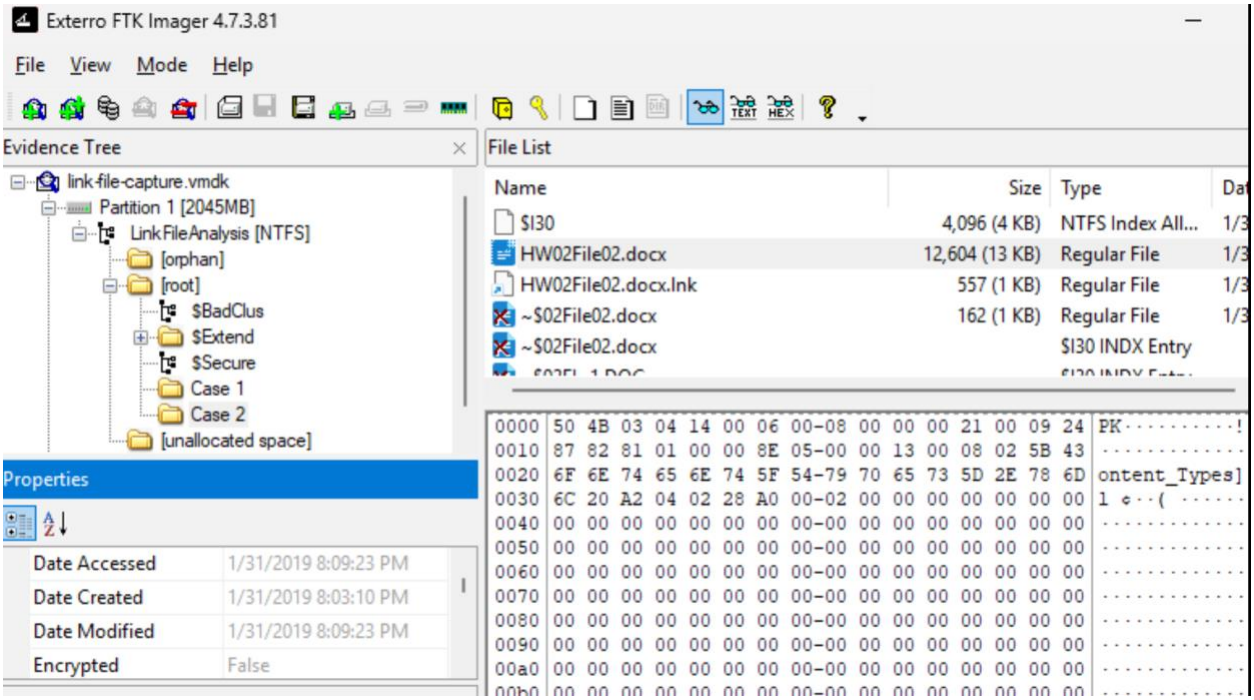

Figure 10. FTK Imager of the HW02File02.docx file

2. Report the MAC d/t stamps for the corresponding link file.

- **Analysis Performed:**
  - The link-file-capture.vmdk was analyzed through the AccessData FTK Imager application.
  - The examiner navigated to "HW02File02.docx.lnk", which was provided, to report the MAC d/t stamps for the file which is shown in Figure 11.
  - Path: *link-file-capture.vmdk/Partition 1 [2045MB]/LinkFileAnalysis [NTFS]/[root]/Case 2/HW02File02.docx.lnk*
- **Answer:**
  The date that the file named "HW02File02.docx.lnk" was modified was **1/31/2019 8:08:23 PM UTC**. The date that the file named "HW02File02.docx.lnk" was accessed was **1/31/2019 8:08:23 PM UTC.** The date that the file named "HW02File02.docx.lnk" was created was **1/31/2019 8:03:10 PM UTC.**

- **Supporting Evidence:**

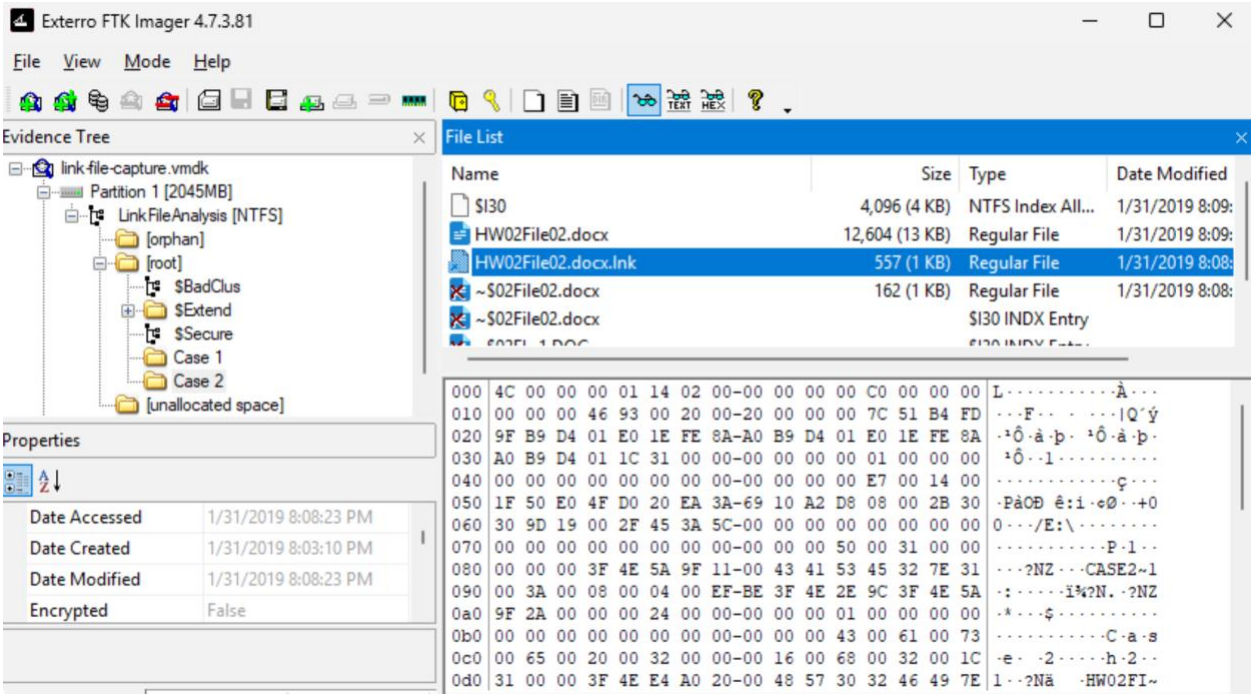

Figure 11. FTK Imager of the HW02File02.docx.lnk link file showing the MAC d/t timestamps

3. Report the embedded MAC d/t stamps of the target file within the link file.

- **Analysis Performed:**
  - The link-file-capture.vmdk was analyzed through the AccessData FTK Imager application.
  - Based off of the previous case, the examiner mounted image file with READ-ONLY to use Eric Zimmerman's LECmd.exe tool to learn more about the link file.
  - The examiner navigated to "HW02File02.docx.lnk", which was provided, to report the embedded MAC d/t stamps of the target file within the link file which is shown in Figures 12, 13, and 14.
  - In Figure 12, it shows the offset 28 with a length of 8B which shows the embedded file creation time.
  - In Figure 13, it shows the offset 36 with a length of 8B which shows the embedded file access time.
  - In Figure 14, it shows the offset 44 with a length of 8B which shows the embedded file modified time.
  - In Figure 15, it shows LECmd.exe command used and output further checking that the previous information was correct, which it is.
  - Path: *link-file-capture.vmdk/Partition 1 [2045MB]/LinkFileAnalysis [NTFS]/[root]/Case 2/HW02File02.docx.lnk*
- **Answer:**
  The embedded creation d/t of the target file within the link file is **1/31/2019 8:03:10 PM UTC.** The embedded access d/t of the target file within the link file is **1/31/2019 8:07:07 PM UTC.** The embedded modified d/t of the target file within the link file is **1/31/2019 8:07:07 PM UTC.**

- **Supporting Evidence:**

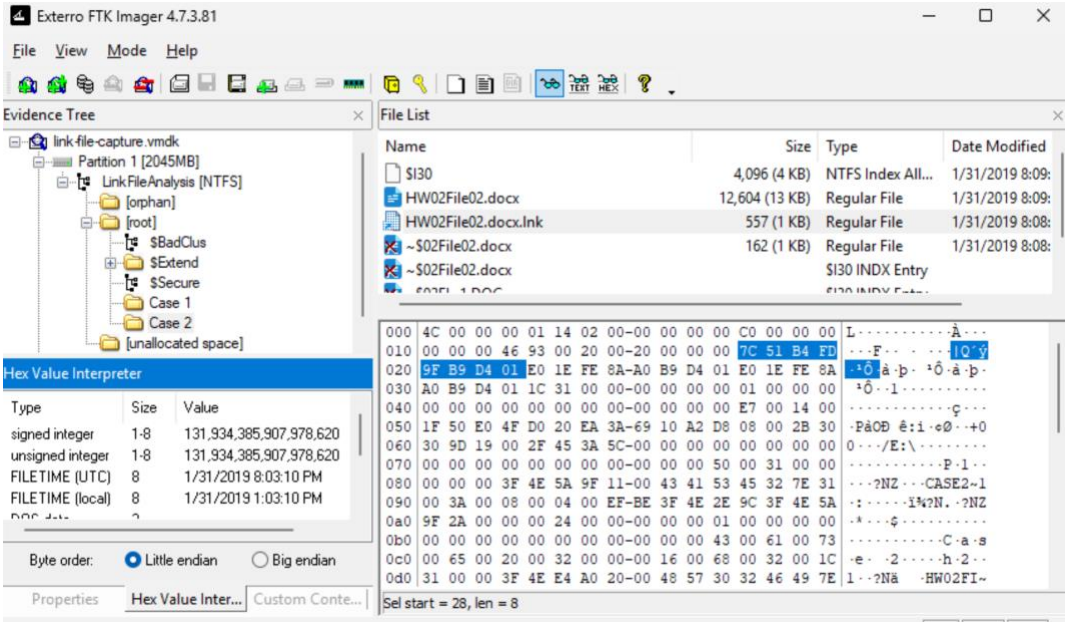

Figure 12. FTK Imager showing the hex for HW02File02.doc.lnk that shows the offset 28

