Cyberster

INTERNSHIP REPORT

A Weekly Progress and Learning Summary

Phase Two : Disk Imaging, File Systems, and Evidence Analysis

Submitted to: Cyberster Internship Program

Intern Name: Abdul Muqeet Tabraiz

Roll Number: CSI-B1-353

A Weekly Progress and Learning Summary

Executive Summary

This report documents the Week 7 activities of the Cyberster Blue Team Internship, covering Phase Two digital forensics foundations. The week was structured around three blocks of work: forensic disk imaging and hash verification, NTFS file-system analysis (including Alternate Data Streams, the $MFT, and MACB timestamps), and an initial Autopsy triage of the acquired image. All activities were performed on a Windows 10 host. Kali Linux was used only briefly for the EXT4 research portion of the file-system comparison.

The exercise began with a FAT32-formatted USB drive. Because the $MFT is an NTFS-only structure, the drive was reformatted to NTFS (volume label EVIDENCE_01) and re-imaged to allow a complete MFT analysis. Both MD5 and SHA-1 hashes of the source drive and the resulting E01 image matched, confirming forensic integrity. The image was then processed in Autopsy 4.22.1, where ingest modules, deleted-file recovery, timeline analysis, and report generation were performed.

Key Outcomes

- Forensic E01 image acquired with verified MD5 and SHA-1 hashes (match confirmed).
- Software write-protection applied via Windows registry (WriteProtect = 1).
- Alternate Data Stream creation, detection, and legitimate Zone.Identifier ADS demonstrated.
- $MFT extracted from the NTFS image and analysed in MFT Explorer with full MACB timestamps.
- Five-stage MACB timestamp experiment completed (create, open, modify, copy, move).
- Autopsy case CYB_W7_USB_Investigation created; deleted files recovered and HTML report generated.

Acquisition Integrity — Verified Hashes (Final NTFS Image)

Algorithm	Source Drive	E01 Image	Match
MD5	1127cba3f65d7025b963ab92e942f299	1127cba3f65d7025b963ab92e942f299	YES
SHA-1	72b809fa1dd637395b1b0807f480cfc11fdbe35e	72b809fa1dd637395b1b0807f480cfc11fdbe35e	YES

1. Scope and Environment

1.1 Week Objectives

- Perform a forensically sound acquisition of a USB storage device in E01 format.
- Verify acquisition integrity using MD5 and SHA-1 hash comparison of source vs. image.
- Demonstrate NTFS-specific artefacts: Alternate Data Streams, $MFT records, MACB timestamps.
- Compare NTFS, FAT16/FAT32 and EXT4 file systems from a forensic perspective.
- Load the acquired image into Autopsy, configure ingest modules, and perform initial triage.
- Recover deleted files and generate a formal HTML case report.

1.2 Toolset

Tool	Version	Purpose	Platform
FTK Imager	4.7.1	Disk acquisition & E01 creation	Windows 10
Arsenal Image Mounter	Current	Mount E01 image read-only	Windows 10
MFT Explorer	v2.1.0	Parse and analyse the $MFT	Windows 10
Autopsy	4.22.1	Case management & forensic triage	Windows 10
Sysinternals Streams	v1.60	Detect NTFS Alternate Data Streams	Windows 10
Windows Registry Editor	10.0.19045	Software write-blocking (WriteProtect)	Windows 10
Kali Linux (research only)	Current	EXT4 file-system reference research	Linux

1.3 Important Note on Platform

Windows-Based Week

Week 7 is entirely Windows-based. All primary tools — FTK Imager, Arsenal Image Mounter, MFT Explorer and Autopsy — run on the Windows host machine. Kali Linux was used only briefly for the EXT4 research section of the file-system comparison document.

2. Days 46–47 — Disk Imaging and Hash Verification

This section documents the complete imaging workflow. The initial acquisition was performed against a FAT32-formatted USB drive. Because FAT32 does not contain an MFT, the drive was subsequently reformatted to NTFS and re-imaged so that the $MFT-based analysis in Section 3 could proceed. Screenshots from the initial FAT32 acquisition are retained in this report as process evidence; the final, authoritative image used for downstream analysis is the NTFS E01.

2.1 Step 1 — Prepare USB Drive as Evidence Source

A 7.48 GB USB drive (labelled CLAY, D:) was used as the evidence source. Test data was created on the drive so there would be meaningful artefacts to examine later in Autopsy.

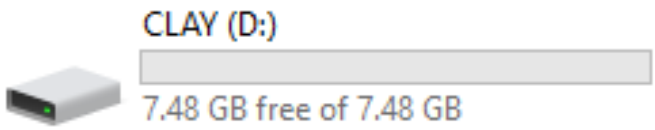

Figure 2.1 — USB drive (CLAY, D:) selected as evidence source (7.48 GB free of 7.48 GB).

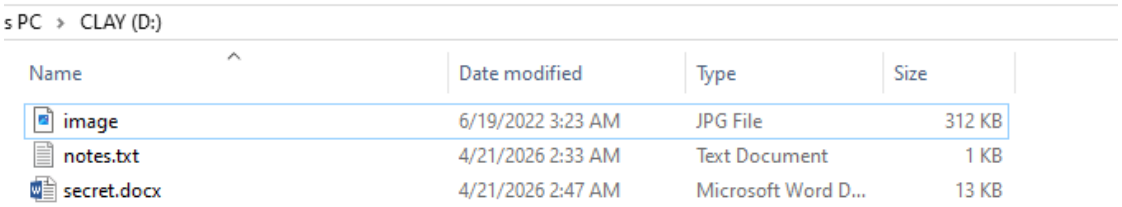

Figure 2.2 — Initial USB contents: image.jpg, notes.txt and secret.docx created as test artefacts.

One of the files (secret.docx) was then deleted to produce a recoverable artefact for later Autopsy triage.

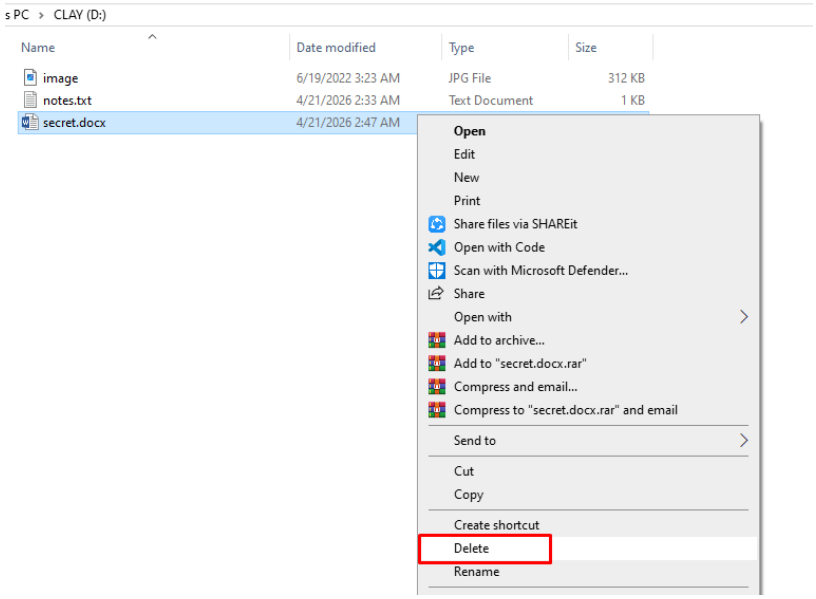

Figure 2.3 — Right-click menu used to Delete secret.docx, producing a recoverable deleted artefact.

2.2 Step 2 — Apply Software Write-Protection

Because no hardware write-blocker was available, Windows registry write-protection was used as the write-blocking method for this lab. The key HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Control\StorageDevicePolicies was located, and a DWORD (32-bit) value named WriteProtect was set to 1.

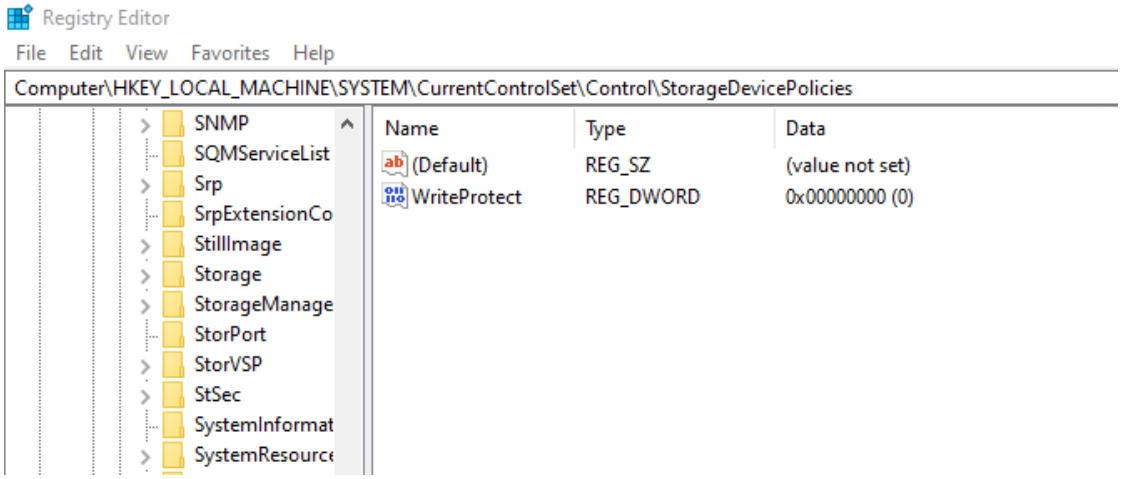

Figure 2.4 — StorageDevicePolicies key located under HKLM\SYSTEM\CurrentControlSet\Control.

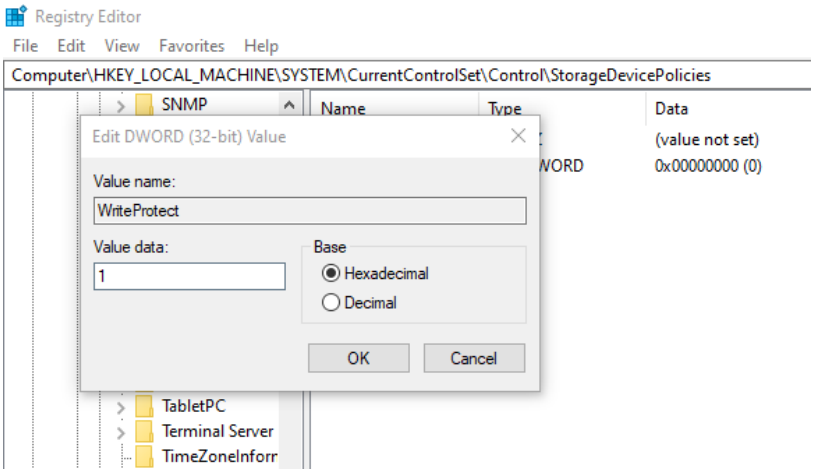

Figure 2.5 — WriteProtect DWORD value set to 1 (Hexadecimal) to enable software write-blocking.

Write-Blocking Limitation (Documented)

Software write-protection via the Windows registry WriteProtect = 1 was applied. This is a software control, not a hardware write-blocker. A hardware write-blocker provides stronger forensic assurance; the registry method is documented here as a lab-environment limitation and is disclosed openly in the Chain-of-Custody document.

2.3 Step 3 — FTK Imager: Create Disk Image (Source Selection)

FTK Imager 4.7.3.81 was launched. File → Create Disk Image was selected, with Physical Drive as the source evidence type and the target USB (\\.\PHYSICALDRIVE1 — Generic Flash Disk USB Device, 8 GB) chosen from the dropdown.

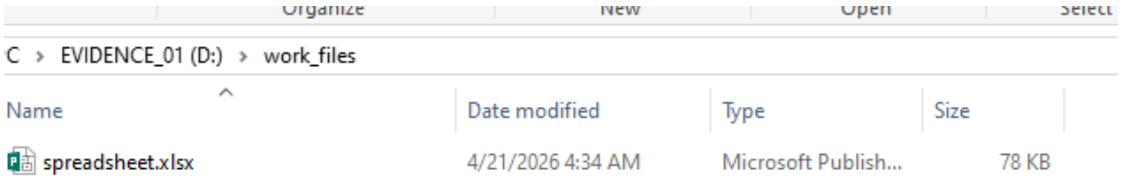

Figure 2.6 — FTK Imager Select Source dialog with Physical Drive selected.

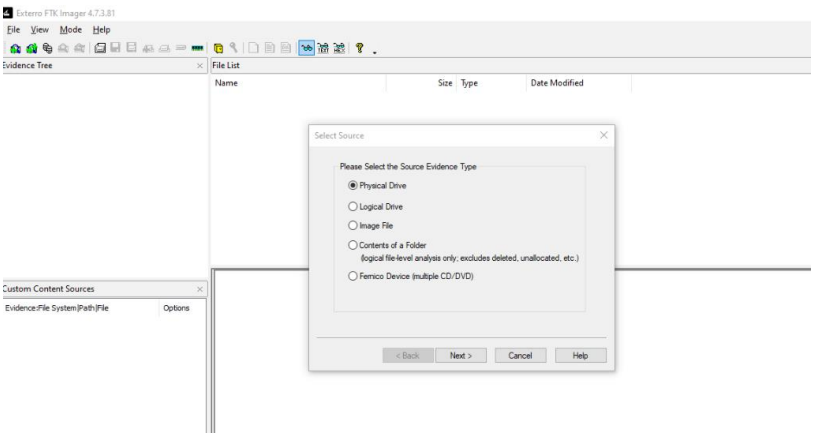

Figure 2.7 — Source drive selection: \\.\PHYSICALDRIVE1 — Generic Flash Disk USB Device (8 GB).

2.4 Step 4 — Configure E01 Output Format

E01 (EnCase) was selected as the destination image type. E01 embeds case metadata (examiner, case number, acquisition date) directly in the image file and supports compression and segmentation — a courtroom-preferred format.

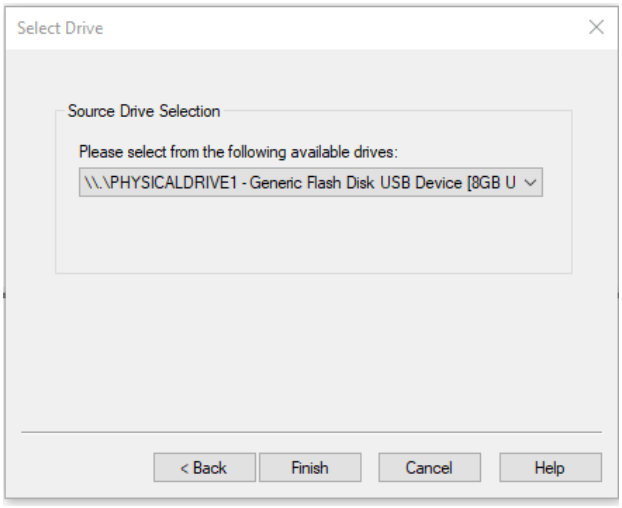

Figure 2.8 — Destination image type set to E01.

The Evidence Item Information was completed exactly as required:

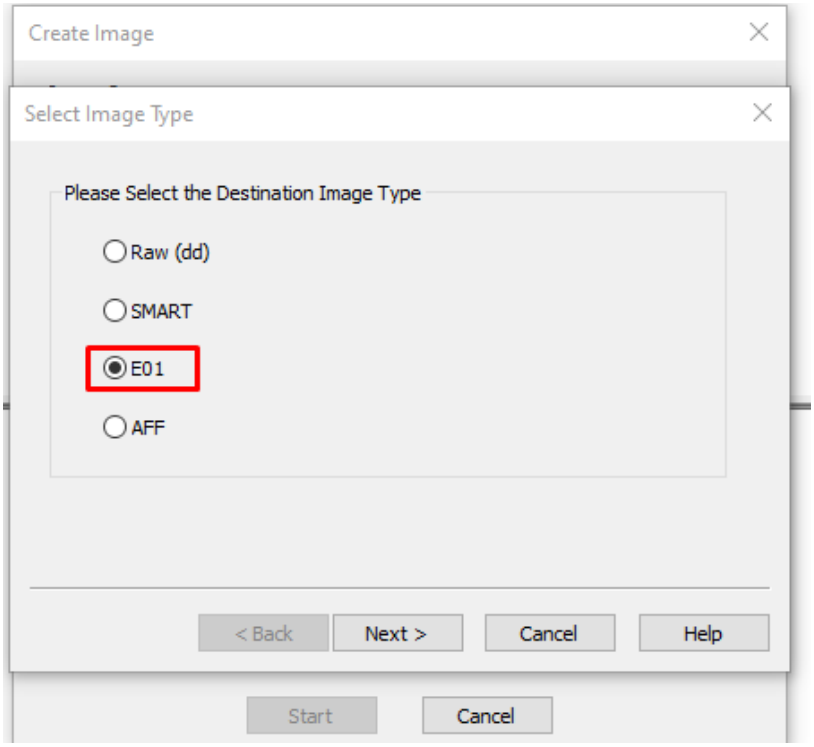

Figure 2.9 — FTK Imager Evidence Item Information form completed with case metadata.

