



AUBURN

UNIVERSITY

COMP 6350 Project 2 - NTFS

Liam Maher

Lkm0049

10/14/2024

Executive Summary

The objective of this forensic analysis was to investigate an NTFS partition found on the disk image "ShadowLaptop.dd" to recover files related to a digital heist involving a hacker known as "Shadow". The analysis identified and recovered three files, one of which contained an embedded gile using steganography. The recovered files provided valuable evidence related to the hacker's identity, communication, and plans. Below is a summary of key findings:

Question	Answer
Q1) Specify the number and type of partitions on the disk image.	There is 1 NTFS partition on the disk image, named "ShadowLaptop.dd1"
Q2) Specify the number of files, file names, and file size of each file on the partition.	3 Files (1 had an additional embedded file): <ul style="list-style-type: none">• Name: for_ghost• Ext: txt• Size (Real): 151 bytes• Name: passphrase• Ext: txt• Size (Real): 10 bytes• Name: innocent_cat• Ext: jpg• Size (Real): 58493 bytes• Embedded File Name: secret• Embedded File Ext: txt• Embedded File Size: 135 bytes
Q3) Specify the starting and ending byte offset location of each file on the partition.	Innocent_cat.jpg: <ul style="list-style-type: none">• Starting: $133120 * 512 = 68,157,440$• Ending: $133240 * 512 = 68,218,880$ For_ghost.txt: <ul style="list-style-type: none">• Starting: $133240 * 512 = 68,218,880$• Ending: $133248 * 512 = 68,222,976$ Passphrase.txt: <ul style="list-style-type: none">• Starting: $133248 * 512 = 68,222,976$• Ending: $133256 * 512 = 68,227,072$

<p>Q4) Provide a thorough analysis of the recovered files: Determine the contents of these files to understand the objective, the plan, and any other critical information about the hack.</p>	<p>innocent_cat.jpg: This file contained an image of a cat, as well as an embedded file named "secret.txt"</p> <p>secret.txt: This file contained a message to ghost from Shadow1, who's real name is Sean, as well as his address: 7th Brown Ave, Apt #4. It is saying that Ghost and Sean will meet at his place to discuss the plan.</p> <p>for_ghost.txt: This is a message to Ghost from Shadow1 saying that his steganography skills might come in handy and talking about files he left on his laptop. This is referring to the embedded "secret.txt" in innocent_cat.jpg</p> <p>passphrase.txt: This is the passphrase to access the embedded files in innocent_cat.jpg, it is encoded in base64 and decodes to "1234".</p>
--	--

Table of Contents

Executive Summary.....	2
List of Figures	5
List of Tables.....	7
1 Introduction	8
2 Background.....	8
3 Methodology	9
3.1 Partition Identification with “fdisk”	9
3.2 Boot Sector Analysis with “hexdump” and Active Disk Editor	10
3.3 Master File Table (MFT) System Record Analysis	11
3.4 First File MFT Record Analysis	12
3.5 Second File MFT Record Analysis.....	14
3.6 Third File MFT Record Analysis	16
3.7 File Extraction with “dd” command.....	18
4 Results and Discussion.....	21
5 Conclusions and recommendations	23
6 Acknowledgements.....	24
Appendix A: File Recovery and Offset Information	25