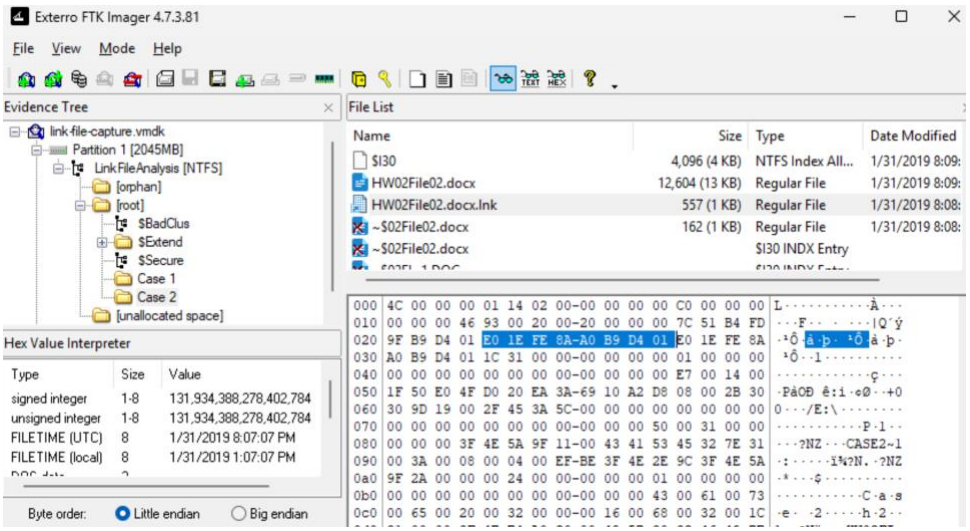

Figure 13. FTK Imager of the HW02File02.docx.link showing hex showing offset 36

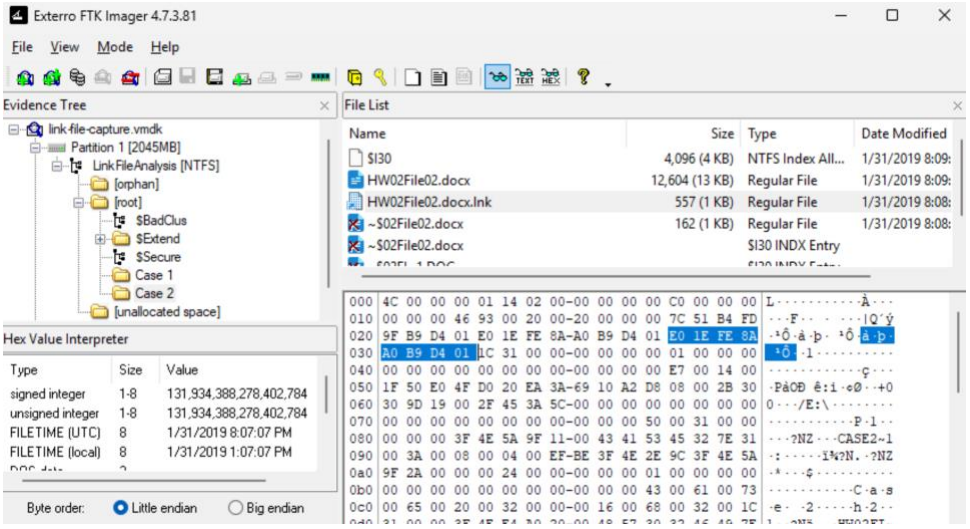

Figure 14. FTK Imager of HW02File02.docx.lnk showing hex showing the offset 44

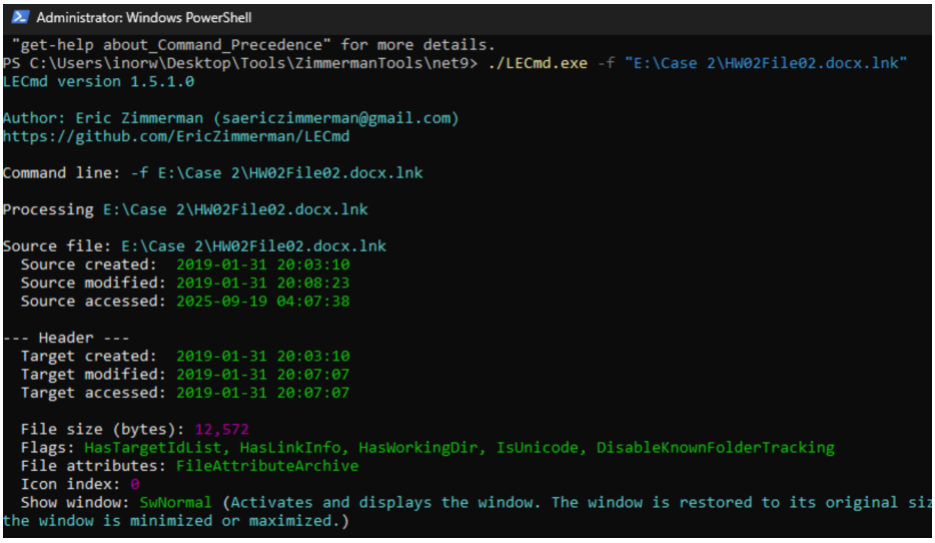

Figure 15. LECmd.exe command and output of the HW02File02.docx.lnk linklist file

4. For each of the following activities, report if the activity occurred. If so, report when it occurred, and how you know. (If it cannot be determined, indicate as much.): a. Target File created – b. Target File last opened – c. Target File last saved – d. Target File last closed

- **Analysis Performed:**
  - The link-file-capture.vmdk was analyzed through the AccessData FTK Imager application.
  - During proper .lnk file creation, the embedded target modified and access date stamps are cached and embedded within the .lnk file. Then .lnk file was fully created which creates its own metadata (time stamps). Then if the target file is later saved/accessed, then it's modified and access date stamps are modified on the target file and since the .lnk isn't rewritten, the embedded target time stays at the original date time.
  - NOTE: All of the files' creation date/time stamp is the same.
  - Path: *link-file-capture.vmdk/Partition 1 [2045MB]/LinkFileAnalysis [NTFS]/[root]/Case 2/HW02File02.docx.lnk*
  - Path: *link-file-capture.vmdk/Partition 1 [2045MB]/LinkFileAnalysis [NTFS]/[root]/Case 2/HW02File02.docx*
  -
- **Answer:**

  As you may notice, the target file's modified/access time, the link file's modified/access time, and the embedded modified/access time of the target file within the link file is inconsistent as shown in Figures 16, 17, and 18. One may presume that this is an error however it is completely normal. The creation time of all three is the same so the target file was definitely created on **01/31/2019 08:03:10 PM UTC.** Since the modified/access time is different for all three, it is imperative to rely on the metadata of the target file itself to get the target file's accurate last opened and saved datetime stamps.

  a. The Target file was created on **01/31/2019 08:03:10 PM UTC**, this is shown in Figure 16. This is obtained by using the FTK Imager application to find the MAC d/t.
  b. The Target file last opened was on **01/31/2019 08:09:23 PM UTC,** this is shown in Figure 16. This is obtained by using the FTK Imager application to find the MAC d/t.
  c. The Target file last saved was on **01/31/2019 08:09:23 PM UTC**, this is shown in Figure 16. This is obtained by using the FTK Imager application to find the MAC d/t.
  d. Due to the nature of FTK Imager and the file's metadata, **it cannot be determined to find the Target file's last closed date/time stamp.**