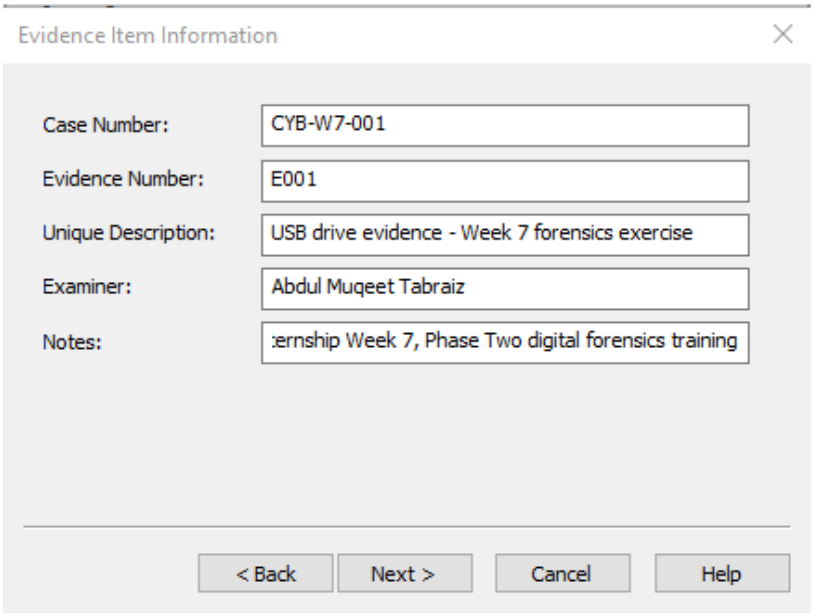

Figure 2.10 — Destination folder E:\Cyberster Internship\Week 7\ForensicLab Images and filename W7_Evidence_USB set. Fragment size 1500 MB, compression 6.

2.5 Step 5 — Enable Hash Verification and Run Acquisition

Both "Verify images after they are created" and "Create directory listings of all files in the image" were enabled. MD5 and SHA-1 hashing were confirmed as active. The acquisition was then started.

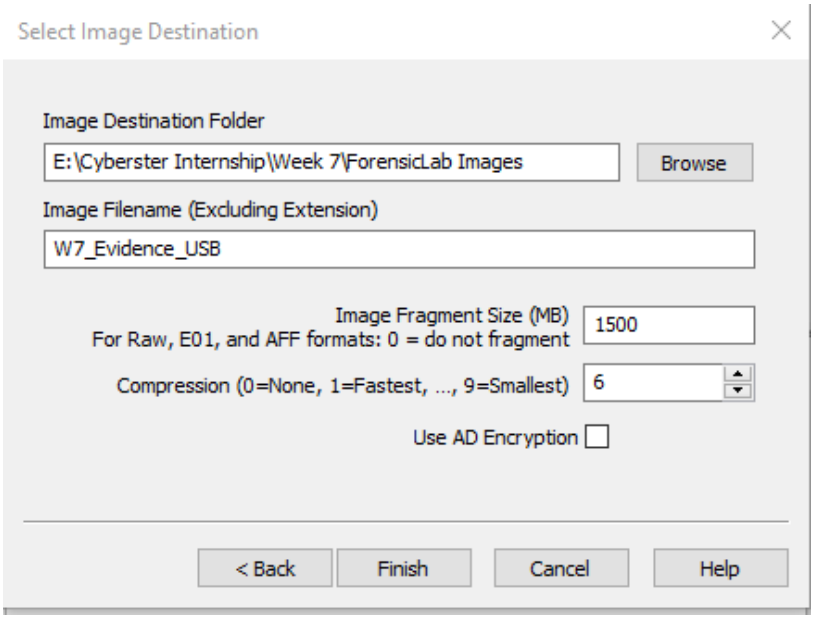

Figure 2.11 — Create Image dialog with Verify and Directory Listing options enabled, ready to Start.

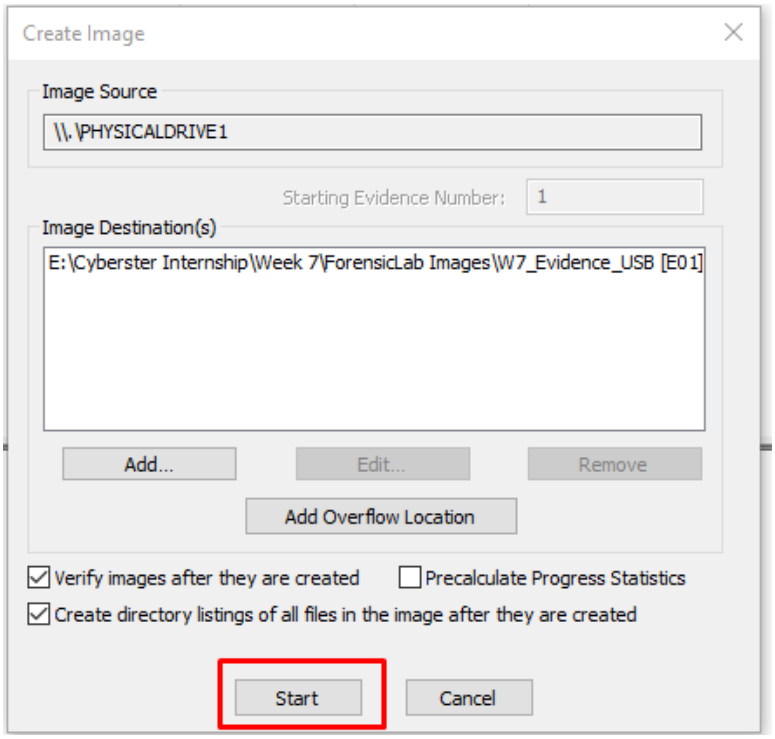

Figure 2.12 — Acquisition in progress: bit-for-bit copy of the source drive being written to the E01 image.

2.6 Step 6 — Hash Verification (FAT32 image — first acquisition)

At completion, FTK Imager produced the Drive/Image Verify Results dialog. Both MD5 and SHA-1 hashes of the source drive and the created image matched, and no bad blocks were found.

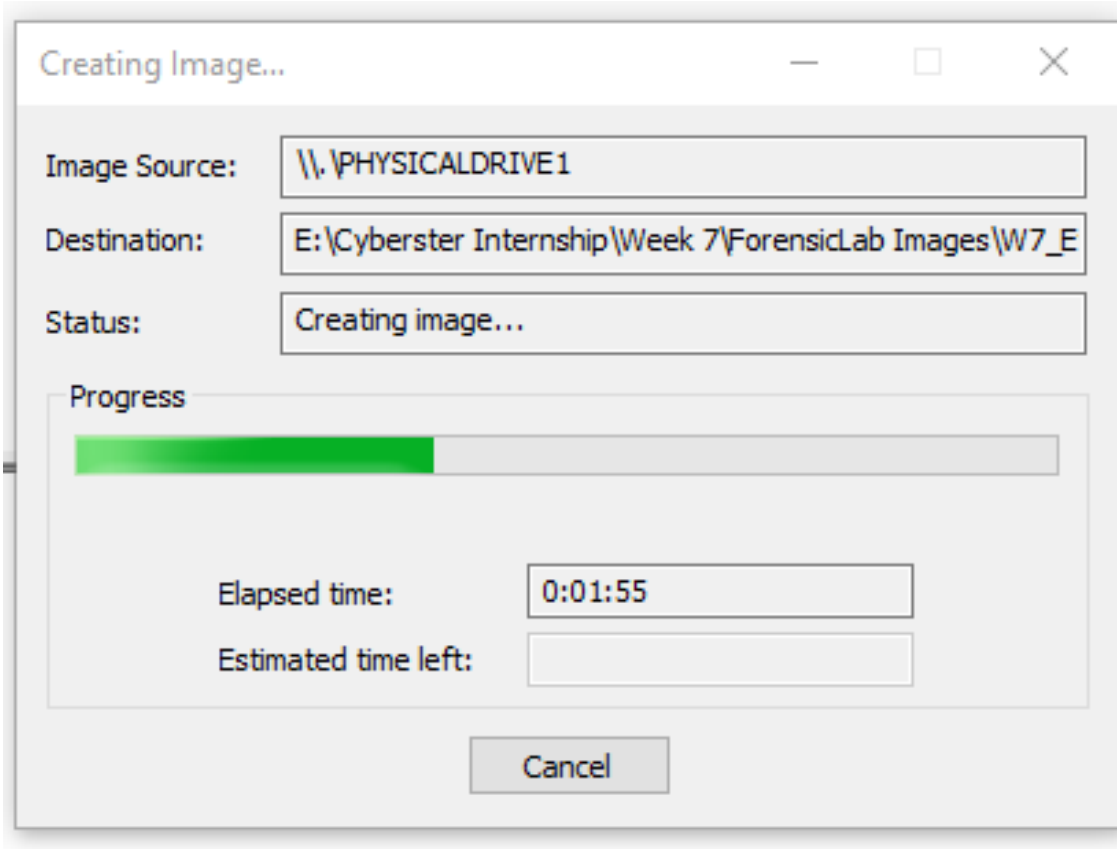

Figure 2.13 — FTK Imager Drive/Image Verify Results for the FAT32 acquisition — MD5 and SHA-1 match confirmed.

FAT32 Acquisition Hash Values

Algorithm	Value	Verify Result
MD5	8069e499cb22190ca556a4b08f4443d2	Match
SHA-1	d0fb78909cd989d99fbb4bf2ee61c9c6e1dcccf7	Match

2.7 Step 7 — Imaged Output Files

The imaging process produced three files in the destination folder: the E01 image, an Excel-compatible summary, and a plain-text case log.

Figure 2.14 — Image output folder showing W7_Evidence_USB.E01, case CSV and case log.

2.8 Step 8 — Mount the E01 Image (Arsenal Image Mounter)

The resulting E01 was mounted using Arsenal Image Mounter in Disk device, read only mode — the mandatory option for forensic review.

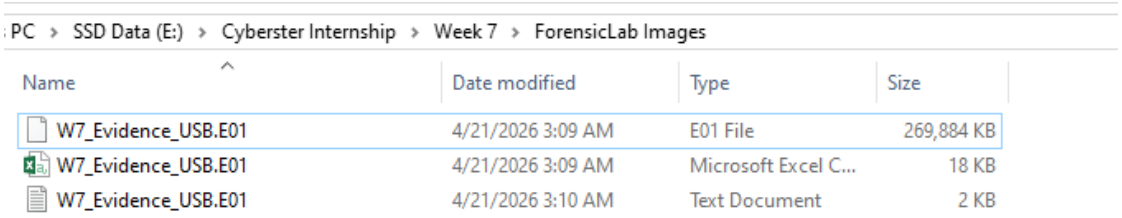

Figure 2.15 — Arsenal Image Mounter options — Disk device, read only selected.

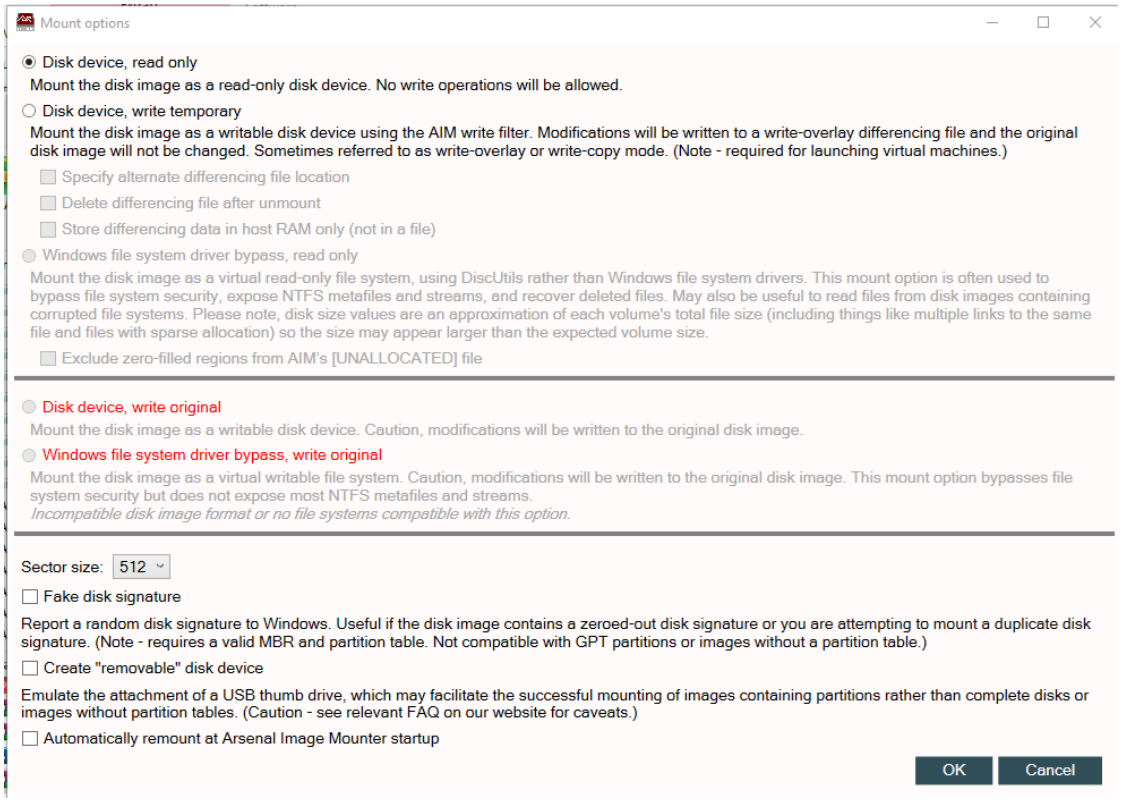

Figure 2.16 — Arsenal Image Mounter showing the W7_Evidence_USB.E01 mounted as drive I:.

The mounted drive letter was then opened in Windows Explorer to confirm the image contents were intact.

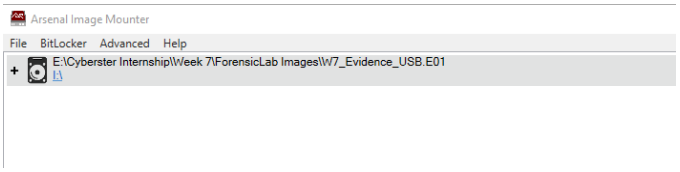

Figure 2.17 — Mounted E01 drive (CLAY, I:) showing the original files — confirms read-only accessibility of the image.

2.9 Step 9 — Unmount After Verification

After verification the image was safely removed from Arsenal Image Mounter, and the Windows registry WriteProtect value was reset to 0 to allow normal USB usage again.

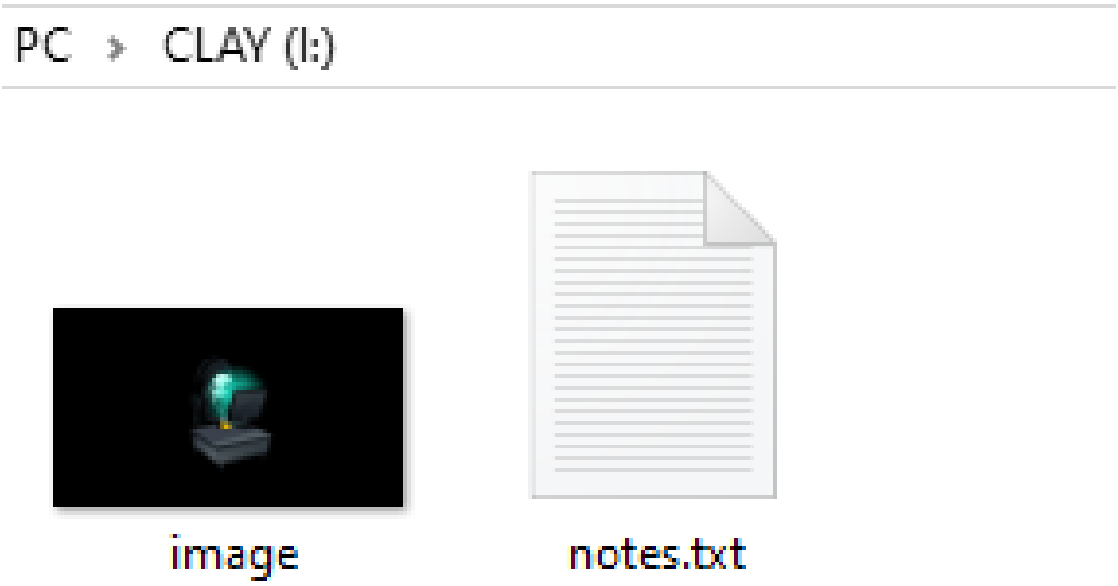

Figure 2.18 — Arsenal Image Mounter — Remove button used to safely detach the mounted image.