List of Figures

Figure 1: Output of fdisk showing the NTFS partition	9
Figure 2: Boot sector analysis with “hexdump”.	10
Figure 3: Boot sector analysis with Active Disk Editor.....	10
Figure 4: Skip command for MFT System Record in Active Disk Editor.	11
Figure 5: MFT System Record Analysis in Active Disk Editor.	11
Figure 6: Skip command for first file MFT record in Active Disk Editor.....	12
Figure 7: First file MFT Analysis with Active Disk Editor.....	13
Figure 8: First file MFT Analysis shows that it is in use.	13
Figure 9: First file MFT analysis shows first and last VCN, allocated vs. real size, and non-resident flags.	13
Figure 10: First file MFT analysis shows cluster count and first cluster of file data.....	14
Figure 11: First file MFT analysis shows file name in hexadecimal notation.	14
Figure 12: Skip command for second file MFT record in Active Disk Editor.	14
Figure 13: Second file MFT Analysis with Active Disk Editor.	15
Figure 14: Second file MFT Analysis shows that it is in use.....	15
Figure 15: Second file MFT analysis shows first and last VCN, allocated vs. real size, and non-resident flags.....	15
Figure 16: Second file MFT analysis shows cluster count and first cluster of file data.	16
Figure 17: Second file MFT analysis shows file name in hexadecimal notation.	16
Figure 18: Skip command for third file MFT record in Active Disk Editor.	16
Figure 19: Third file MFT Analysis with Active Disk Editor.	17
Figure 20: Third file MFT Analysis shows that it is in use	17
Figure 21: Third file MFT analysis shows first and last VCN, allocated vs. real size, and non-resident flags.	17

Figure 22: Third file MFT analysis shows cluster count and first cluster of file data.	18
Figure 23: Third file MFT analysis shows file name in hexadecimal notation.	18
Figure 24: Mapping of NTFS partition.	18
Figure 25: “dd” command to extract innocent_cat.jpg.	19
Figure 26: Recovered contents of innocent_cat.jpg.	19
Figure 27: “dd” command to extract for_ghost.txt.	19
Figure 28: Recovered contents of for_ghost.txt.	20
Figure 29: “dd” command to extract passphrase.txt.	20
Figure 30: Recovered contents of passphrase.txt.	20
Figure 31: Steghide info command using decoded password on innocent_cat.jpg.	20
Figure 32: : Steghide extract command using decoded password on innocent_cat.jpg.	21
Figure 33: Recovered contents of secret.txt.	21

List of Tables

Table 1: Excel spreadsheet containing detailed information regarding the disk image analysis.	25
--	----

1 Introduction

The goal of this project was to successfully analyze a New Technology File System (NTFS) partition on a disk image and retrieve critical files that could provide insights into the operations of a hacker known as "Shadow." The disk image, titled *ShadowLaptop.dd*, was acquired from a laptop belonging to the hacker, and the objective was to recover encoded messages, passphrases, and other relevant information. Through this analysis, key evidence related to the hacker's plans was identified, providing valuable clues for law enforcement.

This report is structured as follows:

- **Background:** Provides context for the investigation and how the laptop image was acquired.
- **Methodology:** Describes the forensic techniques and tools used to analyze the NTFS partition and recover the hidden data.
- **Results and Discussion:** Presents the partition details, byte offsets, and a thorough analysis of the file contents.
- **Conclusions and Recommendations:** Summarizes the findings and suggests next steps for further analysis.
- **Acknowledgements:** Recognizes the resources and guidance that contributed to the project.

2 Background

This project is part of an ongoing investigation into a digital heist orchestrated by a hacker known as "Shadow". Law Enforcement managed to capture Shadow's partner, "Ghost", who provided critical information pertaining to the planning and execution of the heist. During interrogation, it was revealed that Ghost had an old laptop belonging to Shadow.

This laptop was seized by Law Enforcement and a forensic image of it's hard drive, *ShadowLaptop.dd*, was created for analysis. The objective was to recover encoded messages and other relevant information from the disk image, which could provide more evidence related to Shadow's whereabouts or other plans.

The disk image contained a single NTFS partition. This partition was suspected to contain files with hidden data, which had been camouflaged with techniques such as base64 encoding and steganography. To uncover this information, I utilized Active Disk Editor, "fdisk", "dd", "hexdump", and Steghide. Using these tools, I was able to locate the files, determine their byte offsets, and extract hidden messages embedded within the file system.

3 Methodology

This section outlines the forensic steps taken to analyze the *ShadowLaptop.dd* disk image, focusing on partition identification, file system analysis, and file recovery. Each phase of the investigation is accompanied by a figure(s) illustrating the steps taken.