- **Supporting Evidence:**

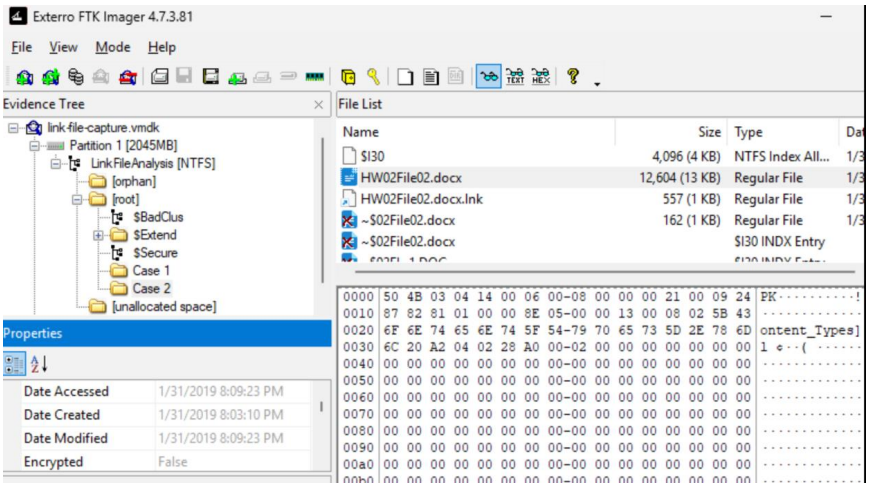

Figure 16. FTK Imager of HW02File02.docx

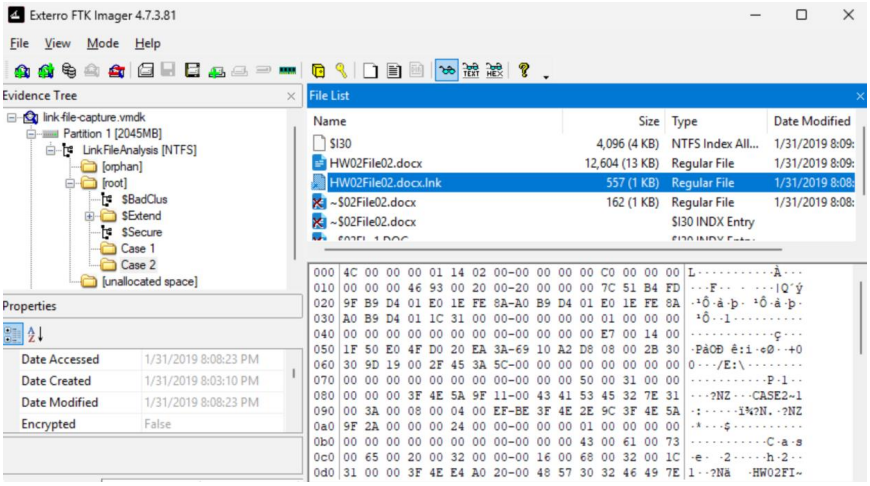

Figure 17. FTK Imager of HW02File02.docx.lnk

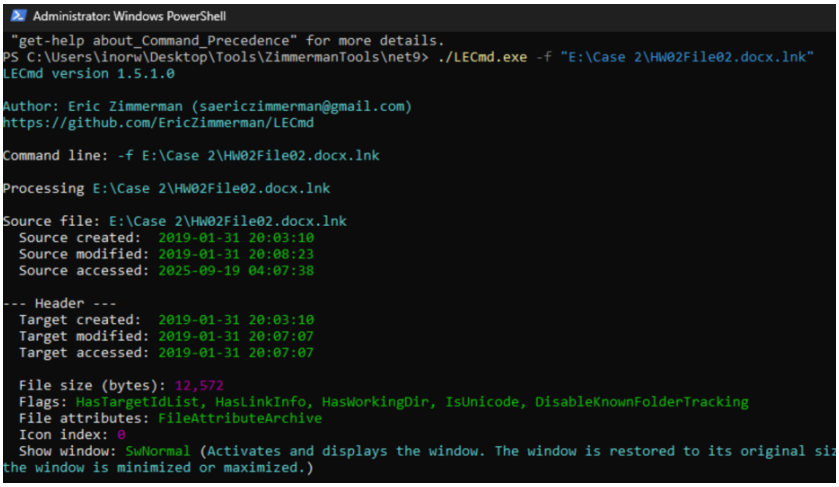

Figure 18. LECmd.exe command and output of HW02File02.docx.lnk showing the embedded MAC d/t of the target file from the link file

5. Why is the target file's modified time, the link file's modified time, and the embedded modified time of the target file within the link file inconsistent (different from each other)?

- **Analysis Performed:**
  - The link-file-capture.vmdk was analyzed through the AccessData FTK Imager application and Eric Zimmerman's LECmd.exe tool.
  - The target file was analyzed through FTK Imager as shown in Figure 19.
  - The link file was analyzed through FTK Imager as shown in Figure 20.
  - The embedded MAC d/t of the target file within the link file was analyzed through LECmd.exe as shown in Figure 21.
  - The examiner used OSINT resources to understand the concepts of the target file, link file, and embedded MAC d/t.
  - Path: *link-file-capture.vmdk/Partition 1 [2045MB]/LinkFileAnalysis [NTFS]/[root]/Case 2/HW02File02.docx.lnk*
  - Path: *link-file-capture.vmdk/Partition 1 [2045MB]/LinkFileAnalysis [NTFS]/[root]/Case 2/HW02File02.docx*
- **Answer:**
  The three timestamps differ because they describe different events on different artifacts, captured at different moments. The **target file's modified time** is the NTFS LastWrite for the document itself and **reflects the actual moment the file was saved**. The **.lnk file's modified time belongs to the shortcut and records when Windows wrote or refreshed the shortcut** (e.g., at creation or when updating path/icon/tracker data); **this is independent of later edits to the document.** The **embedded target modified time inside the .lnk is merely a cached snapshot of the target's timestamps taken when the shortcut was last written, and it does not update if the document is edited afterward or opened directly.** In this case, creation times line up (confirming the creation event), while modified/access times diverge because the shortcut was written at one time and the **document was saved again later**.

- **Supporting Evidence:**

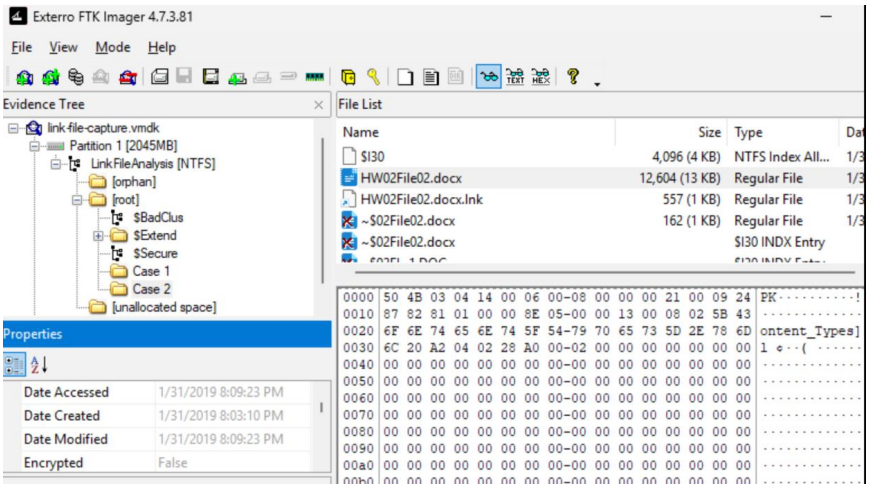

Figure 19. FTK Imager of HW02File02.docx

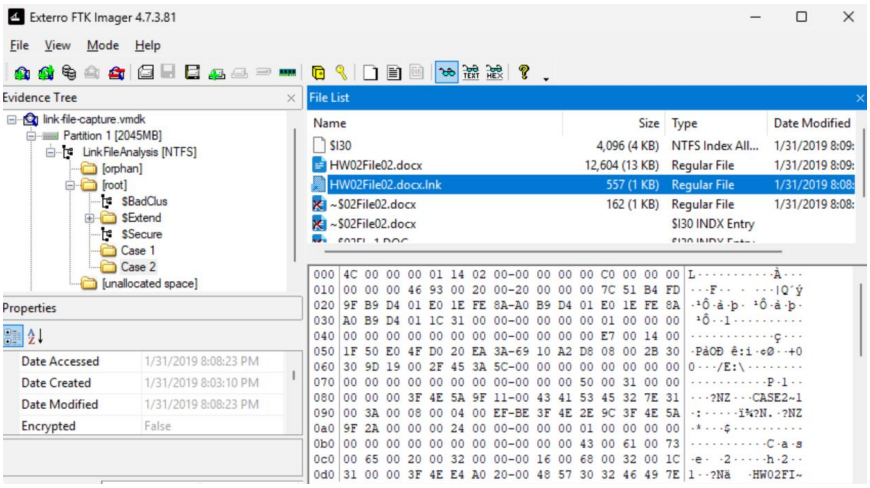

Figure 20. FTK Imager of HW02File02.docx.lnk

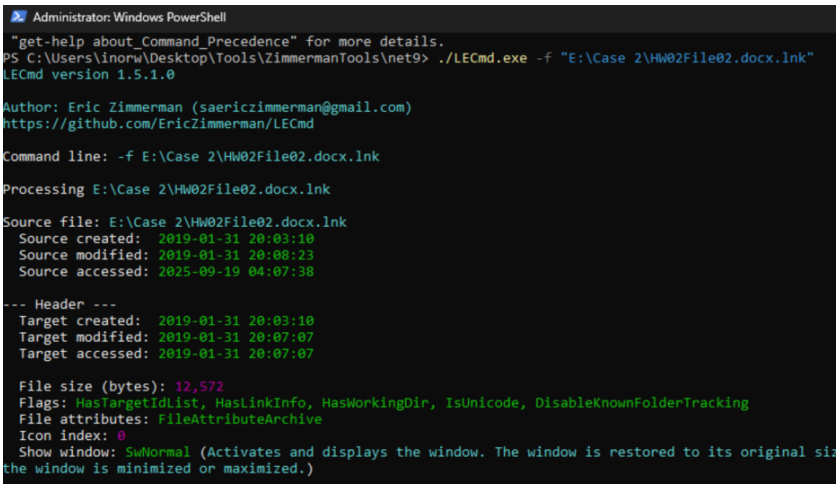

Figure 21. LECmd.exe command and output of HW02File02.docx.lnk showing the embedded MAC d/t of the target file from the link file

--- General Questions ---

1. During your investigation, you find a target file whose creation time comes after the creation time of its associated link file. Explain what happened (assume the user did not deliberately changed the d/t stamps).

- **Answer:**
  If the target file's creation time is later than the creation time of its .lnk, the explanation is that the two artifacts recorded at different moments. Windows wrote the shortcut first (when an earlier instance of the document was opened), then later the target file at that path was deleted and then recreated at the same directory and name, which resets the target file's creation time, while the shortcut was not rewritten since the target file was not opened. Because the .lnk stores only a cached snapshot of the target's timestamps at the moment the shortcut was written, its times (and the embedded target times it carries) can predate the current file. **In short: the shortcut is older, the target is a newer instance, and the difference is expected without any deliberate timestamp manipulation.**

  This is shown as the examiner recreated this situation shown in Figures 22 to 24.