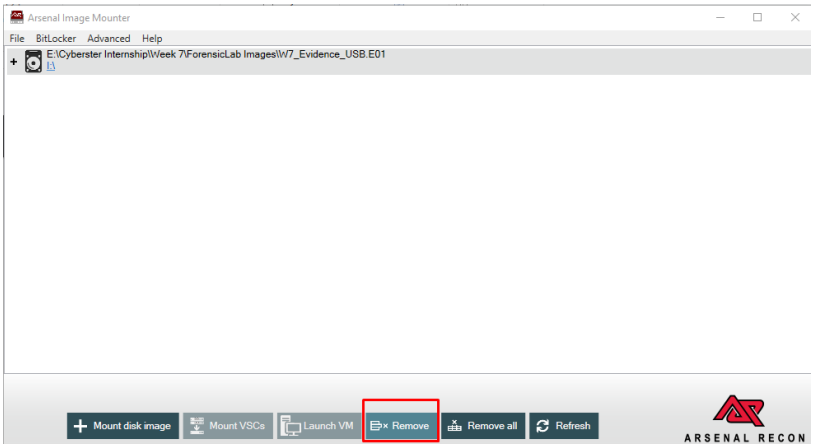

Figure 2.19 — WriteProtect DWORD value reset to 0 so normal USB drives work again.

2.10 Step 10 — Chain-of-Custody Document (Initial)

The Chain-of-Custody document was prepared using the required template fields. A full copy of the final document appears in Appendix A; the initial version (following the first FAT32 acquisition) is shown below.

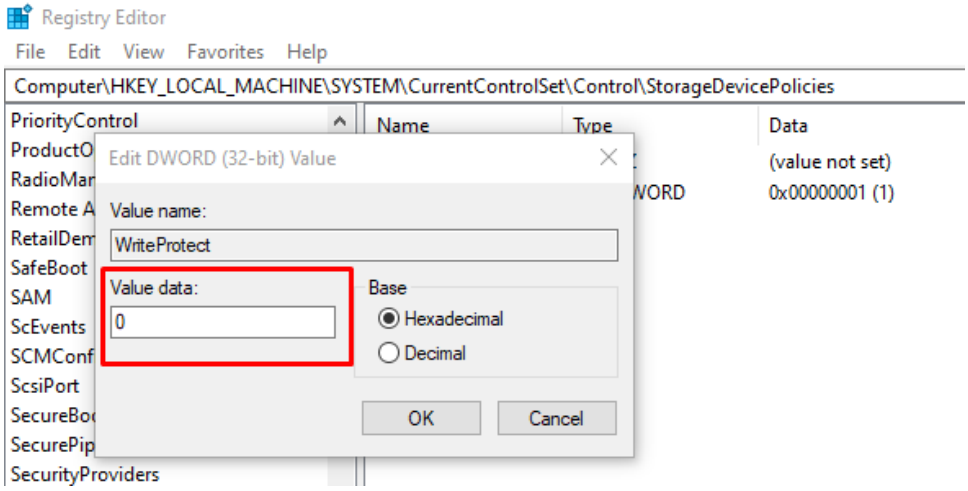

Figure 2.20 — Initial Chain-of-Custody document filled in after the first (FAT32) acquisition.

3. Days 48–49 — File System Analysis

3.1 Step 11 — Alternate Data Streams (ADS) Demonstration

The ADS demonstration was performed on the Windows host in a regular directory (C:\Users\abdul\Desktop\ADS_Test), not against the forensic image. A visible text file was created, and a hidden Alternate Data Stream was attached to that same filename.

3.1.1 Creating the ADS

```
cd C:\Users\abdul\Desktop
mkdir ADS_Test
cd ADS_Test
echo This is the visible content > visible_file.txt
notepad visible_file.txt:hidden_stream.txt
```

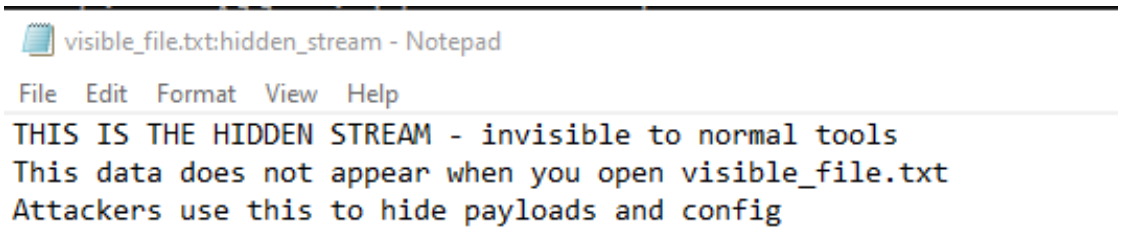

Figure 3.1 — Command Prompt: ADS_Test folder created and Notepad launched with the ADS syntax to write the hidden stream.

The following text was written into the hidden stream and saved:

```
THIS IS THE HIDDEN STREAM - invisible to normal tools
This data does not appear when you open visible_file.txt
Attackers use this to hide payloads and config
```

Field	Details
Case Number	CYB-W7-001
Evidence Item	USB Flash Drive
Evidence Number	E001
Description	USB drive containing evidence files for Week 7 forensics exercise
Manufacturer/Model	Generic Flash Disk USB Device/0
Date of Acquisition	4/21/2026
Time of Acquisition	03:09:41
Examiner Name	Abdul Muqeet Tabraiz
Examiner Organization	Cyberster Blue Team Internship
Harware Used	Windows 10
Write-Blocke Method	Software — Windows Registry WriteProtect = 1 (limitation: not hardware blocker)
Image Format	E01 (EnCase format)
Image File Path	E:\Cyberster Internship\Week 7\ForensicLab Images\W7_Evidence_USB.E01
MD5 Hash (Source)	8069e499cb22190ca556a4b08f4443d2
MD5 Hash (Image)	8069e499cb22190ca556a4b08f4443d2
SHA1 Hash (Source)	d0fb78909cd989d99fbb4bf2ee61c9c6e1dcccf7
SHA1 Hash (Image)	d0fb78909cd989d99fbb4bf2ee61c9c6e1dcccf7
Hash Match Confirmed	YES- both MD5 and SHA1 hashes match
Storage Location	E:\Cyberster Internship\Week 7\ForensicLab Images

Figure 3.2 — Notepad writing the hidden stream content into visible_file.txt:hidden_stream.txt.

3.1.2 Proving the Stream Is Hidden

Opening visible_file.txt normally shows only the visible content — the hidden stream is completely absent from the default Notepad view:

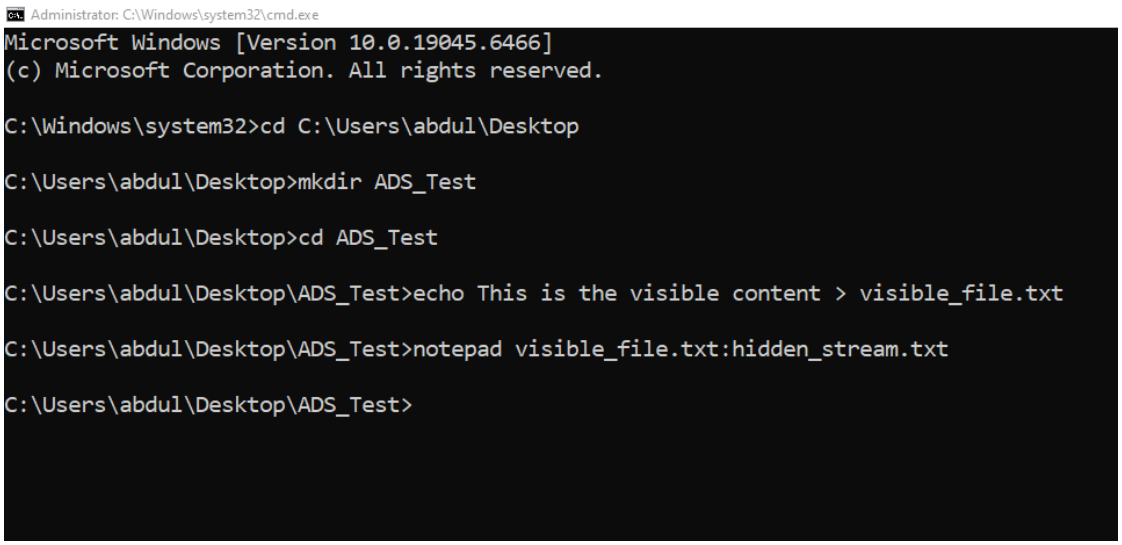

Figure 3.3 — Notepad showing only the visible content of visible_file.txt — hidden stream is not visible.

The default dir listing also gives no indication of the hidden stream; only the main file (30 bytes) is reported:

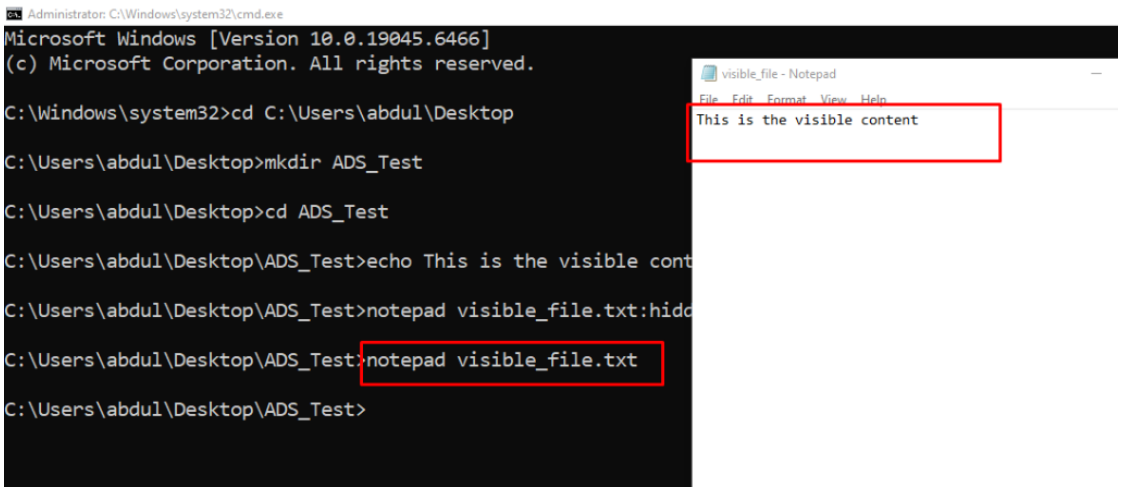

Figure 3.4 — dir output for ADS_Test — only visible_file.txt (30 bytes) is listed; the ADS is invisible.

3.1.3 Detecting the ADS with dir /r

The /r switch reveals Alternate Data Streams. The hidden 159-byte stream visible_file.txt:hidden_stream.txt:$DATA is now disclosed:

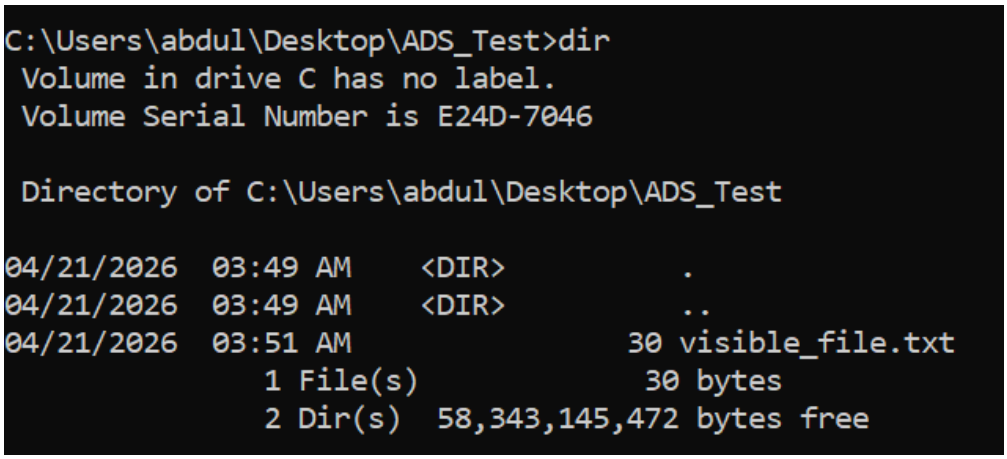

Figure 3.5 — dir /r output — the ADS visible_file.txt:hidden_stream.txt:$DATA (159 bytes) is clearly listed.

3.1.4 Accessing the Hidden Stream Directly

The hidden stream content can be read back directly by referencing the stream path in Notepad — confirming that the data not only exists but is fully retrievable by an attacker who knows the stream name.

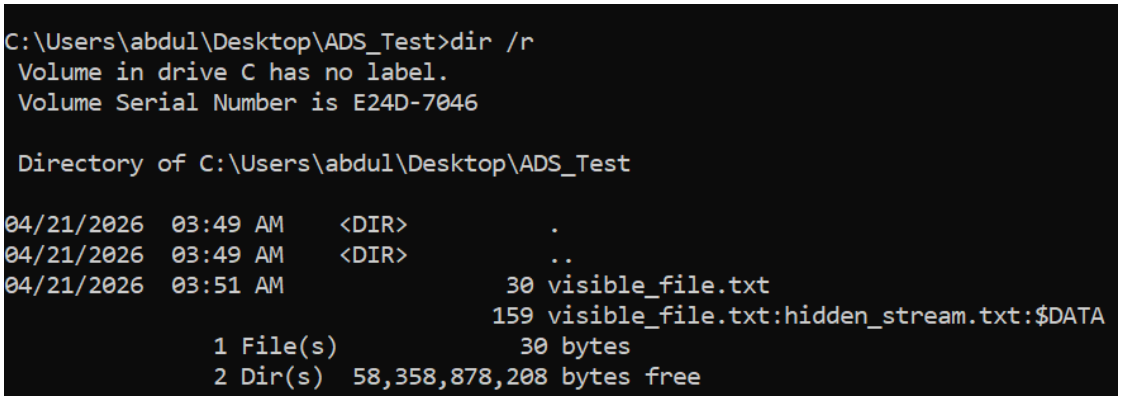

Figure 3.6 — Notepad showing the retrieved content of the hidden stream when opened directly.

3.1.5 Confirmation with Sysinternals Streams

Mark Russinovich's Sysinternals streams.exe v1.60 was placed in the test folder and run against visible_file.txt. The tool independently confirmed the presence of the hidden :hidden_stream.txt:$DATA stream and its size (159 bytes).

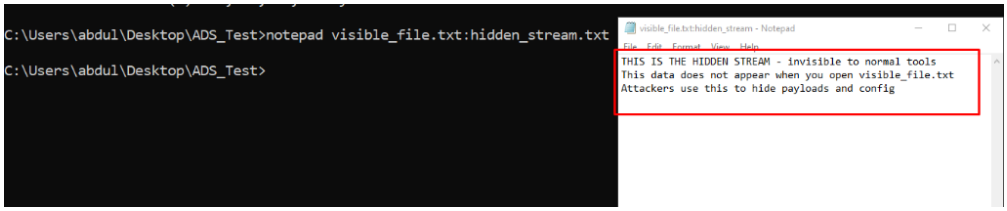

Figure 3.7 — streams.exe copied alongside visible_file.txt in the ADS_Test folder.