3.1 Partition Identification with "fdisk"

The first step involved identifying the number and type of partitions on the disk image using the "fdisk" command. The output revealed a single NTFS partition.

```
user18@siftworkstation: ~/Documents/Project2
$ fdisk -l ShadowLaptop.dd
Disk ShadowLaptop.dd: 143.05 MiB, 149999616 bytes, 292968 sectors
Units: sectors of 1 * 512 = 512 bytes
Sector size (logical/physical): 512 bytes / 512 bytes
I/O size (minimum/optimal): 512 bytes / 512 bytes
Disklabel type: dos
Disk identifier: 0xf1039d95

Device            Boot Start    End Sectors  Size Id Type
ShadowLaptop.dd1   2048 255999 253952    124M  7 HPFS/NTFS/exFAT
```

Figure 1: Output of fdisk showing the NTFS partition

3.2 Boot Sector Analysis with “hexdump” and Active Disk Editor

To verify the integrity of the NTFS partition, I examined the boot sector using both the “hexdump” command and Active Disk Editor. The hexadecimal data from “hexdump” was cross-referenced with the values obtained through Active Disk Editor to ensure accuracy.

```
user18@siftworkstation: ~/Documents/Project2
$ hexdump -C -s $((2048*512)) -n $((1*512)) ShadowLaptop.dd
00100000 eb 52 90 4e 54 46 53 20 20 20 20 00 02 08 00 00 |.R.NTFS.....|
00100010 00 00 00 00 00 00 f8 00 00 00 00 00 00 00 00 00 |.....|
00100020 00 00 00 00 00 80 00 80 00 ff df 03 00 00 00 00 00 |.....|
00100030 04 00 00 00 00 00 00 00 00 ff 3d 00 00 00 00 00 00 |.....|=.....|
00100040 f6 00 00 00 01 00 00 00 67 66 62 12 19 64 7e 30 |.....gfb..d-0|
00100050 00 00 00 00 0e 1f be 71 7c ac 22 c0 74 0b 56 b4 |.....q|. ".t.V.|
00100060 0e bb 07 00 cd 10 5e eb f0 32 e4 cd 16 cd 19 eb |.....^..2.....|
00100070 fe 54 68 69 73 20 69 73 20 6e 6f 74 20 61 20 62 |.This is not a b|
00100080 6f 6f 74 61 62 6c 65 20 64 69 73 6b 2e 20 50 6c |ootable disk. Pl|
00100090 65 61 73 65 20 69 6e 73 65 72 74 20 61 20 62 6f |ease insert a bo|
001000a0 6f 74 61 62 6c 65 20 66 6c 6f 70 70 79 20 61 6e |otable floppy an|
001000b0 64 0d 0a 70 72 65 73 73 20 61 6e 79 20 6b 65 79 |d..press any key|
001000c0 20 74 6f 20 74 72 79 20 61 67 61 69 6e 20 2e 2e | to try again ..|
001000d0 2e 20 0d 0a 00 00 00 00 00 00 00 00 00 00 00 00 |.....|
001000e0 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 |.....|
*
001001f0 00 00 00 00 00 00 00 00 00 00 00 00 55 aa |.....U.|
00100200
```

Figure 2: Boot sector analysis with “hexdump”.