- **Supporting Evidence:**

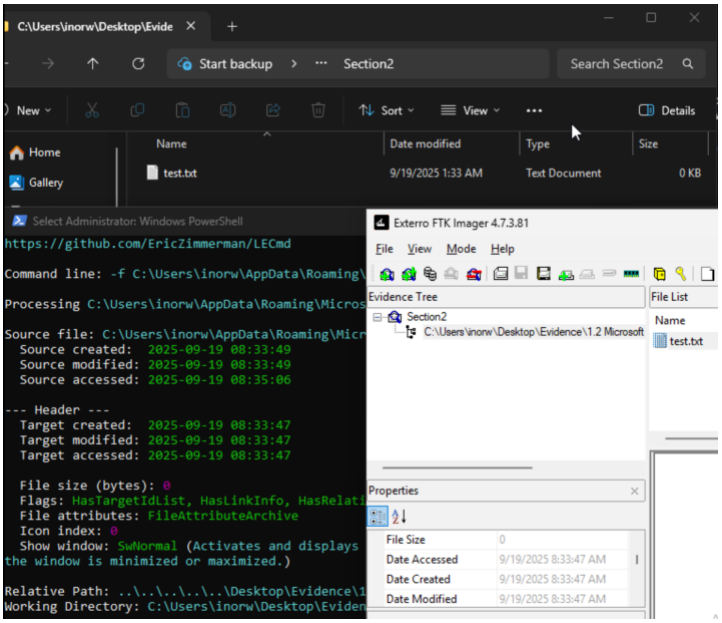

Figure 22. Showing the test.txt (and it's date created within FTK Imager) and the corresponding created time of the .lnk file associated

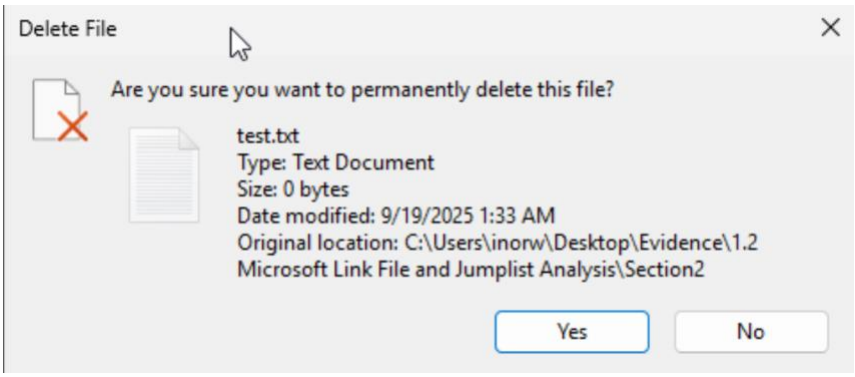

Figure 23. Permanently deleted the file

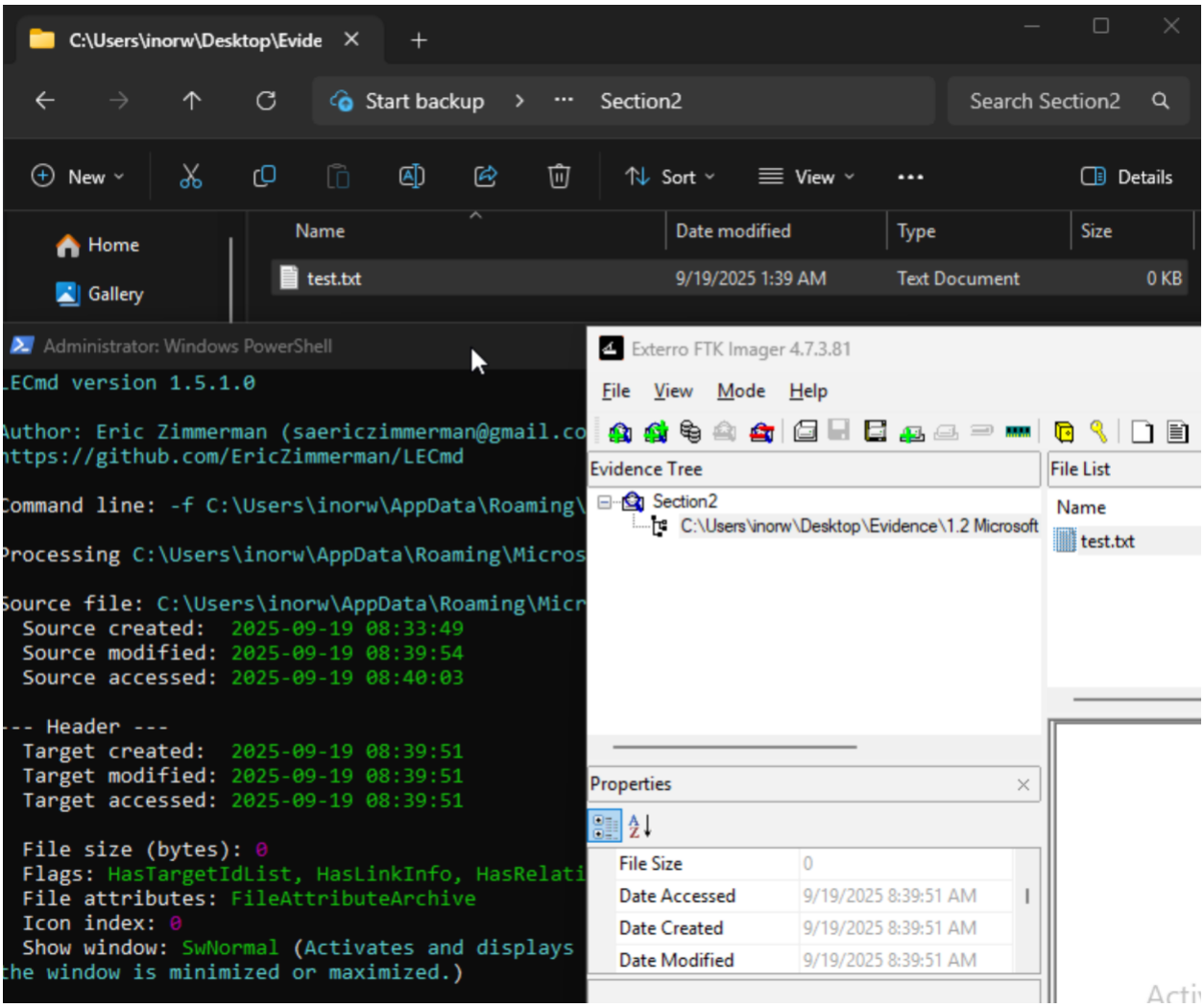

Figure 24. Recreated the test.txt file, reran the LECmd.exe and FTK Imager, shows that target file's date created is after the .lnk file's date created

2. When a target file is moved to another directory, what happens to the associated link file? The target file is then opened from its new directory and immediately closed; what happens to the link file, now? Report the steps you took or the references you consulted to answer these questions.

- **Analysis Performed:**
  - The target file that the examiner created was analyzed through Eric Zimmerman's LECmd.exe tool.
  - The examiner created the file, "inort.txt", on the Desktop which represents the target file. The examiner also accessed and modified the target file.
  - As shown in Figure 25, the examiner used LECmd.exe to examine the working directory of the .lnk file that was created after the examiner accessed and modified the target file.
  - As shown in Figure 26, the examiner used LECmd.exe to examine the working directory of the .lnk file **after** the target file was moved to a different directory.
  - As shown in Figure 27, the examiner used LECmd.exe to examine the working directory of the .lnk file **after** the target file was moved to a different directory (in the previous step) **and** opened from its new directory and immediately closed.
- **Answer:**
  The examiner determined that when the target file was moved to a different directory, the existing shortcut (.lnk) **did not change**, its WorkingDirectory/path remained the same because the target file was never accessed. After the target was then opened from its new directory and immediately closed, the original .lnk still showed **a change**, it's WorkingDirectory/path changed and reflected the new directory that the target file resided in.

  Firstly, the examiner created the file, "inort.txt", on the Desktop which represents the target file. The examiner also accessed and modified the target file. The examiner used LECmd.exe to examine the working directory of the .lnk file. Secondly, the examiner moved the target file into another directory. The examiner used LECmd.exe to examine the working directory of the .lnk file, which was **not changed**. Lastly, the examiner accessed and immediately closed the target file that was moved into a new directory. The examiner used LECmd.exe to examine the working directory of the .lnk file, which was **changed**, and reflected the directory that the target file was moved to.

  Therefore, moving the directory of the target file does not change the link file. However, accessing the target file does change the link file.

- **Supporting Evidence:**

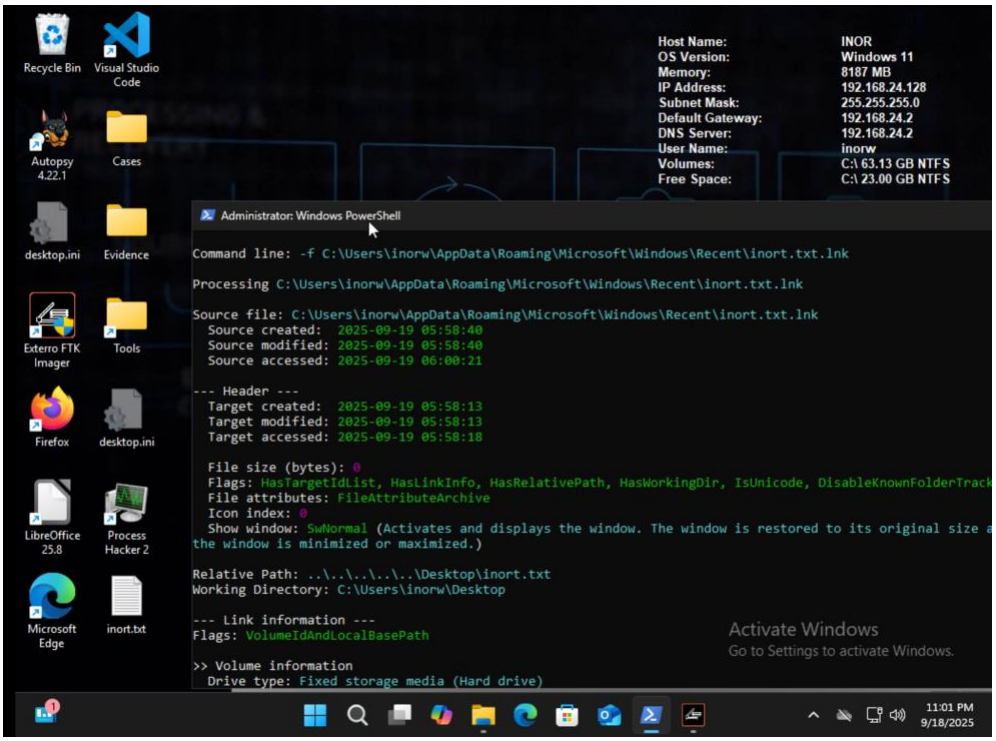

Figure 25. LECmd.exe output of inort.txt.lnk, showing that the working directory is: C:\Users\inorw\Desktop. (THIS IS BEFORE MOVING THE TARGET FILE)