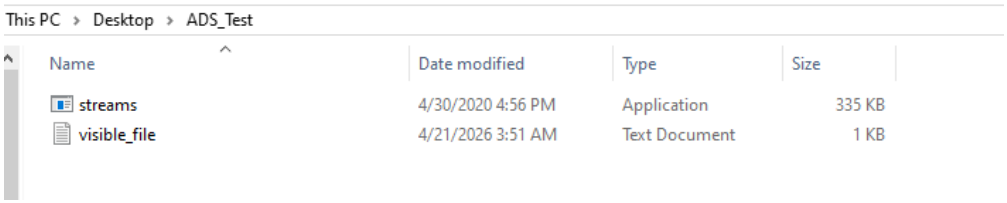

Figure 3.8 — streams.exe output reporting :hidden_stream.txt:$DATA (159 bytes) on visible_file.txt.

3.2 Step 12 — Legitimate ADS Example: Zone.Identifier

Alternate Data Streams are not only used for malicious concealment — Windows itself uses an ADS called Zone.Identifier to mark files downloaded from the internet (the source of the "This file came from another computer" security warning). A downloaded file (testing.doc, 100,352 bytes) in the Downloads folder was examined with dir /r, revealing the Zone.Identifier:$DATA stream (211 bytes).

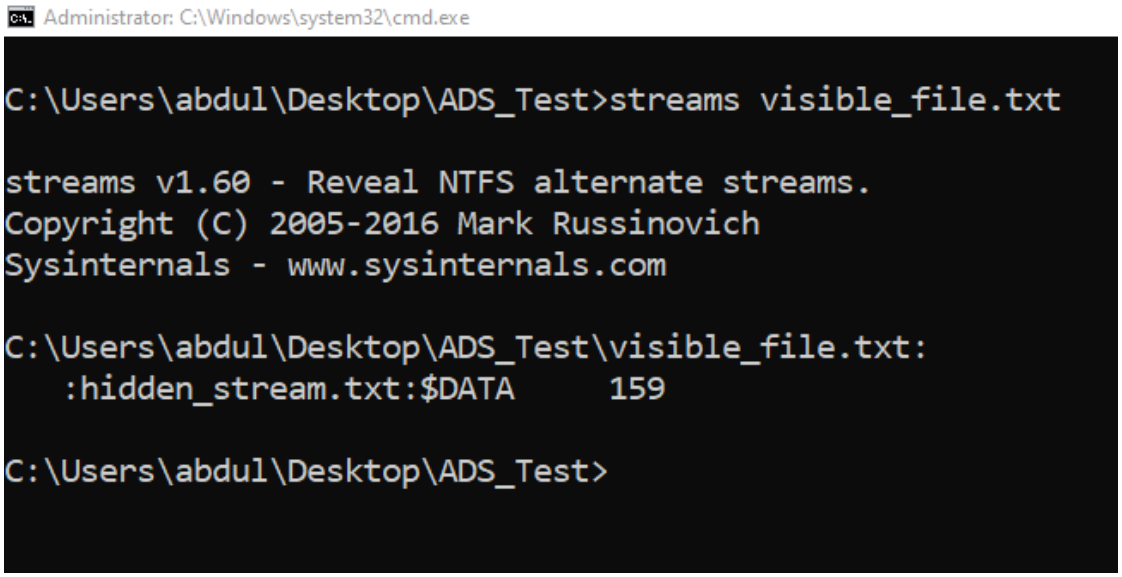

Figure 3.9 — dir /r on a downloaded file showing the legitimate Zone.Identifier:$DATA ADS — 211 bytes.

3.3 Step 13 — Reformat USB to NTFS and Re-image

Why Re-image? FAT32 → NTFS

During $MFT extraction on the FAT32 image it became clear that FAT32 does not contain an MFT — it uses a File Allocation Table instead, and the $MFT structure is NTFS-only.

To allow full MFT analysis the USB was reformatted to NTFS (volume label EVIDENCE_01) and re-imaged. The original FAT32 image is retained as process evidence; the NTFS image is the authoritative artefact used for Sections 3–5.

A full format (Quick Format disabled) was performed to ensure slack space was overwritten and a clean NTFS volume was produced.

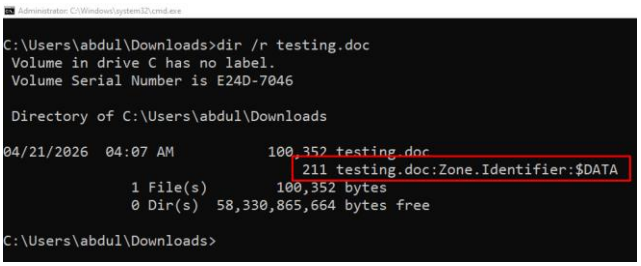

Figure 3.10 — Format dialog: NTFS file system, Allocation unit 4096 bytes, Volume label EVIDENCE_01 — full format in progress.

The NTFS drive was populated with an evidence layout containing three folders and several test files:

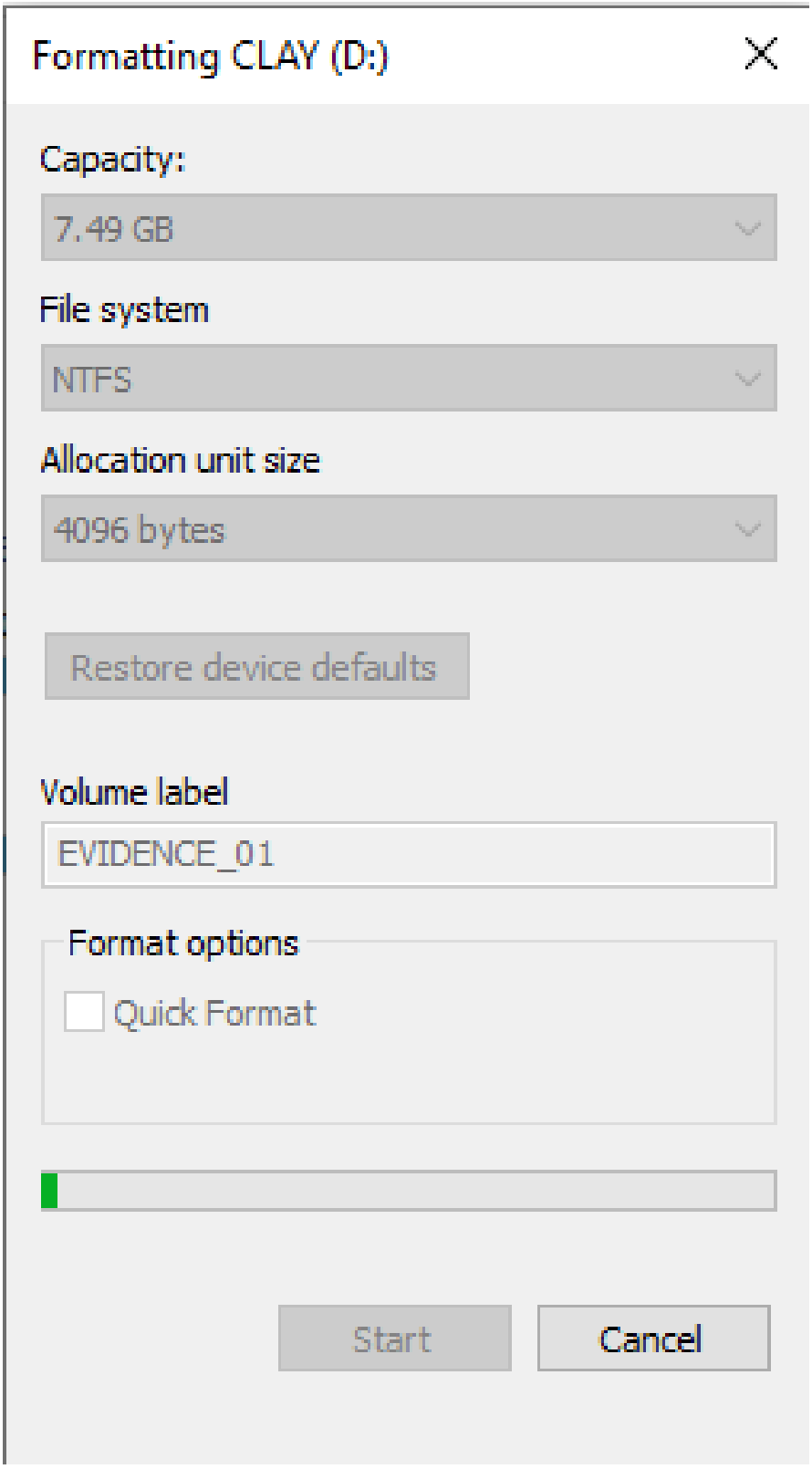

Figure 3.11 — EVIDENCE_01 (D:) root showing the three test folders: evidence_test, personal, work_files.

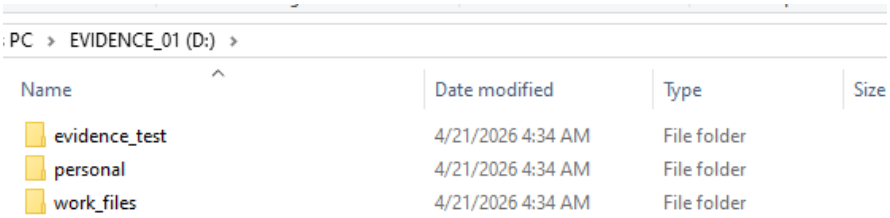

Figure 3.12 — evidence_test folder contents: image.jpg and notes.txt (report.docx and diary.txt later deleted as recoverable artefacts).

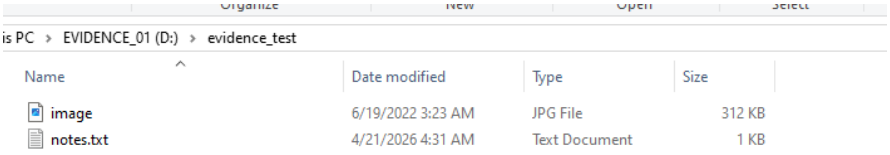

Figure 3.13 — personal folder shown empty in Explorer — diary.txt was deleted, producing a recoverable artefact.

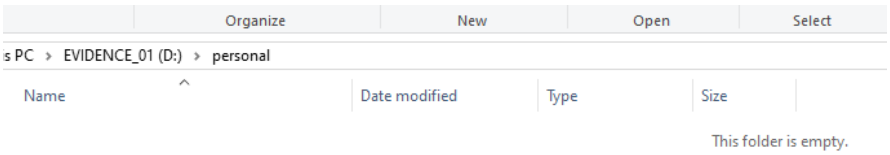

Figure 3.14 — work_files folder containing spreadsheet.xlsx.

3.3.1 Re-imaging the NTFS Volume

Steps 3–7 from Section 2 were repeated against the reformatted USB to produce the authoritative NTFS image W7_Evidence_USB_NTFS.E01. The Evidence Item Information was updated accordingly.

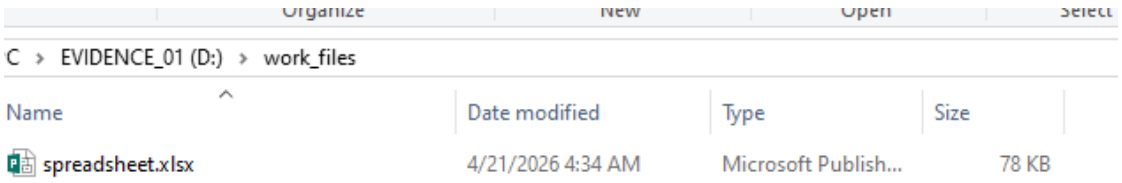

Figure 3.15 — FTK Imager source selection (NTFS USB).

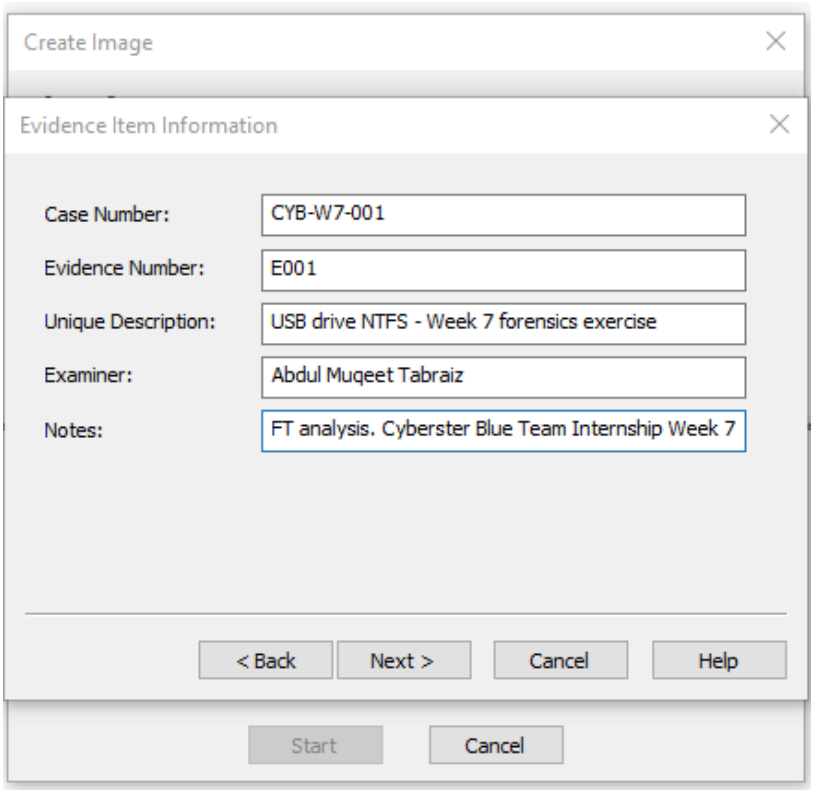

Figure 3.16 — Destination filename updated to W7_Evidence_USB_NTFS for the NTFS acquisition.

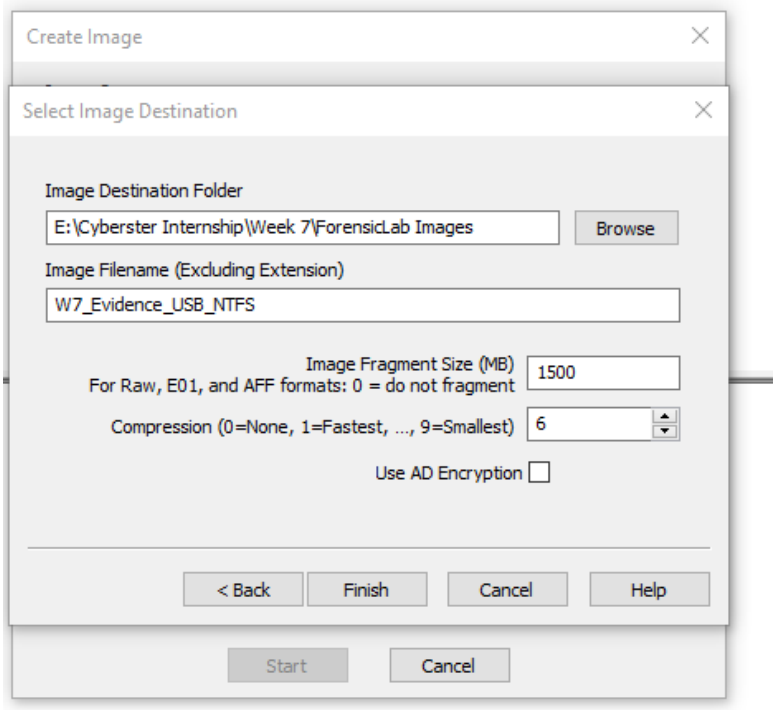

Figure 3.17 — NTFS acquisition in progress from \\.\PHYSICALDRIVE2.