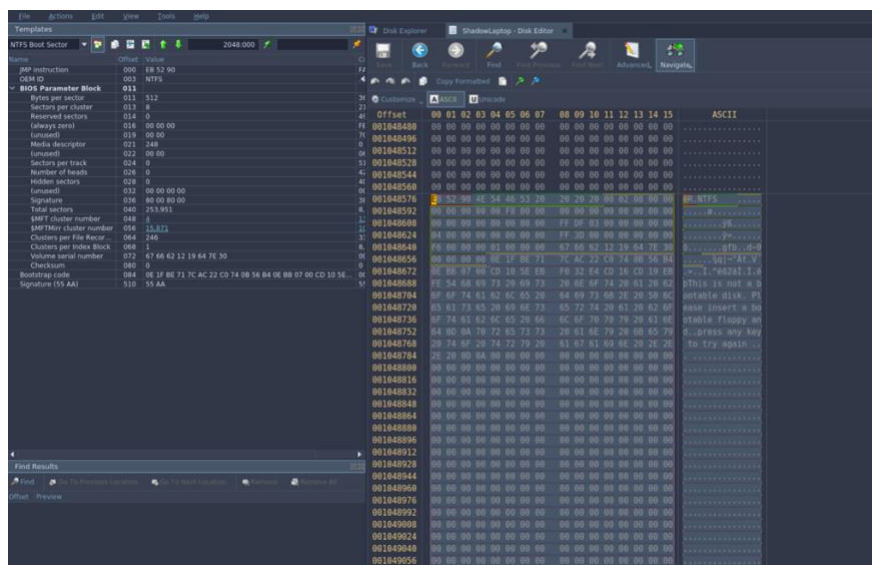


Figure 3: Boot sector analysis with Active Disk Editor.

3.3 Master File Table (MFT) System Record Analysis

After examining the boot sector, I analyzed the Master File Table (MFT) system record, which helps determine the layout of the partition and where the file records are located. Active Disk Editor was used to retrieve and verify the attributes of the MFT system record.

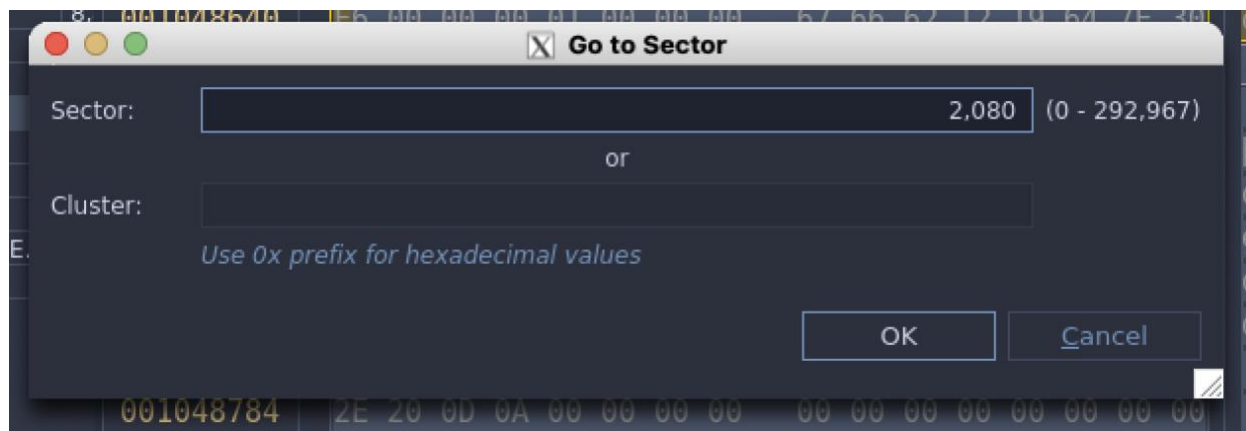


Figure 4: Skip command for MFT System Record in Active Disk Editor.