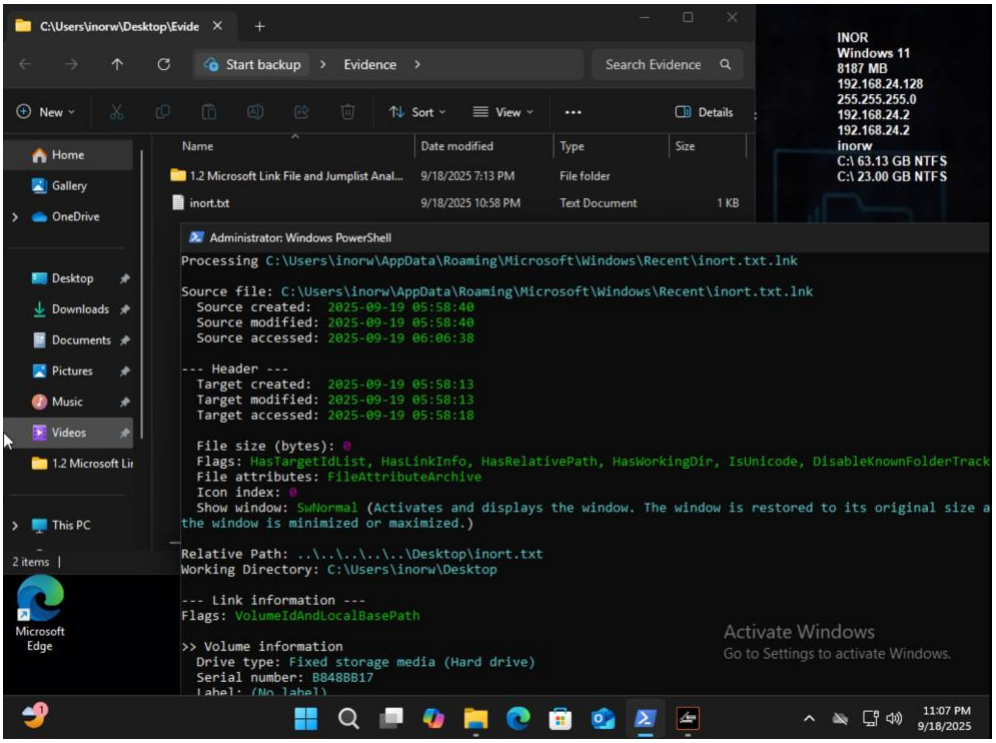

Figure 26. LECmd.exe output of inort.txt.lnk, showing that the working directory is: C:\Users\inorw\Desktop. (THIS IS AFTER MOVING THE TARGET FILE)

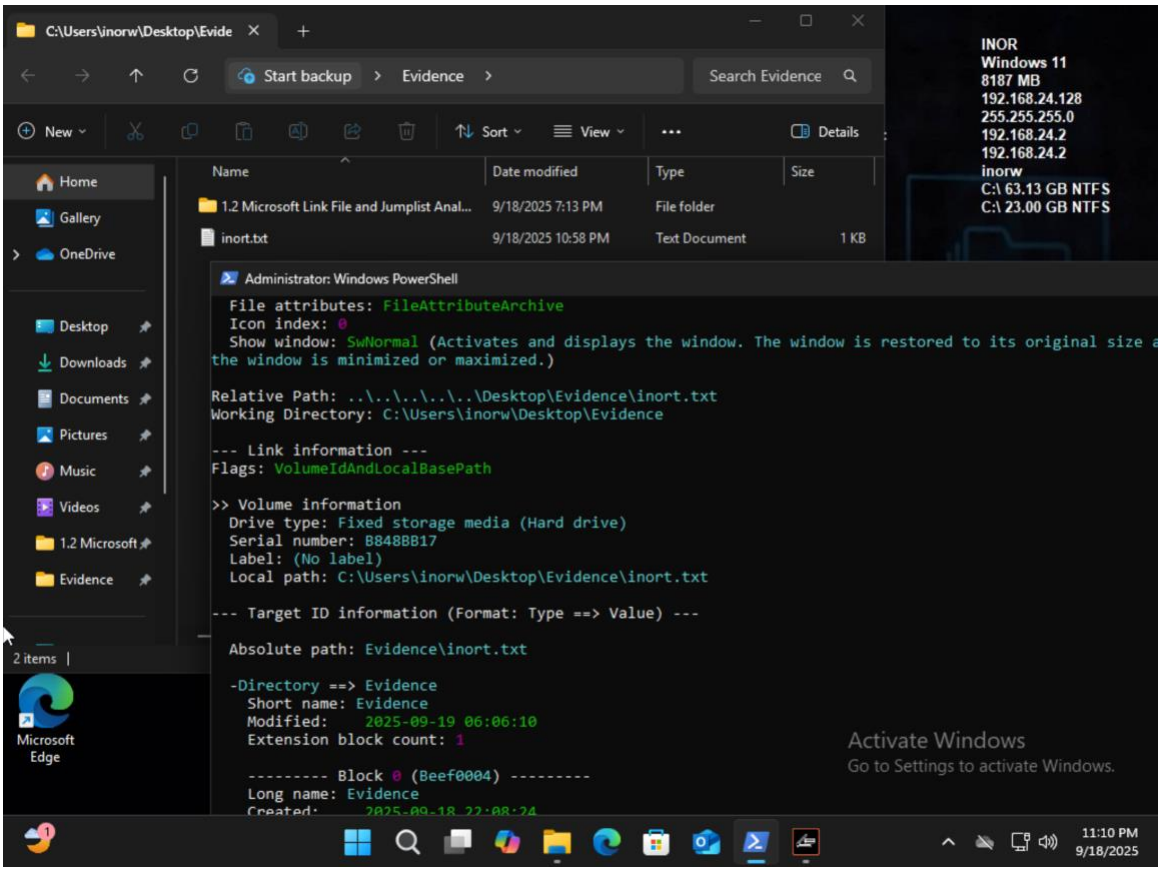

Figure 27. LECmd.exe output of inort.txt.lnk, showing that the working directory is C:\Users\inorw\Desktop\Evidence. (THIS IS AFTER OPENING THE TARGET FILE)

Section 2 – Jumplist

--- Preparation ---

1. Create a word document with the filename "Evidence01.docx" and some content, and save it anywhere you wish.

- **Answer:**
  The examiner firstly downloaded the Microsoft Office suite containing Word. Then, the examiner created a word document with the filename "Evidence01.docx" and added the content "my name is inor!", and saved it to "*C:\Users\inorw\Desktop\Evidence\1.2 Microsoft Link File and Jumplist Analysis\Section2*" as shown in Figure 28.

- **Supporting Evidence:**

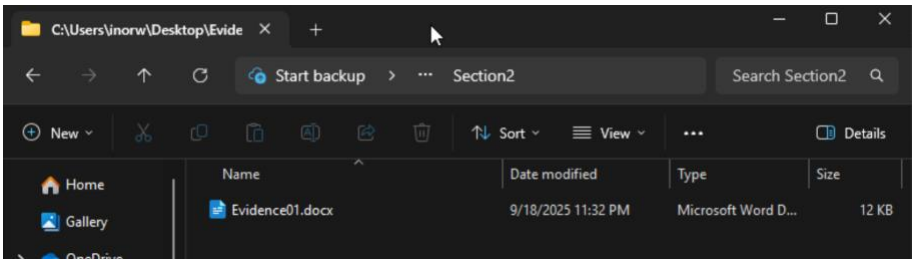

Figure 28. Picture of the Eidence01.docx created

2. Take a screenshot of the recently opened jumplist shown for Word in the Start Menu.

- **Answer:**
  After creating the word document and adding content within it, the examiner accessed and closed the Evidence01.docx file again. Then, pinned the Word application to the Start menu and right clicked the Word application and saw that the recently opened jumplist is shown for Word in the Start Menu as shown in Figure 29.

- **Supporting Evidence:**

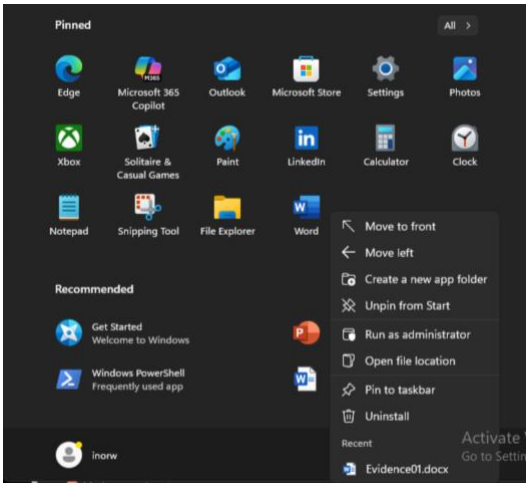

Figure 29. Screenshot of the recently opened jumplist shown for Word in the Start Menu

3. Delete the file you just made. If the file was sent to the Recycle Bin, go to the Recycle Bin and delete it permanently.

- **Answer:**
  The examiner deleted the Evidence01.docx file and it was sent to the Recycle Bin. The examiner then went to the Recycle Bin and deleted it permanently as shown in Figure 30.