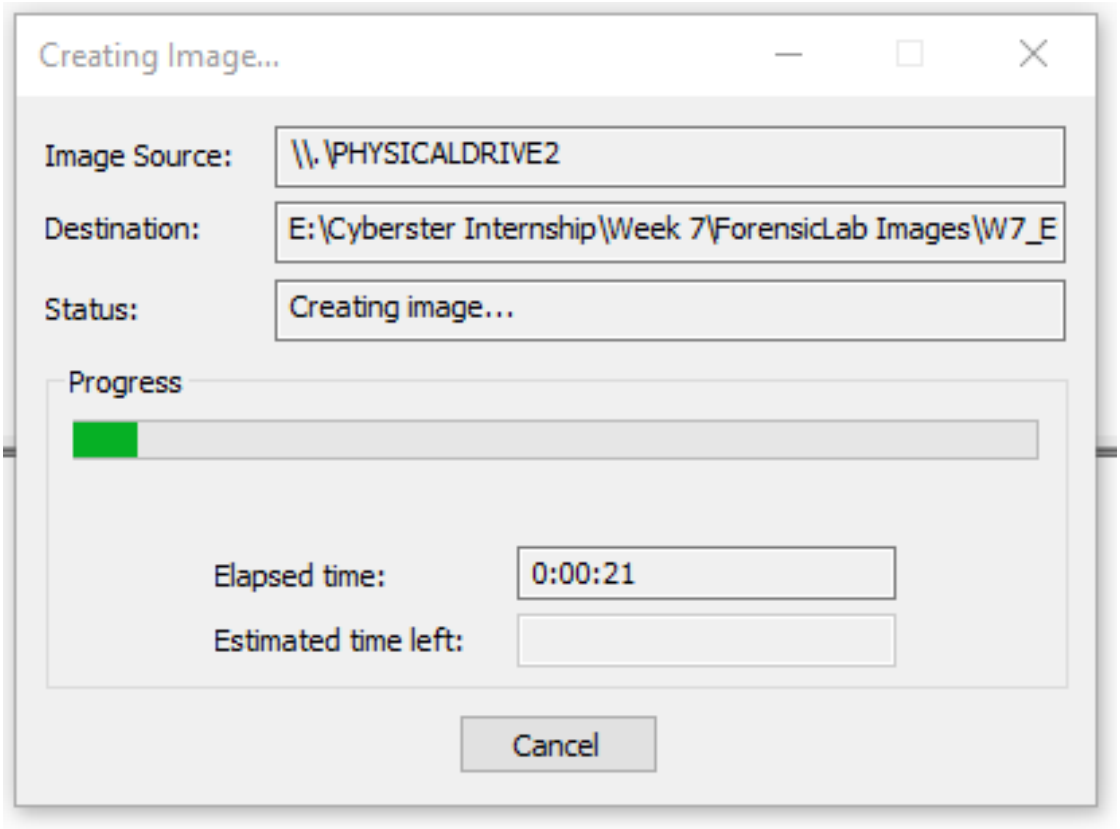

Figure 3.18 — Drive/Image Verify Results for the NTFS image — MD5 and SHA-1 both Match. This is the primary integrity deliverable.

Authoritative NTFS Acquisition Hashes

Algorithm	Source Drive	E01 Image	Verify
MD5	1127cba3f65d7025b963ab92e942f299	1127cba3f65d7025b963ab92e942f299	Match
SHA-1	72b809fa1dd637395b1b0807f480cfc11fdbe35e	72b809fa1dd637395b1b0807f480cfc11fdbe35e	Match

Figure 3.19 — FTK Imager Image Summary for the NTFS image — sector count 15,728,640; acquisition started 04:38:12 and verified 04:43:53 on 21 Apr 2026.

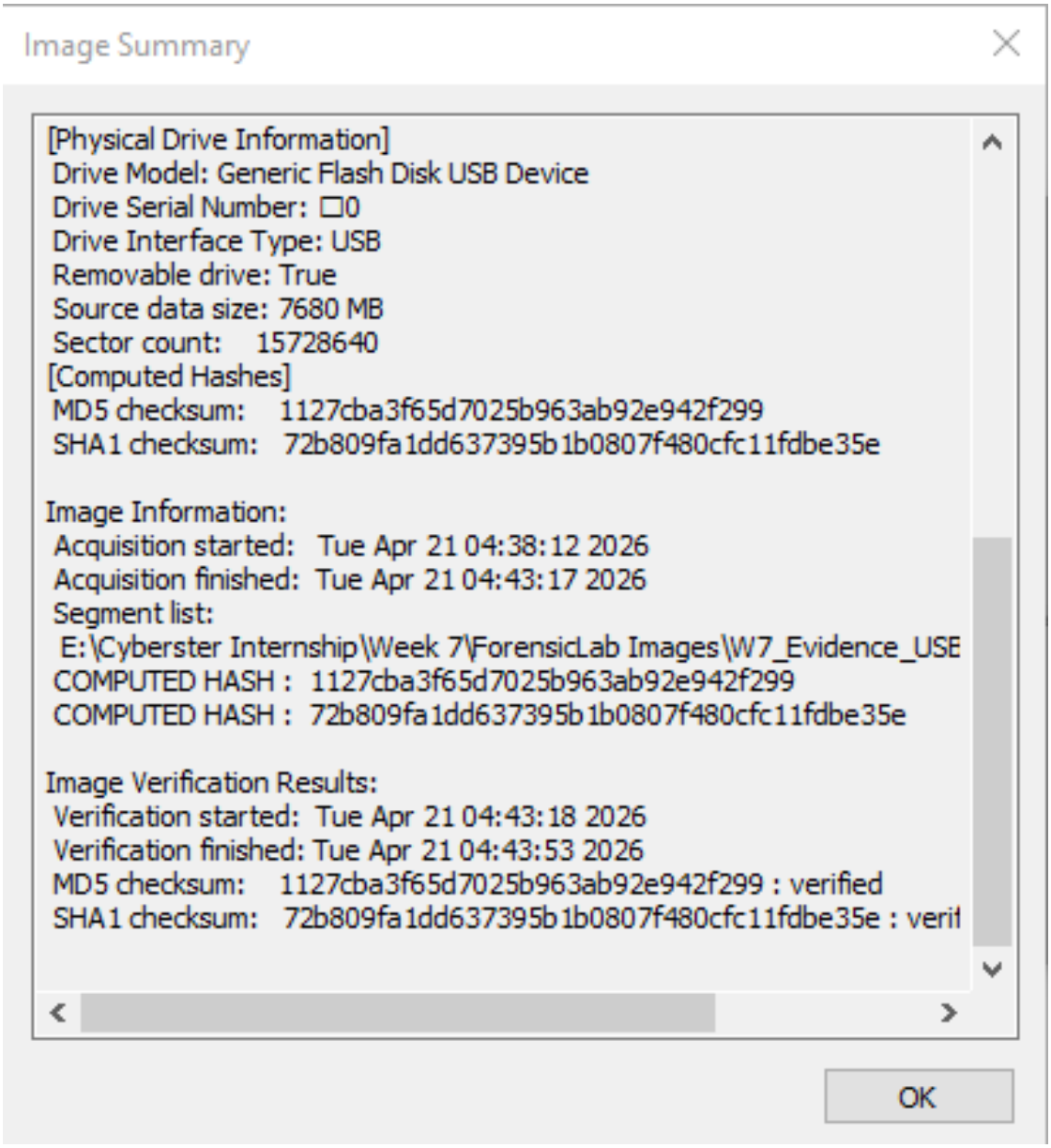

Figure 3.20 — Arsenal Image Mounter confirming W7_Evidence_USB_NTFS.E01 mounted read-only.

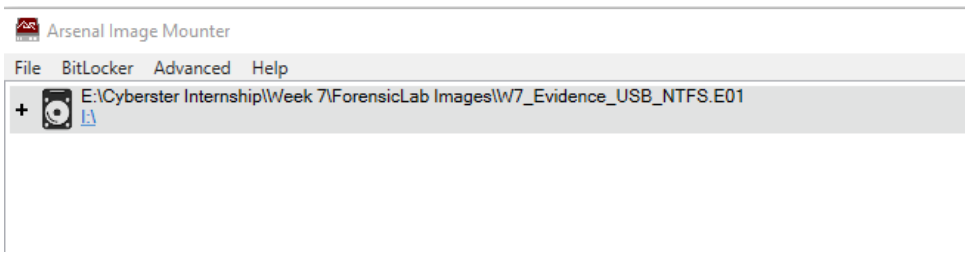

Figure 3.21 — Updated Chain-of-Custody document containing the final NTFS image metadata and hash values. Full copy reproduced in Appendix A.

3.4 Step 14 — Extract $MFT from the NTFS Image

FTK Imager was used to add W7_Evidence_USB_NTFS.E01 as an Image File evidence item. The evidence tree was expanded to Partition 1 [NTFS] → [root], and the NTFS system files starting with "$" became visible. $MFT (262,144 bytes — 256 KB) was selected.

Field	Details
Case Number	CYB-W7-001
Evidence Item	USB Flash Drive
Evidence Number	E001
Description	USB Flash Drive — NTFS formatted, EVIDENCE_01 volume label
Manufacturer/Model	Generic Flash Disk USB Device/0
Date of Acquisition	4/21/2026
Time of Acquisition	03:09:41
Examiner Name	Abdul Muqeet Tabraiz
Examiner Organization	Cyberster Blue Team Internship
Harware Used	Windows 10
Write-Blocke Method	Software — Windows Registry WriteProtect = 1 (limitation: not hardware blocker)
Image Format	E01 (EnCase format)
Image File Path	E:\Cyberster Internship\Week 7\ForensicLab Images\W7_Evidence_USB_NTFS.E01
MD5 Hash (Source)	1127cba3f65d7025b963ab92e942f299
MD5 Hash (Image)	1127cba3f65d7025b963ab92e942f299
SHA1 Hash (Source)	72b809fa1dd637395b1b0807f480cfc11fdbe35e
SHA1 Hash (Image)	72b809fa1dd637395b1b0807f480cfc11fdbe35e
Hash Match Confirmed	YES- both MD5 and SHA1 hashes match
Storage Location	E:\Cyberster Internship\Week 7\ForensicLab Images

Figure 3.22 — FTK Imager evidence tree for the NTFS image showing $MFT at the root of Partition 1.

$MFT was exported to the local analysis folder (C:\ForensicsLab\MFT\).

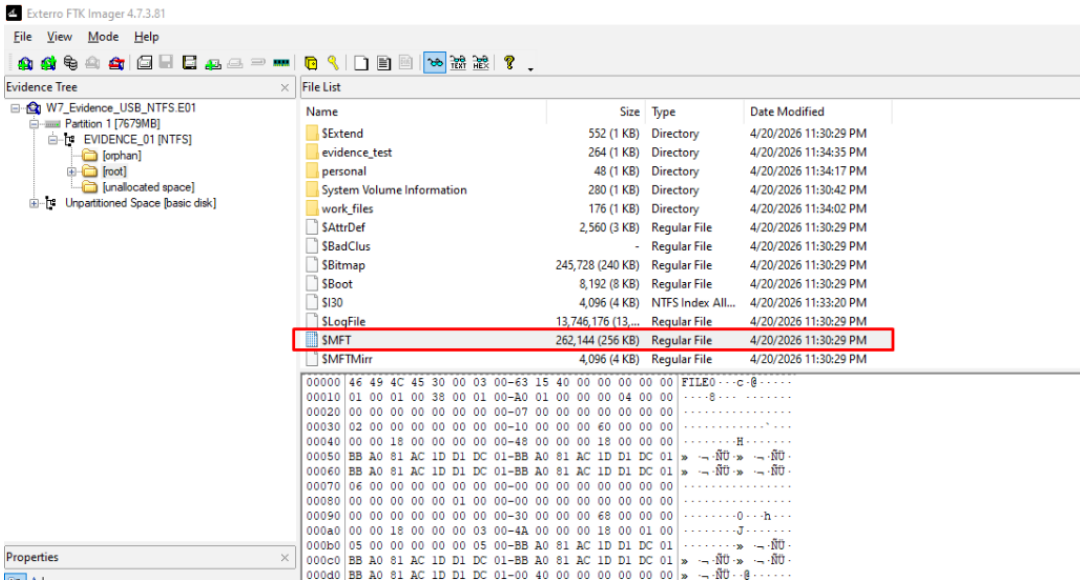

Figure 3.23 — FTK Imager export confirmation: $MFT exported successfully (262,144 bytes copied).

3.5 Step 15 — Analyse $MFT in MFT Explorer

The exported $MFT was loaded in Eric Zimmerman's MFT Explorer v2.1.0. The left pane rendered the NTFS directory tree (evidence_test, personal, work_files, $Extend, System Volume Information). Each

file's MFT record exposes the full MACB timestamps (Modified, Accessed, Changed, Birth) from both the $STANDARD_INFORMATION and $FILE_NAME attributes.

3.5.1 Record 1 — image.jpg (evidence_test)

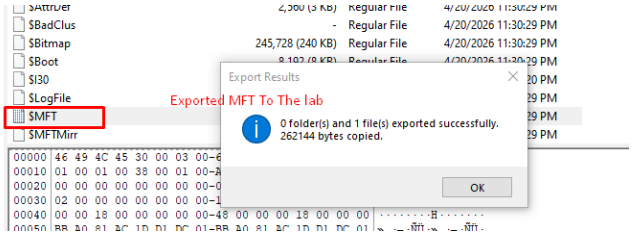

Figure 3.24 — MFT Explorer — full MFT record for image.jpg showing STANDARD_INFO and FILE_NAME attributes, non-resident DATA runs, and MACB timestamps.

3.5.2 Record 2 — notes.txt (evidence_test)

Figure 3.25 — MFT Explorer — MFT record for notes.txt showing STANDARD_INFO, FILE_NAME, ObjectID attributes and full MACB values.

The volume EVIDENCE_01 was also verified as NTFS via Windows (7.47 GB free of 7.49 GB), confirming the target for MFT analysis.

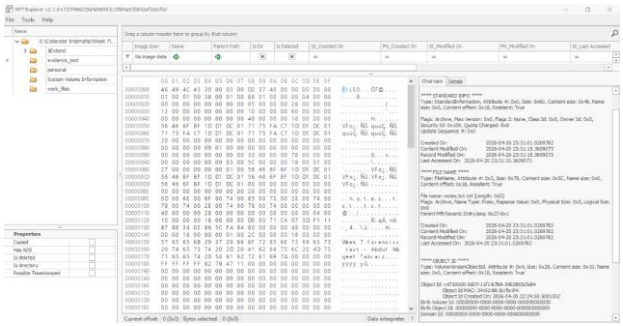

Figure 3.26 — EVIDENCE_01 (I:) drive properties after mount — confirms the analysed volume.

3.5.3 Deleted File Still Present in the MFT

The deleted file diary.txt (originally in .\personal) remains fully represented in the MFT with its Is Deleted flag set. This directly demonstrates the NTFS deletion model: the record is marked as available but the metadata (timestamps, size, attributes) is preserved until the MFT record is overwritten — which is why deleted files are so often recoverable.

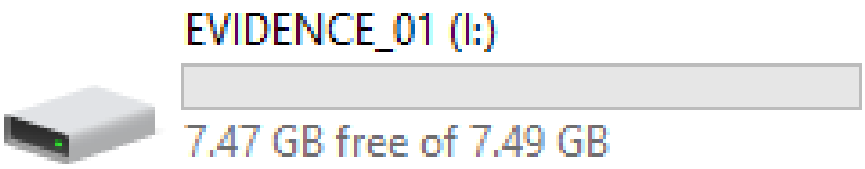

Figure 3.27 — MFT Explorer showing diary.txt in .\personal with the Is Deleted flag set. MFT retains full MACB timestamps for forensic review.

3.6 Step 16 — MACB Timestamps Experiment

A five-stage experiment was conducted on the Windows host in C:\TimestampTest\ to observe MACB timestamp behaviour across common file operations. PowerShell's Get-Item was used to read CreationTime, LastWriteTime and LastAccessTime after each operation.

Stage A — File Creation

```
mkdir C:\TimestampTest
cd C:\TimestampTest
echo "Original content" > test_file.txt
powershell "Get-Item test_file.txt | Select-Object Name, CreationTime, LastWriteTime,
LastAccessTime"
```

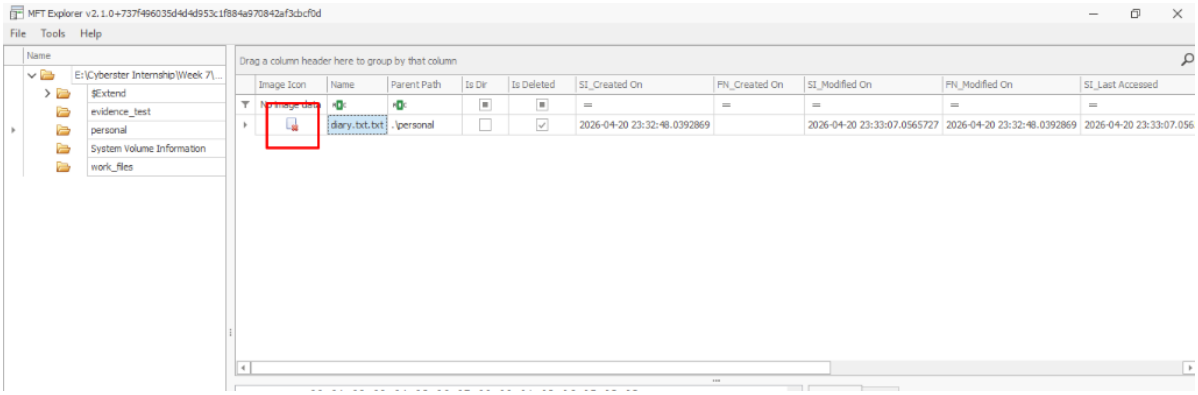

Figure 3.28 — Stage A — test_file.txt immediately after creation: all three timestamps equal the creation time (05:04:30).