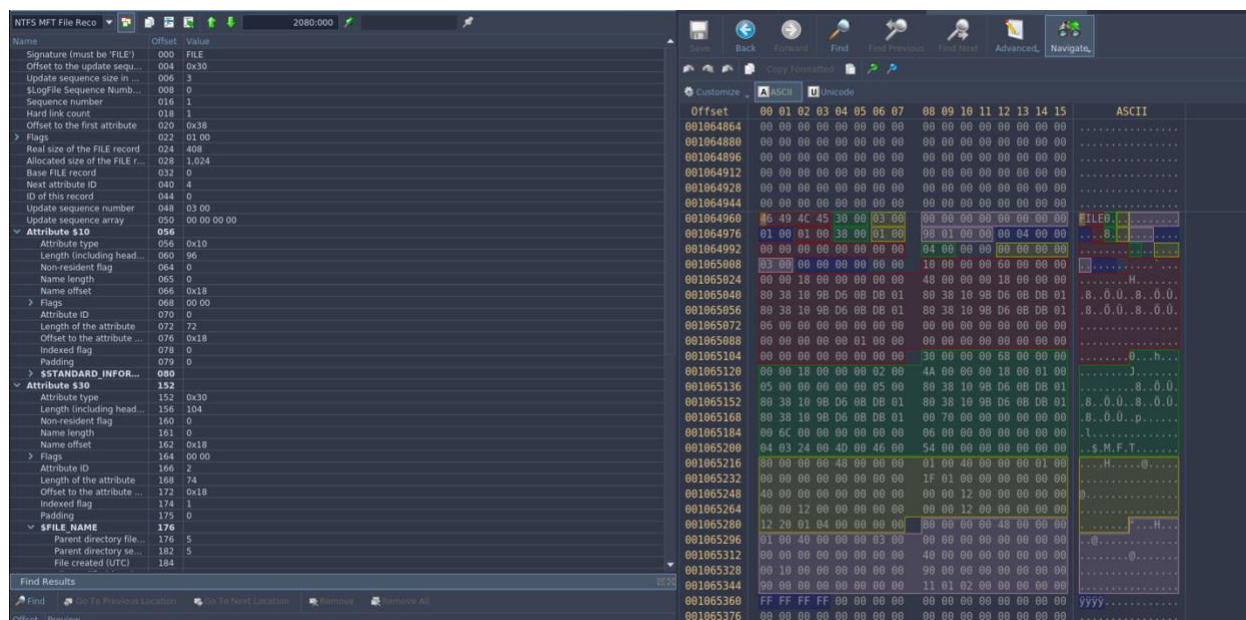


Figure 5: MFT System Record Analysis in Active Disk Editor.

3.4 First File MFT Record Analysis

The first file MFT record was examined to extract metadata such as the file name, file size, and other relevant attributes. To find the first file MFT record, the allocated sectors of the MFT System Records (54 sectors) was added to the start of the MFT System Records (2080), which means the first file MFT Record begins at sector 2134. Each MFT file record occupies 1024 bytes, or 2 sectors. The necessary information was extracted by utilizing Active Disk Editor and applying the MFT File Record template on the first byte of sector 2134. The following figures depict the process of retrieving the necessary information, regarding the first file, from the Active Disk Editor utility.

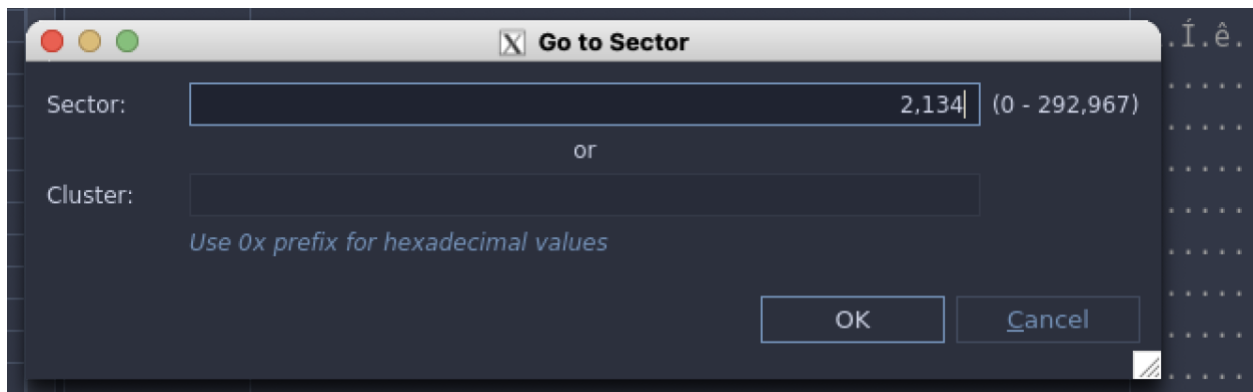


Figure 6: Skip command for first file MFT record in Active Disk Editor.