- **Supporting Evidence:**

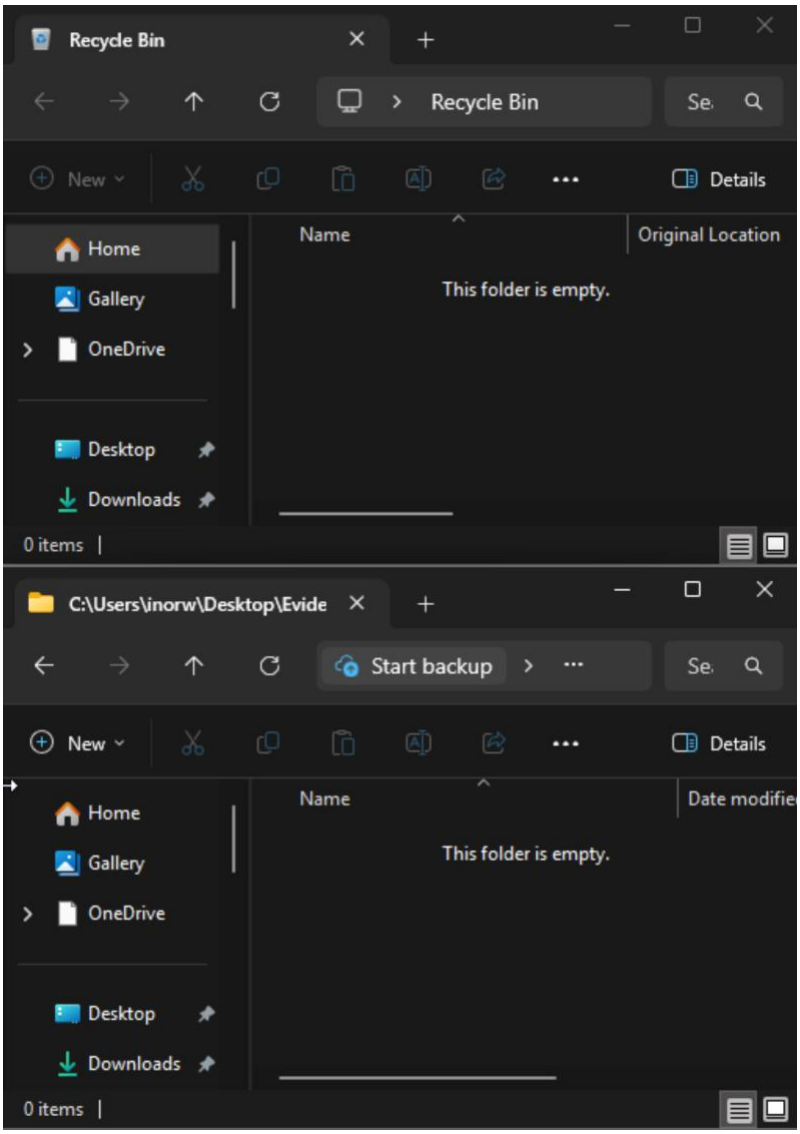

Figure 30. Screenshot of the Recycle Bin and the Section 2 folder showing that the .docx file was deleted

--- Analysis of Jumplists ---

1. Take a screenshot of the recently opened jumplist shown for Word in the Start Menu again. Is Evidence01.docx still there?

- **Answer:**

  The recently opened jumplist shown for Word in the Start Menu of Evidence01.docx is still there, as shown in Figure 31, even after permanently deleting the Evidence.docx file.

- **Supporting Evidence:**

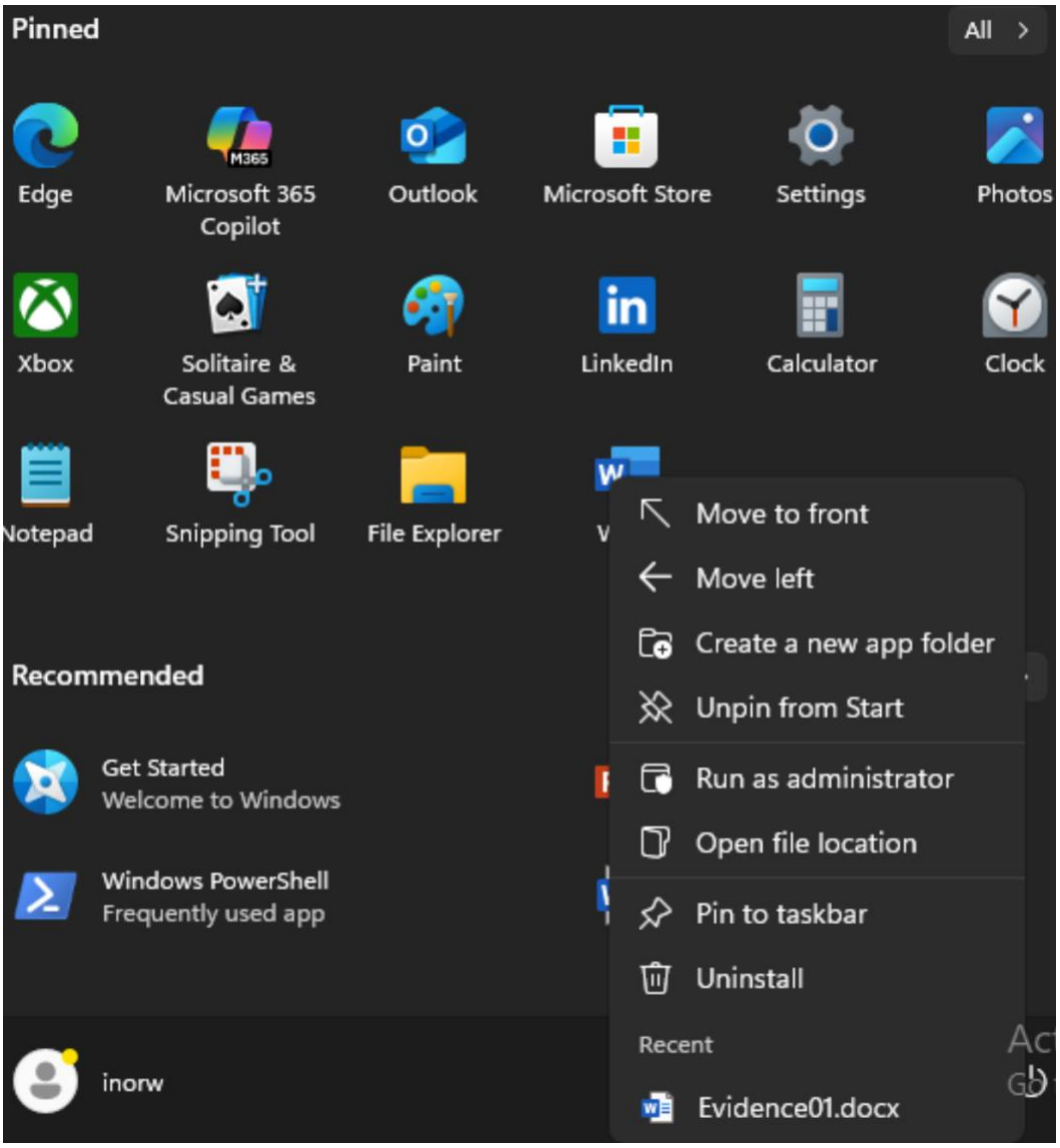

Figure 31. Screenshot of the recently opened jumplist shown for Word in the Start Menu

2. Find the jumplist files in the AutomaticDestinations folder using either CMD or File Explorer. Take a screenshot of these files.

- **Answer:**
  The examiner found the jumplist files in the AutomaticDestinations folder using Windows Powershell as shown in Figure 32. The absolute path is, "*C:\Users\inorw\AppData|Roaming\Microsoft\Windows\Recent\AutomaticDestinations"*. The AutomaticDestinations folder is where Windows stores Jump list data. Jump Lists are the "Recent" and "Pinned" files that appear when you right-click an application as shown in previous steps. Each program that supports Jump Lists has a separate file in the AutomaticDestinations folder. The files are in a binary format with the extension ".automaticDestinations-ms".

- **Supporting Evidence:**

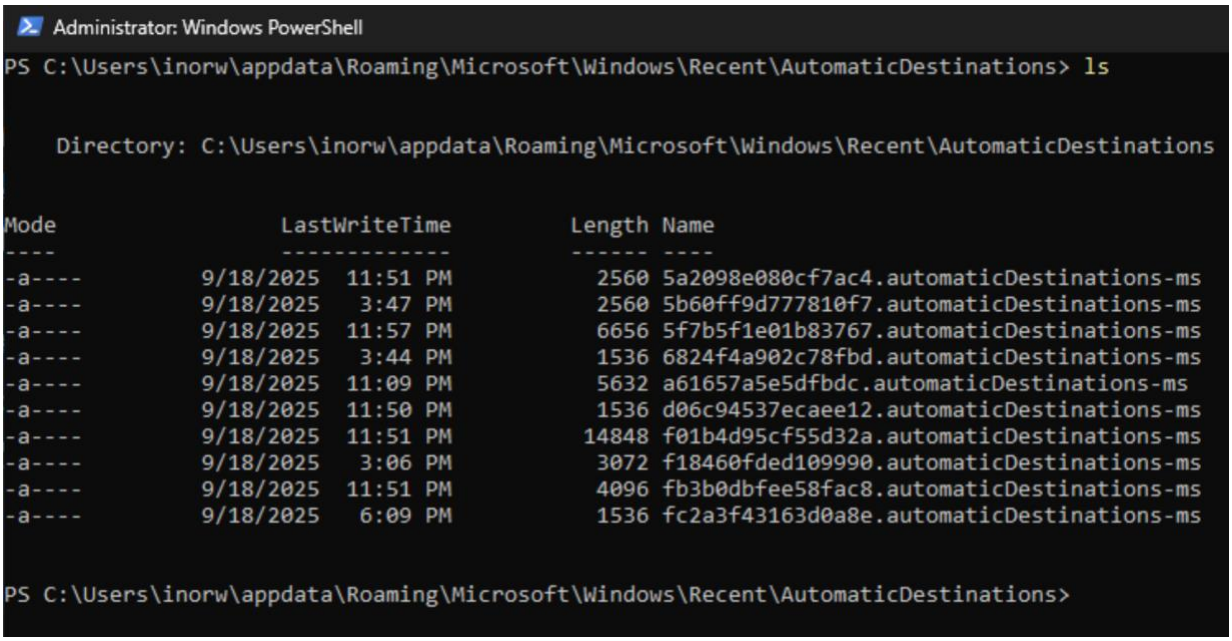

Figure 32. Screenshot of the AutomaticDestinations folder

3. To parse these jumplists, we'll first use NirSoft's JumpListsView tool. After starting the application, find Evidence01.docx in the list. What information do we learn about the file from this Tool? Take note of the corresponding Application ID – we will need it later.