Stage B — File Opened (no changes)

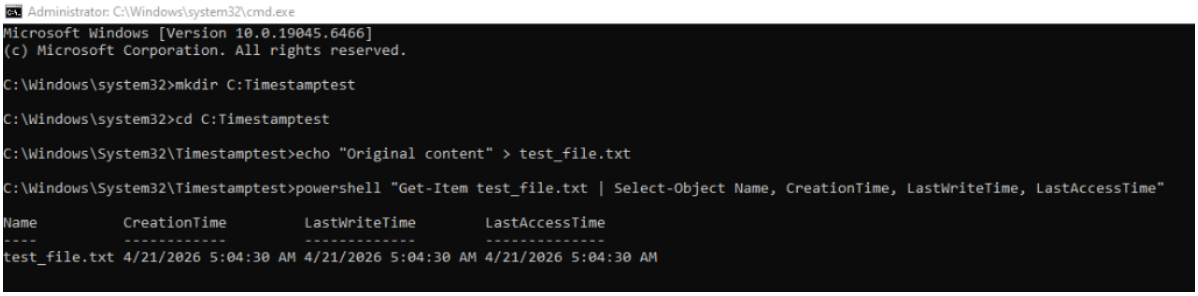

Figure 3.29 — Stage B — after Notepad open/close with no edits: LastAccessTime updated (05:05:17); CreationTime and LastWriteTime unchanged.

Stage C — File Modified

```
echo "Added content" >> test_file.txt
```

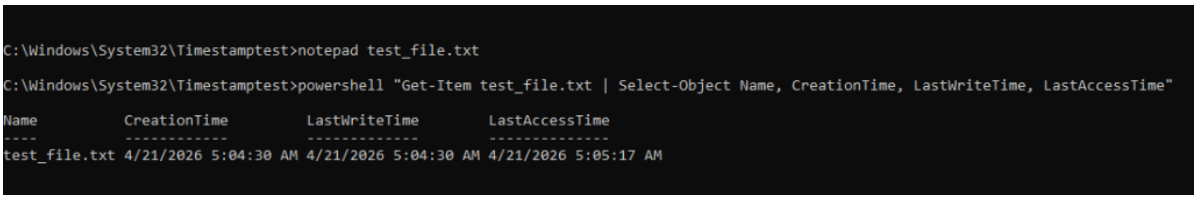

Figure 3.30 — Stage C — after append: LastWriteTime and LastAccessTime both updated to 05:06:32; CreationTime unchanged.

Stage D — File Copied (to new location)

```
copy test_file.txt C:\TimestampTest\test_copy.txt
```

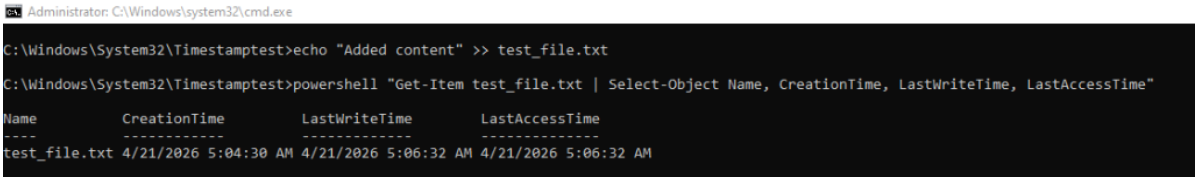

Figure 3.31 — Stage D — the copy's CreationTime is the copy time (05:10:12); LastWriteTime is preserved from the source (05:06:32).

Stage E — File Moved (same volume)

```
mkdir C:\TimestampTest\Subfolder
move test_file.txt C:\TimestampTest\Subfolder\
```

Figure 3.32 — Stage E — after move on the same volume: CreationTime and LastWriteTime preserved, LastAccessTime updated.

3.6.1 Summary of MACB Behaviour

Operation	Modified (M)	Accessed (A)	MFT Changed (C)	Born / Created (B)
File created	Set to creation time	Set to creation time	Set to creation time	Set to creation time
File opened (no change)	No change	Updated	No change	No change
File modified	Updated	Updated	Updated	No change
File copied (new location)	Preserved from source	Updated	Updated	Set to copy time
File moved (same volume)	No change	No change	Updated	No change

3.7 Step 17 — File System Comparison (NTFS vs FAT16/FAT32 vs EXT4)

3.7.1 NTFS — New Technology File System

- Used on: all modern Windows systems (Windows XP and later).
- Metadata structure: Master File Table ($MFT) — one ~1 KB record per file and directory.
- Key metadata files: $MFT (file database), $LogFile (journal for crash recovery), $Bitmap (cluster allocation), $UsnJrnl (change journal).
- Deletion behaviour: MFT record marked as available and clusters freed in $Bitmap — record remains readable until overwritten.
- Forensic artefacts: deleted MFT records, file slack, Alternate Data Streams, MACB timestamps, $UsnJrnl change history, $LogFile transactions.
- Preferred tools: MFT Explorer, Autopsy, FTK Imager.

3.7.2 FAT16 / FAT32 — File Allocation Table

- Used on: USB flash drives, SD cards, legacy partitions, embedded systems.
- Metadata structure: 32-byte directory entries plus the File Allocation Table (cluster chain).
- Deletion behaviour: first byte of the directory entry is changed to 0xE5; clusters are marked free in the FAT. File data remains on disk until overwritten.
- Forensic artefacts: deleted directory entries marked 0xE5, orphaned cluster chains, recoverable raw file data (file-carving is highly effective).
- Preferred tools: Autopsy, Recuva, PhotoRec.
- No MFT — which is why the initial FAT32 image could not be used for Section 3.4's $MFT analysis.

3.7.3 EXT4 — Fourth Extended Filesystem (Linux)

- Used on: Linux distributions (Ubuntu, Kali, Debian, etc.).
- Metadata structure: inodes — each file is described by an inode holding permissions, timestamps, size and data-block pointers. Directory entries map names to inode numbers.
- Deletion behaviour: the inode is marked free, the directory entry is removed, and block pointers are zeroed — more destructive than NTFS/FAT deletion.
- Forensic artefacts: journal (recent transactions), inode table, orphaned inodes, unallocated block groups.
- Preferred tools: Autopsy (via The Sleuth Kit), ext4magic, debugfs.

3.7.4 Side-by-Side Forensic Summary

Aspect	NTFS	FAT32	EXT4
Metadata	$MFT (1 KB records)	Directory entries + FAT	Inodes + directory entries
Journal	$LogFile, $UsnJrnl	None	Journal (ext4)
Deletion	Record flagged; data preserved	0xE5 marker; clusters freed	Pointers zeroed; harder to recover

ADS / streams	Native (multiple streams)	No	No (xattrs exist, different concept)
Timestamps	MACB (4 timestamps × 2 attributes)	3 (limited resolution)	MACB + nanosecond precision
Typical Windows tools	MFT Explorer, Autopsy	Autopsy, PhotoRec	Autopsy (Sleuth Kit), debugfs

4. Days 50–52 — Autopsy Case Setup and Initial Triage

4.1 Step 17 — Create a New Autopsy Case

Autopsy 4.22.1 was installed on the Windows host. A new single-user case was created with the following parameters.

Field	Value
Case Name	CYB_W7_USB_Investigation
Case Number	CYB-W7-001
Examiner	Abdul Muqeet Tabraiz
Base Directory	C:\ForensicsLab\AutopsyCases\
Notes	USB Week 7 Evidence
Case Type	Single-user

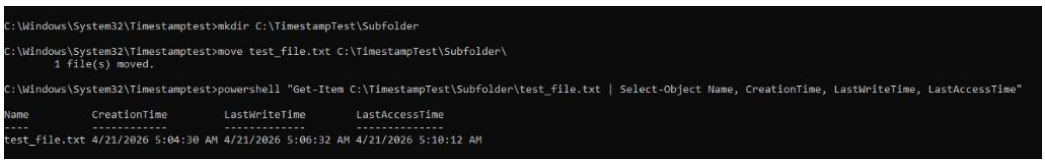

Figure 4.1 — Autopsy New Case wizard — Optional Information tab with case number, examiner and notes completed.

4.2 Step 18 — Add the E01 Image as Data Source

W7_Evidence_USB_NTFS.E01 was added as a Disk Image or VM File data source. Autopsy calculated its MD5/SHA-1 hashes; both matched the values recorded during FTK acquisition, independently confirming that the image had not been altered between tools.

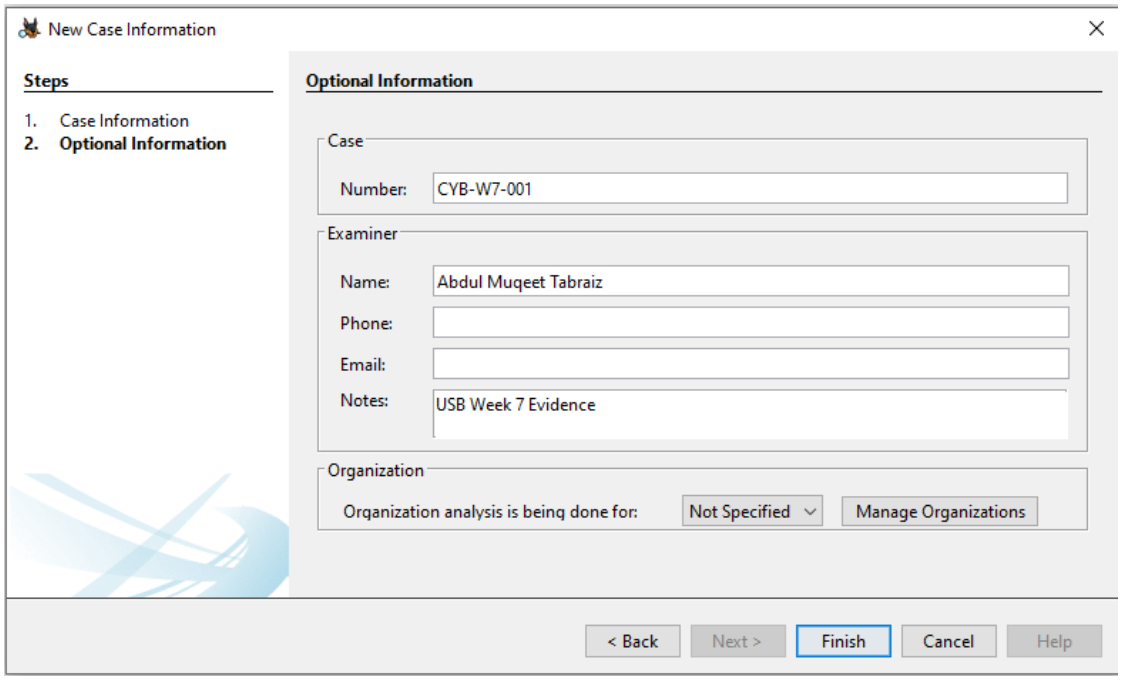

Figure 4.2 — Autopsy main case view after ingest: W7_Evidence_USB_NTFS.E01 loaded as the single data source.

4.3 Step 19 — Configure and Run Ingest Modules

The following ingest modules were enabled:

Module	Purpose
Hash Lookup	Match file hashes against known-good/bad sets.
File Type Identification	Identify file type by signature (not extension).
Keyword Search	Search all content for custom terms (Abdul Muqeet, forensics, evidence).
EXIF Parser	Extract metadata from image files.
Recent Activity	OS artefacts, browser history, recycle bin.
PhotoRec Carver	Carve deleted files by signature.
Extension Mismatch Detector	Flag files whose extension does not match their content.

4.4 Step 20 — Triage: File Tree Examination

Autopsy identified vol1 (Unallocated) and vol2 (NTFS / exFAT 0x07) — the latter being the EVIDENCE_01 volume. The tree was expanded to examine $OrphanFiles, $CarvedFiles, $Extend, $Unalloc and the three user folders.

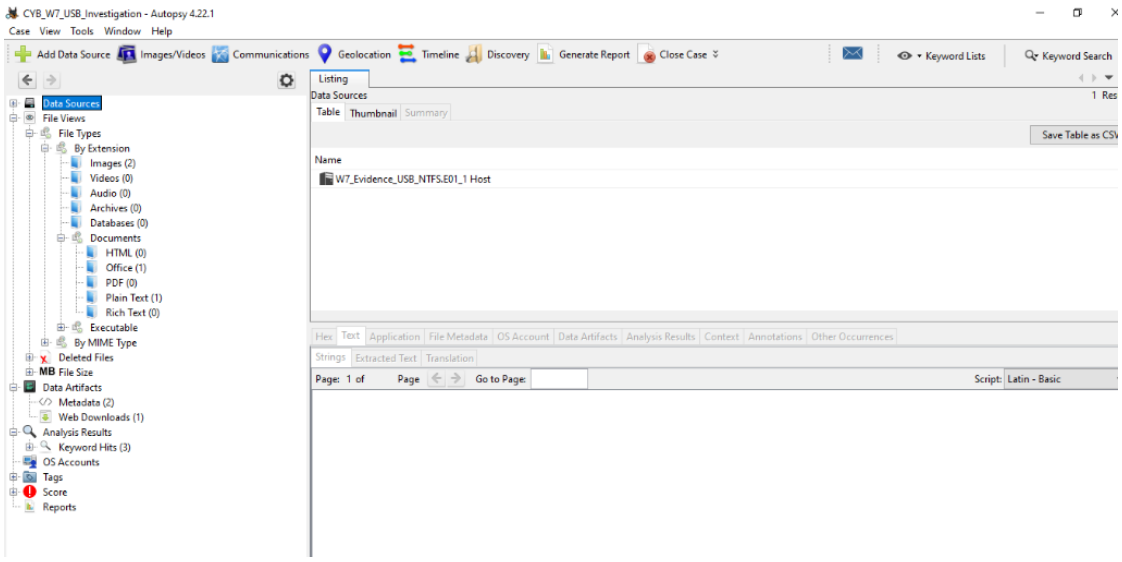

Figure 4.3 — Autopsy file tree: File Types, Images (2), Documents/Office (1), Plain Text (1), Web Downloads (1), Keyword Hits (3).

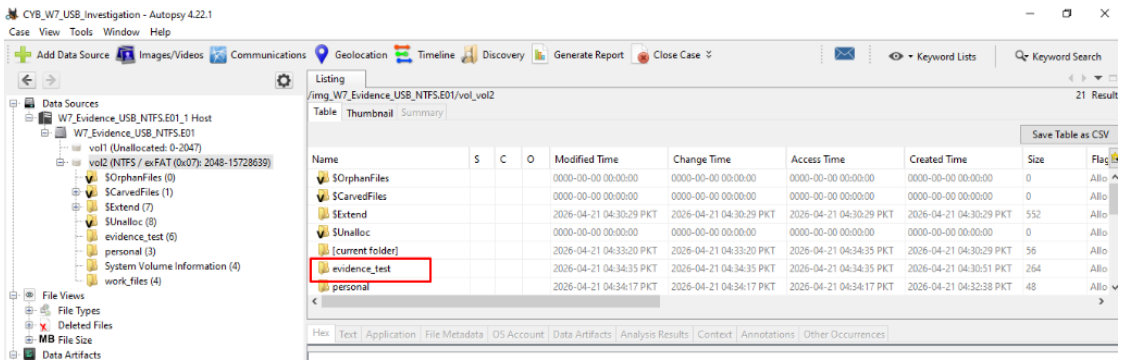

Figure 4.4 — /vol_vol2 contents — evidence_test, personal, work_files folders and NTFS system files visible.