- **Answer:**
  The examiner used Microsoft Edge to download NirSoft's JumpListsView tools from their website. After starting the application, the examiner found Evidence01.docx and double clicked it as shown in Figure 33. This action brought up the properties tab of the jumplist, which contains: filename, full path, record time, created time, modified time, accessed time, file attributes, file size, entry id, application id, application name, file extension, computer name, and jump lists filename.
  NOTE: The Application ID is "**fb3b0dbfee58fac8"**

  NirSoft's JumpListsView is a forensic utility that parses **only** the ".automaticDestinations-ms" and ".customDestinations-ms" files stored in AutomaticDestinations. From this tool, the examiner can extract several important details about user activity. 1. **Application association**; examiner can identify which program the Jump List belongs to. 2. **Recent**; examiner can identify the list of files that were recently accessed. 3. **File Path**; examiner can display the full path to each file referenced by the Jump List. 4. **Timestamps**; examiner can identify timestamps for when items were last accessed. 5. **Metadata**; examiner can identify additional metdata such as file size, entry ID, and application ID.

  NOTE: Jump List artifacts remain even after the original files are deleted which is also shown in this section of the lab. Therefore, jump lists are valuable for understanding what happened even if the original files were deleted.

- **Supporting Evidence:**

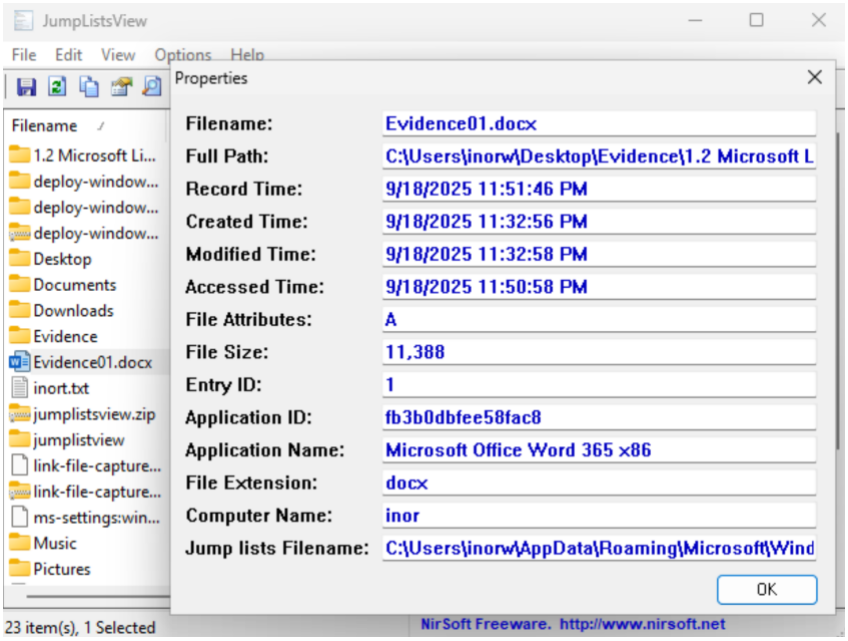

Figure 33. Screenshot of the Properties tab of the Evidence01.docx jumplist

4. Close JumpListsView. Next, we'll be using Eric Zimmerman's JumpLists Explorer tool. If you are having trouble running it, follow the "Requirements and Troubleshooting" instructions at the bottom of the webpage where you downloaded the tool. (You may need to install Microsoft .NET Framework 4.6)

- **Answer:**
  The examiner went to Eric Zimmerman's GitHub to download the JumpListExplorer 2.1.0. After reading the "Requirements and Troubleshooting" instructions at the bottom of the webpage, the examiner noted that it was crucial to use 7-Zip to extract the .zip file as Windows will block the DLLs. After completion, the folder was moved to the Tools directory within the Desktop and the examiner ran the application with success as shown in Figure 34!

- **Supporting Evidence:**

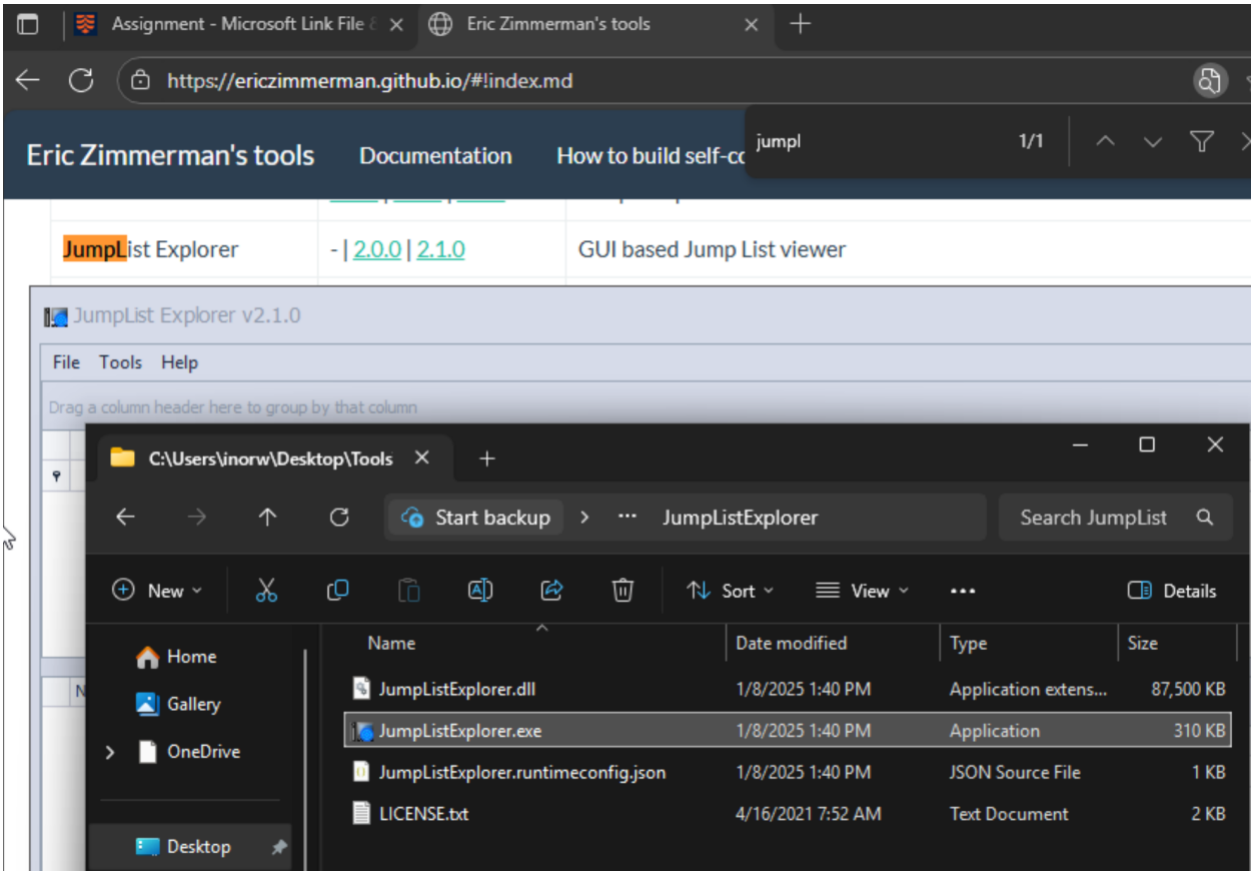

Figure 34. Screenshot showing that Eric Zimmerman's JumpListExplorer tool was successfully installed and running

5. Click File > Load jump lists. Navigate to the AutomaticDestinations folder and select the jumplist file with the AppID you noted in Step 3 of this section. You will see a list of entries – each corresponds to a different Word document recently opened on this machine. Find the entry for Evidence01.docx and double-click it. A new window opens with detailed information regarding the file.

- **Answer:**
  The examiner clicked File > Load jumplists, then navigated to AutomaticDestinations folder (which was hidden, needed to type the absolute path into Explorer), and selected the jumplist file with the AppID noted in Step 3 ("fb3b0dbfee58fac8"), as shown in Figure 35. Then the examiner double-clicked it and a new window opens with detailed information regarding the jumplist file as shown in Figure 36.