4.4.1 evidence_test Folder

The evidence_test folder contained image.jpg (319,078 bytes), image.jpg:Zone.Identifier (140 bytes — evidence the image was downloaded from the internet), notes.txt, and the deleted file report.docx.docx (12,746 bytes, Unallocated). Autopsy's Extracted Text view successfully reconstructed the Word document content:

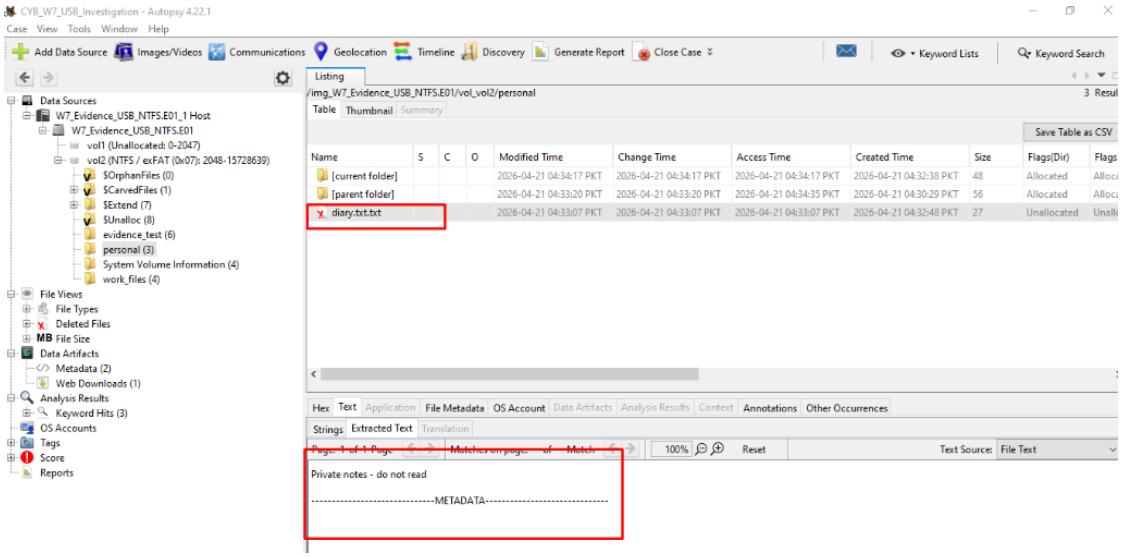

Figure 4.5 — evidence_test listing — report.docx.docx highlighted as an Unallocated (deleted) Word document; extracted text reads: "Hi there this is me Abdul Muqeet Tabraiz…"

4.4.2 personal Folder

The personal folder contained the deleted file diary.txt.txt (27 bytes, Unallocated). Autopsy's Extracted Text view recovered its content: "Private notes - do not read".

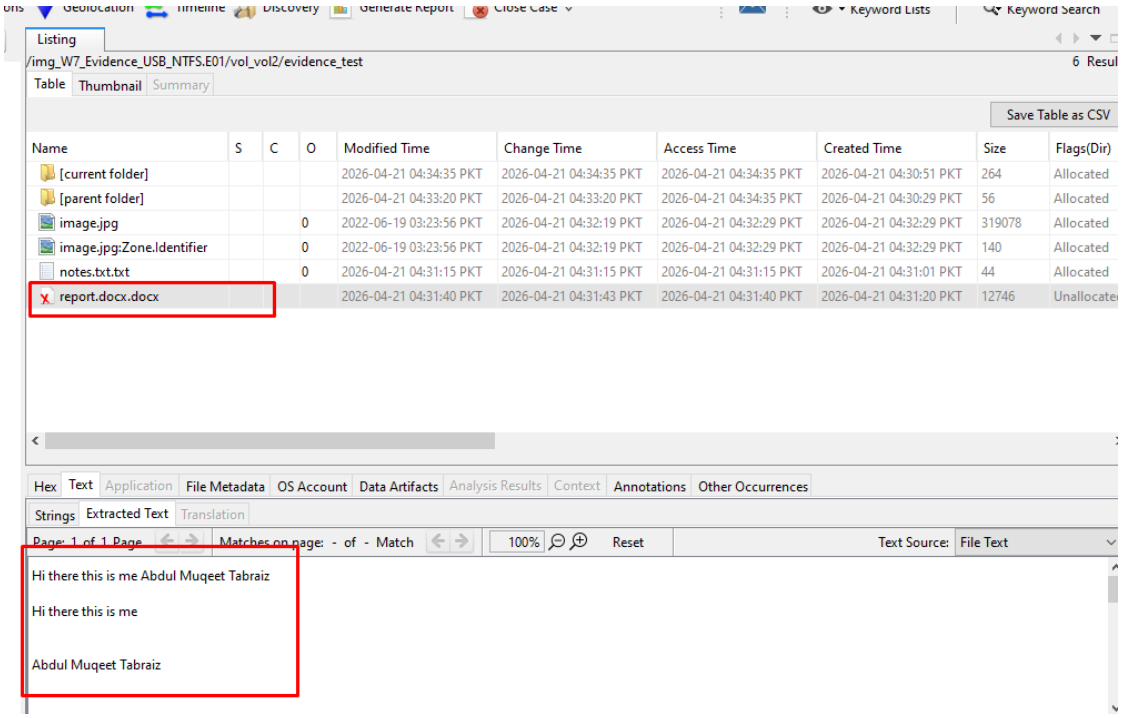

Figure 4.6 — diary.txt.txt deleted in /personal — content recovered via Autopsy's Extracted Text view.

4.4.3 work_files Folder

work_files contained two versions of a spreadsheet — the live spreadsheet.xlsx.pub (79,360 bytes, Allocated) and an Unallocated carved copy (59,904 bytes) flagged by the Extension Mismatch Detector (indicated by the yellow warning icon).

4.5 Step 22 — Deleted Files Recovery

Autopsy's Deleted Files node reported 5 deleted entries in total (3 File System deletions + 2 carved files).

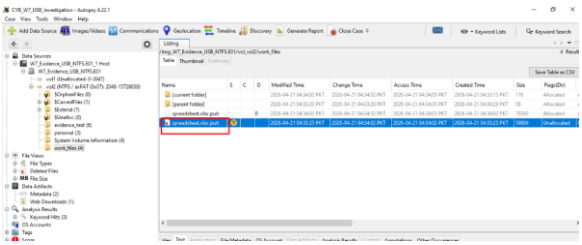

Figure 4.7 — Deleted Files totals: File System (3) and All (5).

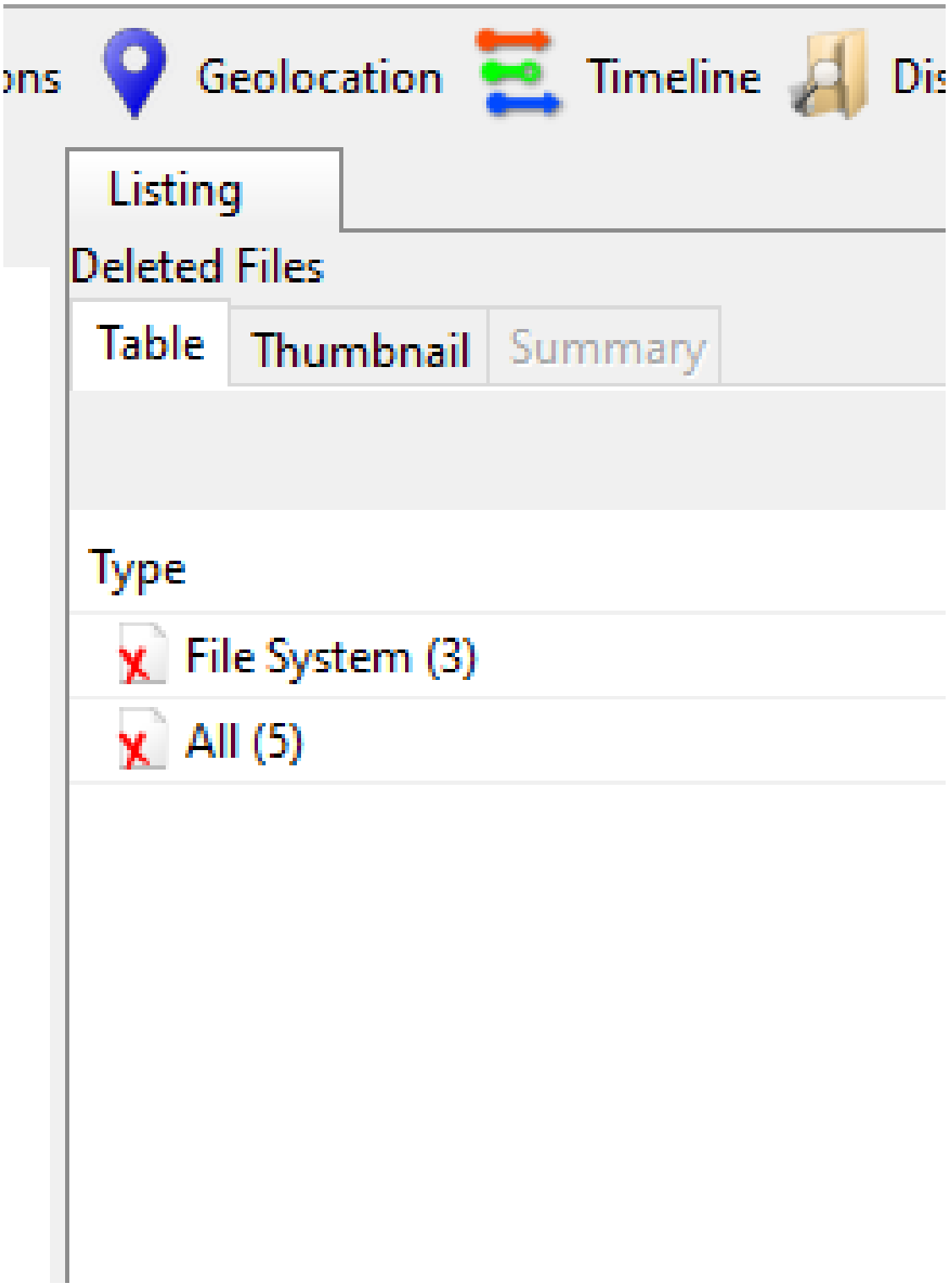

Figure 4.8 — Deleted files recovered: report.docx.docx, diary.txt.txt and spreadsheet.xlsx.pub (carved copy).

Right-click → Extract File was used to save the recovered files to C:\ForensicsLab\Evidence\Recovered\. Autopsy confirmed the extraction:

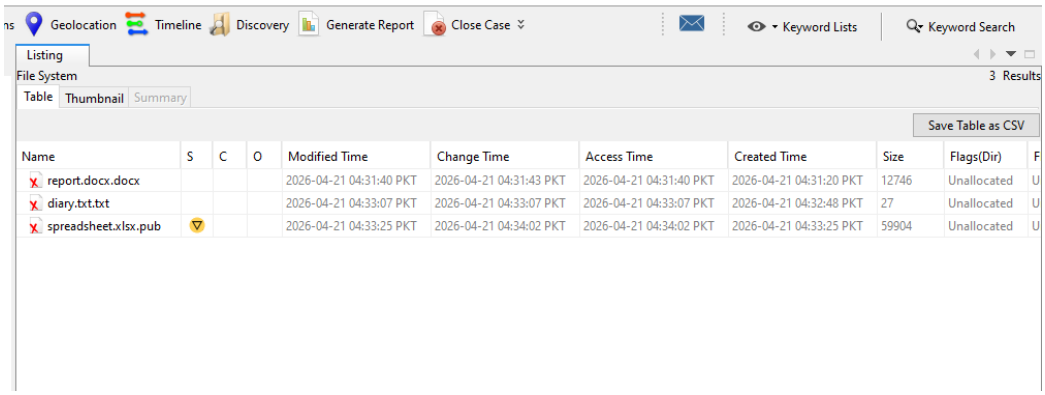

Figure 4.9 — Autopsy confirmation: File(s) extracted — recovered versions written to the analysis folder.

4.6 Step 23 — Timeline View

The Autopsy Timeline groups NTFS metadata events (Created/Modified/Last Saved etc.) across the case. Two dense clusters were visible: an earlier cluster (~June 2022) corresponding to the imported image.jpg's original creation metadata, and the 20-event cluster in April 2026 corresponding to the evidence-preparation activity performed for this exercise.

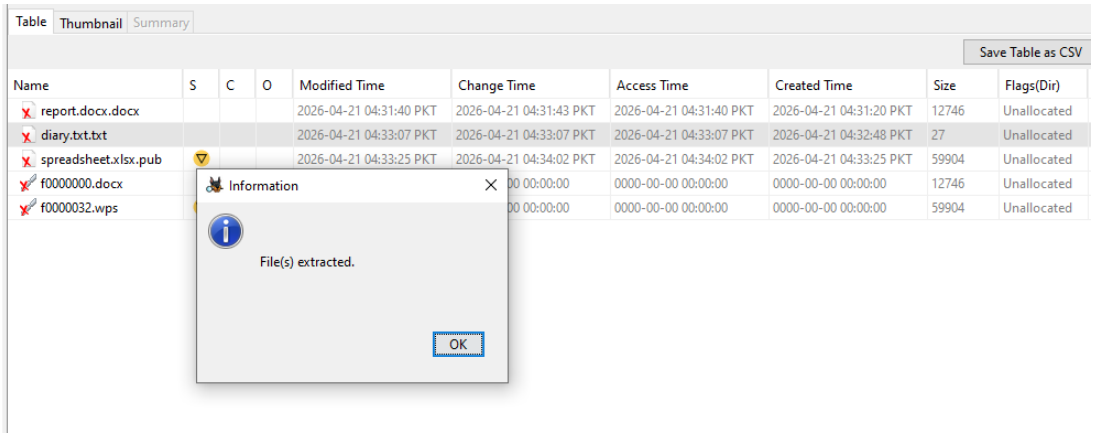

Figure 4.10 — Autopsy Timeline — 20 events on /evidence_test concentrated in April 2026, with NTFS metadata events ($LogFile, $BadClus, $UpCase, $Volume, $Boot, $Extend, $BadClus) itemised on the right.

4.7 Step 24 — Thumbnail View

Under File Types → Images, the Thumbnail view rendered the two image artefacts — image.jpg (the visible JPEG) and its Zone.Identifier ADS — side by side.

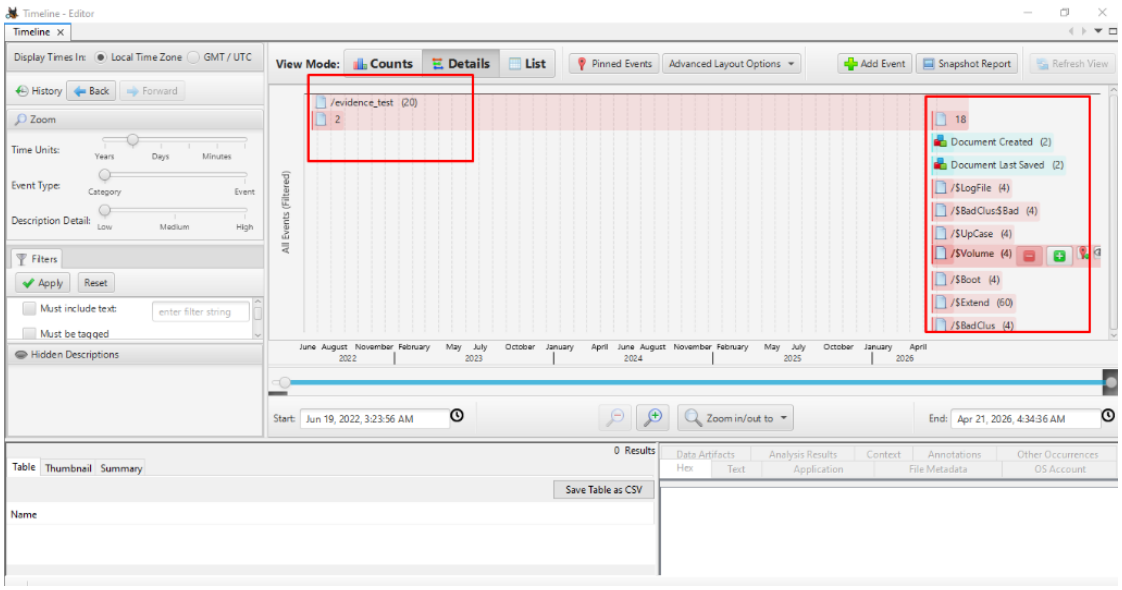

Figure 4.11 — Autopsy Thumbnail view — image.jpg and image.jpg:Zone.Identifier rendered from the NTFS image.

4.8 Step 25 — Generate HTML Report

Tools → Generate Report → HTML Report was executed. Autopsy produced the report at E:\Cyberster Internship\...\CYB_W7_USB_Investigation HTML Report 04-21-2026-05-34-56\report.html.

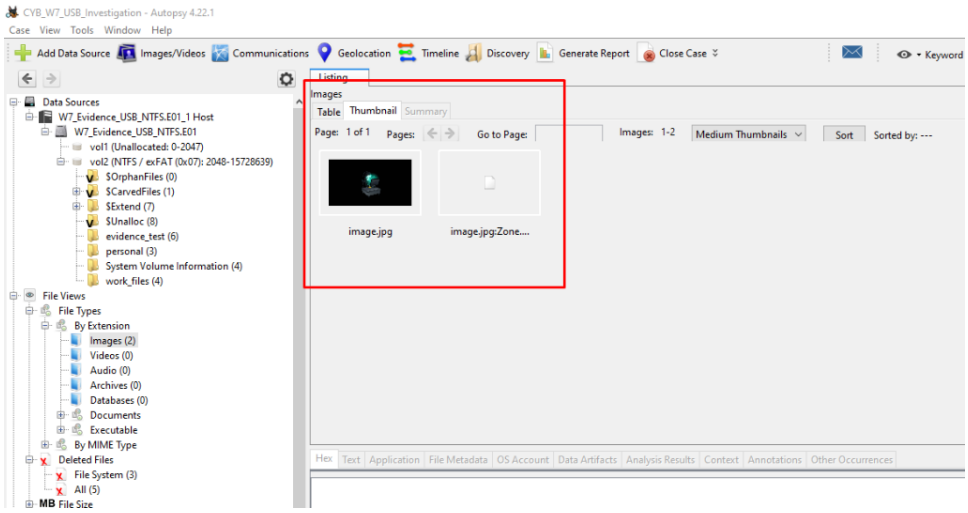

Figure 4.12 — Autopsy Report Generation Progress — Complete.

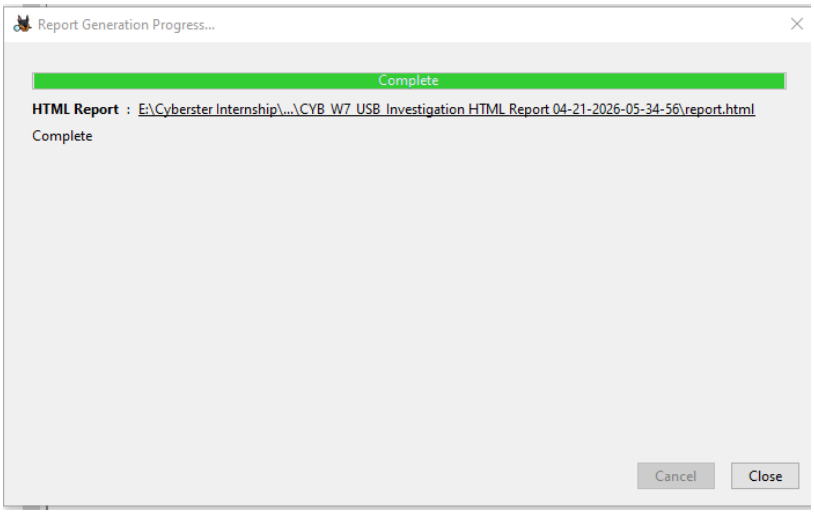

Figure 4.13 — Autopsy Forensic Report opened in browser — case header, examiner, notes and image information all rendered.

4.9 Web Download Artefact

The Recent Activity module flagged a Web Download artefact against /evidence_test/image.jpg: the source URL https://images.hdqwalls.com/wallpapers/skull-hacking-time-4k-rh.jpg with the Zone Identifier "Internet Zone". This is the legitimate Zone.Identifier ADS equivalent of the manual Section 3.2 demonstration, now surfaced automatically by Autopsy.

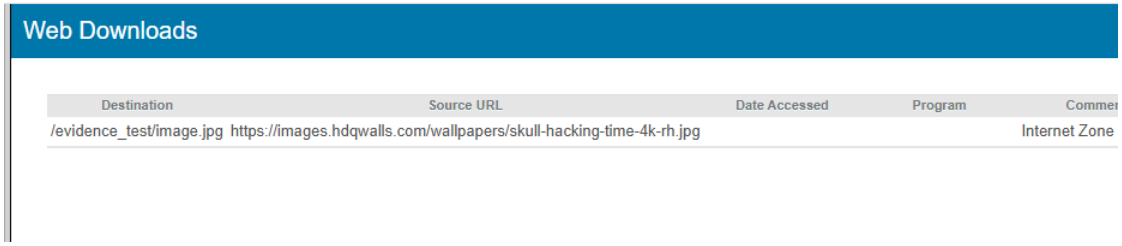

Figure 4.14 — Web Downloads — /evidence_test/image.jpg downloaded from hdqwalls.com, tagged Internet Zone.

4.10 Step 26 — Evidence Log

The significant findings from Autopsy triage are consolidated below.

#	File Path	Type	Size	Created	Modified	State	MD5 Hash	Significance
1	/evidence_test/image.jpg	JPEG	319,078 B	2026-04-21 04:32:29	2022-06-19 03:23:56	Allocated	(see Autopsy metadata)	Downloaded image — Zone.Identifier ADS present.

2	/evidence_test/image.jpg:Zone.Identifier	ADS/Text	140 B	2026-04-21 04:32:29	2022-06-19 03:23:56	Allocated	(NTFS ADS stream)	Confirms image was downloaded from Internet Zone.
3	/evidence_test/notes.txt	Text	44 B	2026-04-21 04:31:01	2026-04-21 04:31:15	Allocated	(see Autopsy metadata)	User-created evidence note.
4	/evidence_test/report.docx.docx	DOCX	12,746 B	2026-04-21 04:31:20	2026-04-21 04:31:40	Unallocated (recovered)	44290299530c9ec62107109097740000	Deleted Word document — recovered by Autopsy; text reconstructed.
5	/personal/diary.txt.txt	Text	27 B	2026-04-21 04:32:48	2026-04-21 04:33:07	Unallocated (recovered)	(see Autopsy metadata)	Deleted file — content recovered ("Private notes - do not read").
6	/work_files/spreadsheet.xlsx.pub	Spreadsheet	59,904 B	2026-04-21 04:33:25	2026-04-21 04:33:25	Unallocated (carved)	(see Autopsy metadata)	Carved duplicate flagged by Extension Mismatch Detector.

Example Autopsy file-metadata panel for report.docx.docx, showing MD5, SHA-256, MIME type and MACB:

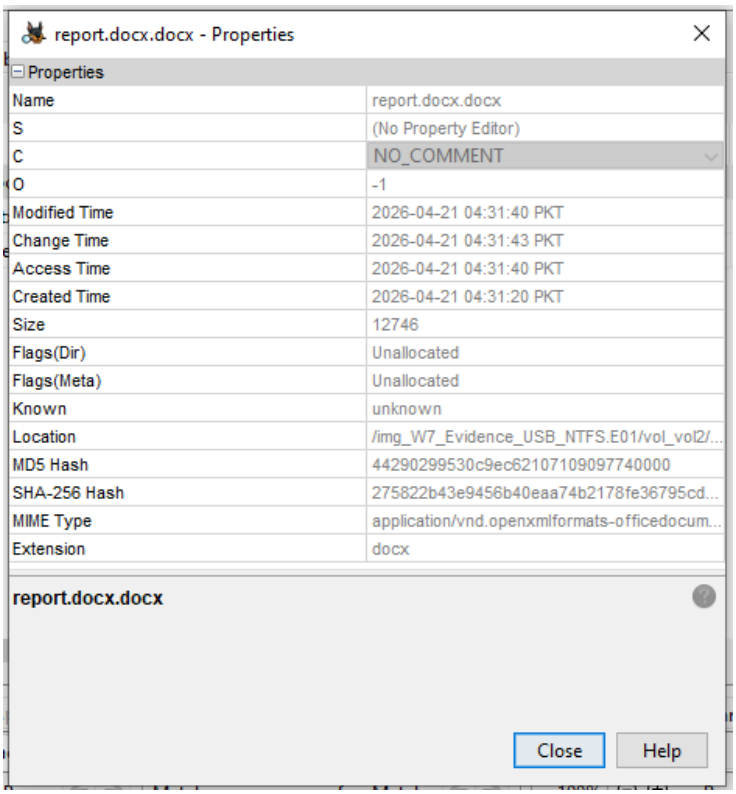

Figure 4.15 — Autopsy file properties for report.docx.docx — MD5 44290299530c9ec62107109097740000, SHA-256 275822b43e9456b40eaa74b2178fe36795cd…, MIME type application/vnd.openxmlformats-officedocument…, Flags Unallocated.

5. Key Concepts and Lessons Learned

5.1 Write-Blocking

Connecting a device for examination without a write-blocker permits the operating system to modify the device (update access timestamps, write mount metadata). A write-blocker prevents write commands from reaching the source device. In professional settings this is a hardware device; in this lab a Windows registry software method (WriteProtect = 1) was used, and its limitations are documented honestly in the Chain-of-Custody.

5.2 Why Hash Values Matter

The MD5 and SHA-1 hashes of the source drive and the resulting E01 image must be identical. This proves that not a single bit changed during the acquisition. Months later, the same image can be re-hashed; if the values still match those recorded at acquisition, the image is demonstrably untampered. For this NTFS acquisition the values matched exactly: MD5 1127cba3f65d7025b963ab92e942f299; SHA-1 72b809fa1dd637395b1b0807f480cfc11fdbe35e.

5.3 E01 vs RAW

E01 embeds case metadata (examiner, case number, acquisition date) directly inside the image file and supports compression and segmentation. RAW is only the raw bits — any metadata must be maintained in separate files. In court proceedings, E01 is preferred because the metadata is inseparable from the image.

5.4 Chain-of-Custody Discipline

Every undocumented transfer of evidence is a gap that opposing counsel can exploit — arguing that the evidence could have been modified during that period. The Chain-of-Custody document (Appendix A) closes every gap by recording who handled the evidence, when, why, and under what conditions.

5.5 FAT32 → NTFS Pivot

The FAT32 acquisition was forensically valid but unsuitable for $MFT analysis because FAT32 has no MFT. Rather than abandon the MFT exercise, the drive was reformatted to NTFS and re-imaged. Both images are retained as process evidence; the NTFS image is the authoritative artefact for downstream analysis. This pivot was disclosed in the Chain-of-Custody document to preserve integrity.

6. Deliverables Checklist

#	Deliverable	Location / Reference	Status
1	E01 disk image (authoritative, NTFS)	W7_Evidence_USB_NTFS.E01 — E:\Cyberster Internship\Week 7\ForensicLab Images\	Complete
2	Chain-of-Custody document	Appendix A (Chain_of_Custody_W7.docx)	Complete
3	MD5 + SHA-1 hash verification	Figure 3.18 + Section 3.3.1	Complete
4	Alternate Data Streams demonstration	Section 3.1 (Figures 3.1–3.8)	Complete
5	Zone.Identifier legitimate ADS example	Section 3.2 (Figure 3.9)	Complete
6	$MFT extracted and analysed	Sections 3.4–3.5 (Figures 3.22–3.27)	Complete
7	MACB timestamps experiment (5 stages)	Section 3.6 (Figures 3.28–3.32 + table)	Complete
8	File system comparison (NTFS/FAT32/EXT4)	Section 3.7	Complete
9	Autopsy case + ingest configuration	Section 4.1–4.3	Complete
10	Deleted file recovery in Autopsy	Section 4.5 (Figures 4.7–4.9)	Complete
11	Autopsy Timeline analysis	Section 4.6 (Figure 4.10)	Complete
12	Autopsy HTML Forensic Report	Section 4.8 (Figures 4.12–4.13)	Complete
13	Evidence log of significant findings	Section 4.10	Complete

Appendix A — Chain of Custody (Final)

The table below reproduces the final Chain-of-Custody document for this case. The original .docx (Chain_of_Custody_W7.docx) is submitted alongside this report.

Field	Details
Case Number	CYB-W7-001
Evidence Item	USB Flash Drive
Evidence Number	E001
Description	USB Flash Drive — NTFS formatted, EVIDENCE_01 volume label
Manufacturer / Model	Generic Flash Disk USB Device / 0
Date of Acquisition	21 April 2026
Time of Acquisition	03:09:41 (initial FAT32) / 04:38:12 (NTFS re-image)
Examiner Name	Abdul Muqeet Tabraiz
Examiner Organisation	Cyberster Blue Team Internship
Hardware Used	Windows 10 host workstation
Write-Block Method	Software — Windows Registry WriteProtect = 1 (limitation: not hardware blocker)
Image Format	E01 (EnCase format)
Image File Path	E:\Cyberster Internship\Week 7\ForensicLab Images\W7_Evidence_USB_NTFS.E01
MD5 Hash (Source)	1127cba3f65d7025b963ab92e942f299
MD5 Hash (Image)	1127cba3f65d7025b963ab92e942f299
SHA-1 Hash (Source)	72b809fa1dd637395b1b0807f480cfc11fdbe35e
SHA-1 Hash (Image)	72b809fa1dd637395b1b0807f480cfc11fdbe35e
Hash Match Confirmed	YES — both MD5 and SHA-1 hashes match
Storage Location	E:\Cyberster Internship\Week 7\ForensicLab Images
Notes	Drive initially FAT32-imaged, then reformatted to NTFS and re-imaged for MFT analysis. Both images retained. Software write-protection used in lieu of hardware write-blocker — disclosed as limitation.

Appendix B — Figure Index

Figure	Description
2.1	USB drive (CLAY, D:) selected as evidence source.
2.2	Initial USB contents (image.jpg, notes.txt, secret.docx).
2.3	Delete secret.docx to produce recoverable artefact.
2.4	Regedit — StorageDevicePolicies key located.
2.5	WriteProtect DWORD set to 1.
2.6	FTK Imager — Physical Drive selected.
2.7	FTK Imager — source drive dropdown.
2.8	FTK Imager — E01 destination type.
2.9	FTK Imager — Evidence Item Information form.
2.10	FTK Imager — destination folder and filename.
2.11	FTK Imager — Verify/Directory Listing options enabled.
2.12	FTK Imager — acquisition in progress.
2.13	FTK Imager — FAT32 Drive/Image Verify Results (hash match).
2.14	Image output folder — E01 + CSV + log.
2.15	Arsenal Image Mounter — read-only options.
2.16	Arsenal Image Mounter — image mounted.
2.17	Explorer — mounted E01 drive contents.
2.18	Arsenal Image Mounter — Remove action.
2.19	Regedit — WriteProtect reset to 0.
2.20	Initial Chain-of-Custody document.
3.1	Command Prompt — ADS_Test setup.
3.2	Notepad writing hidden ADS content.
3.3	Notepad — visible content only.
3.4	dir — ADS not visible.
3.5	dir /r — ADS revealed.
3.6	Notepad — hidden stream accessed directly.
3.7	streams.exe alongside visible_file.txt.
3.8	streams.exe output — ADS size confirmed.

3.9	Zone.Identifier ADS on downloaded file.
3.10	Format dialog — NTFS full format.
3.11	EVIDENCE_01 root (three folders).
3.12	evidence_test folder contents.
3.13	personal folder (empty after delete).
3.14	work_files folder.
3.15	FTK Imager — NTFS Evidence Information.
3.16	FTK Imager — NTFS destination.
3.17	FTK Imager — NTFS acquisition progress.
3.18	FTK Imager — NTFS Drive/Image Verify Results (primary deliverable).
3.19	FTK Imager — Image Summary (NTFS).
3.20	Arsenal Image Mounter — NTFS image mounted.
3.21	Updated Chain-of-Custody document.
3.22	FTK Imager — $MFT at NTFS root.
3.23	FTK Imager — $MFT export confirmation.
3.24	MFT Explorer — record for image.jpg.
3.25	MFT Explorer — record for notes.txt.
3.26	EVIDENCE_01 drive properties confirmation.
3.27	MFT Explorer — deleted diary.txt record preserved.
3.28	Stage A — creation timestamps.
3.29	Stage B — open-without-change timestamps.
3.30	Stage C — modify timestamps.
3.31	Stage D — copy timestamps.
3.32	Stage E — move timestamps.
4.1	Autopsy — New Case wizard.
4.2	Autopsy — main case view after ingest.
4.3	Autopsy — file tree overview.
4.4	Autopsy — /vol_vol2 contents.
4.5	Autopsy — evidence_test listing + recovered text.
4.6	Autopsy — personal/diary.txt.txt recovered.
4.7	Autopsy — Deleted Files totals.
4.8	Autopsy — Deleted files list.

4.9	Autopsy — Extraction confirmation.
4.10	Autopsy — Timeline view.
4.11	Autopsy — Image Thumbnail view.
4.12	Autopsy — Report Generation Progress.
4.13	Autopsy Forensic Report opened in browser.
4.14	Autopsy — Web Downloads (Zone.Identifier source).
4.15	Autopsy — file properties for recovered report.docx.docx.