- **Supporting Evidence:**

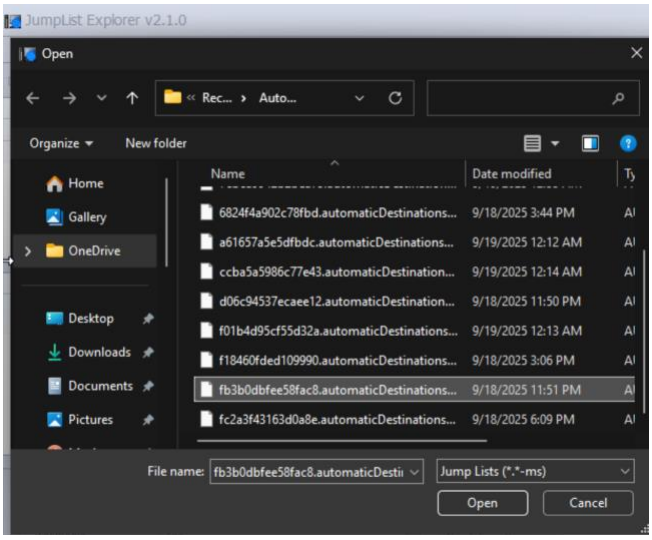

Figure 35. Screenshot showing that the correct Jump List was imported into JumpList Explorer (Evidence01.docx)

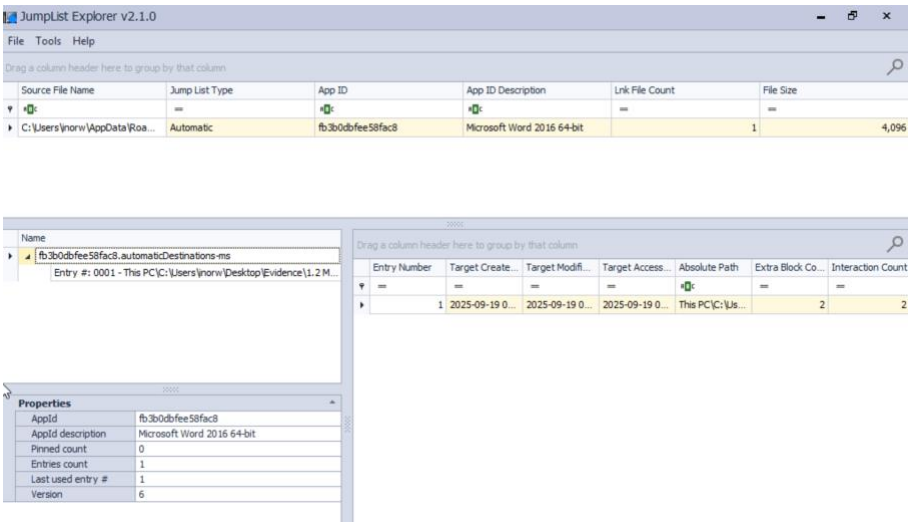

Figure 36. Screemshot of the JumpList Explorer tab

6. Browse the information available in this window; there are multiple tabs. There is one artifact in particular that this tool provides us (and NirSoft's tool does not) which could prove highly useful in a forensic investigation of the Evidence01.docx file. Take a screenshot of this information and explain why it is so potentially useful.

- **Answer:**
  The Upon examining the JumpList Explorer output for Evidence01.docx, a key artifact was identified that is not available in NirSoft's JumpListView tool: the MAC address of the system's network interface card (NIC). In this case, the MAC address 00:29:6a:e6:f3:63 is embedded within the Jump List metadata. This artifact is highly valuable in forensic investigations because the MAC address is a globally unique identifier tied to a specific machine. By capturing it, an examiner can directly associate the creation and modification of Jump List entries with the exact system on which the activity occurred. This strengthens attribution, ensuring that the Evidence01.docx file was accessed or opened on this particular host. In multi-user or multi-device environments, this level of detail helps eliminate ambiguity and provides corroborative evidence when linking a file access event back to a suspect system.

- **Supporting Evidence:**

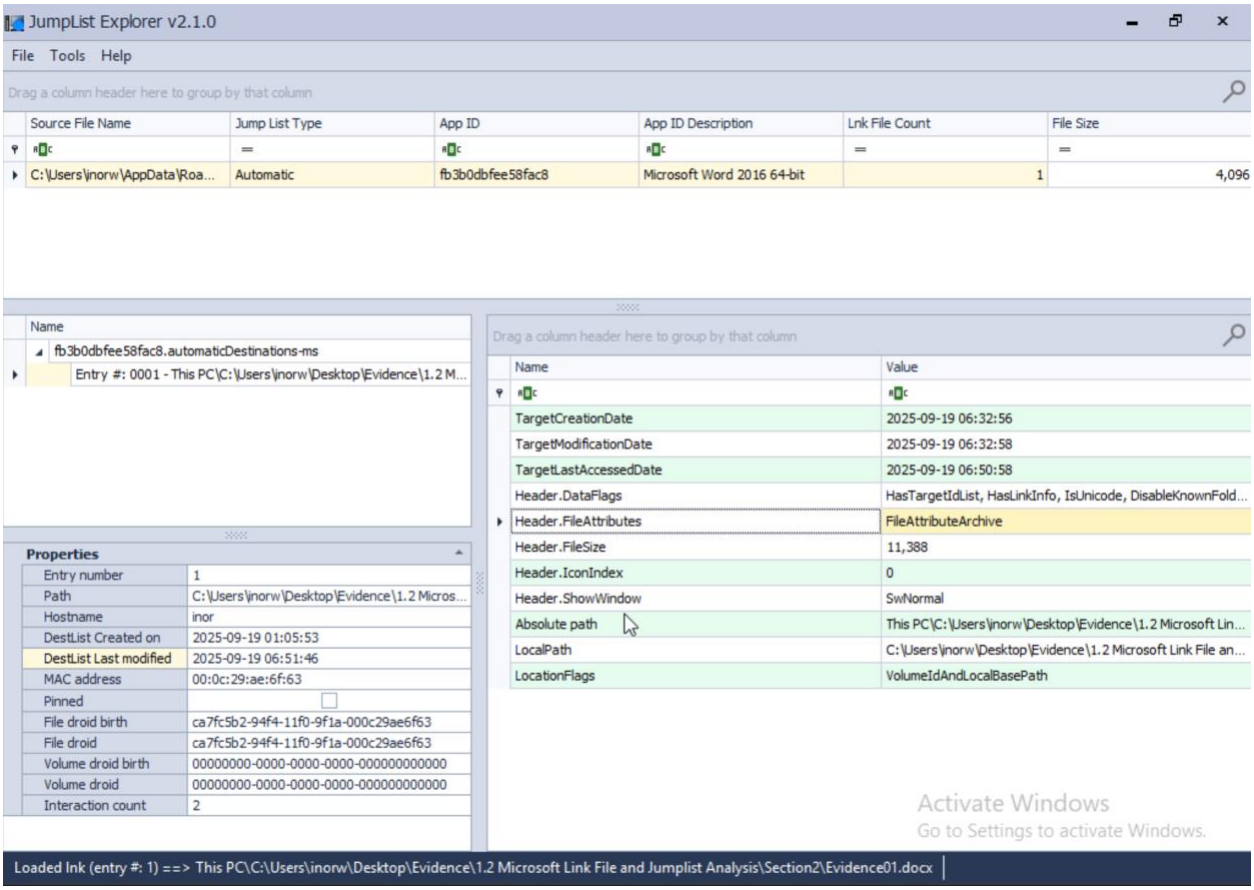

Figure 37. Jumplist Explorer showing the MAC address

Conclusion

The examiner, Inor Wang, enjoyed this lab! There is no critique from me. Thank you.

References

Carvey, H. A. (2014). Windows forensic analysis toolkit: Advanced analysis techniques for Windows 8 (Fourth edition). Syngress.

Cyber Triage. (n.d.). *What is a Jump List?* https://www.cybertriage.com/blog/what-is-a-jump-list/.

Frazer, A. (2020, February 19). *The missing LNK—Correlating user search LNK files*. Google Cloud Blog (Mandiant). https://cloud.google.com/blog/topics/threat-intelligence/the-missing-lnk-correlating-user-search-lnk-files.

Johansen, G., & Safari, an O. M. C. (2020). Digital Forensics and Incident Response—Second Edition.

Ligh, M. H., Case, A., Levy, J., & Walters, A. (2014). The art of memory forensics: Detecting malware and threats in Windows, Linux, and Mac memory. Wiley.

Malware forensics field guide for Windows systems digital forensics field guides. (2012). Syngress.

Microsoft. (n.d.). *Where are .lnk extension files stored?* Microsoft Learn Q&A. https://learn.microsoft.com/en-us/answers/questions/2597485/where-are-ink-extension-files-stored.

Microsoft. (n.d.). *Windows.UI.StartScreen.JumpList class*. Microsoft Learn. https://learn.microsoft.com/en-us/uwp/api/windows.ui.startscreen.jumplist?view=winrt-26100.

Oettinger, W., & Safari, an O. M. C. (2020). Learn Computer Forensics.

Reddit. (n.d.). *The folder* [Online forum post]. r/csharp. Retrieved September 19, 2025, from https://www.reddit.com/r/csharp/comments/tcesr4/the_folder/.

Reddy, N. (2019). Practical cyber forensics: An incident-based approach to forensic investigations. APress. https://doi.org/10.1007/978-1-4842-4460-9.

Rocha, L. (2017, November 22). *Digital forensics – Artifacts of interactive sessions*. Count Upon Security. https://countuponsecurity.com/2017/11/22/digital-forensics-artifacts-of-interactive-sessions/.

Sofer, N. (n.d.). *JumpListsView*. NirSoft. https://www.nirsoft.net/utils/jump_lists_view.html.

Stack Overflow. (2024, June 2). How to convert Windows FILETIME 64-bit hex to date and time? Retrieved from https://stackoverflow.com/questions/78566947/how-to-convert-windows-filetime-64bit-hex-to-date-and-time.

Wikipedia contributors. (n.d.). *Shortcut (computing).* In *Wikipedia, The Free Encyclopedia.* Retrieved September 19, 2025, from https://en.wikipedia.org/wiki/Shortcut_(computing).

Zimmerman, E. (2016, February). *Introducing LECmd*. Binary Foray. https://binaryforay.blogspot.com/2016/02/introducing-lecmd.html.

Zimmerman, E. (n.d.). *Eric Zimmerman's tools (documentation site)*. https://ericzimmerman.github.io/#!index.md.