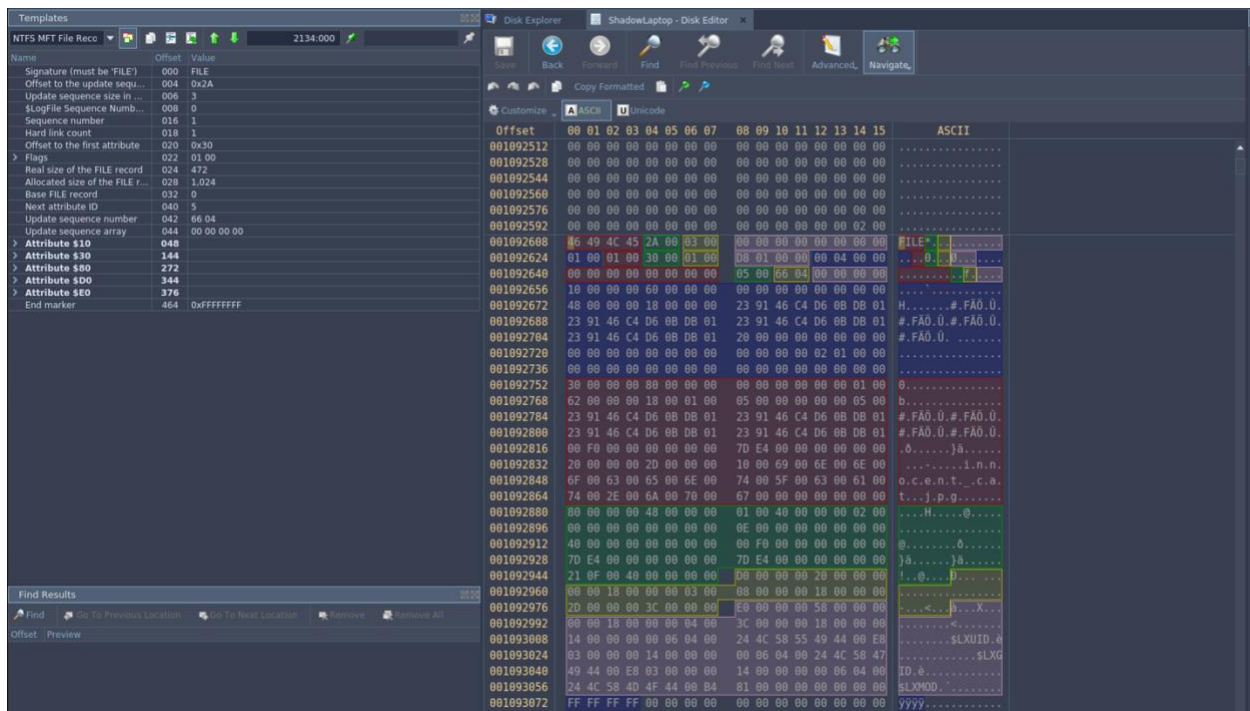


Figure 7: First file MFT Analysis with Active Disk Editor.

Flags	022	01 00
In use	:0	1
Directory	:1	0

Figure 8: First file MFT Analysis shows that it is in use.

Attribute \$80	272	
Attribute type	272	0x80
Length (including head...)	276	72
Non-resident flag	280	1
Name length	281	0
Name offset	282	0x40
Flags	284	00 00
Attribute ID	286	2
First VCN	288	0
Last VCN	296	14
Data runs offset	304	0x40
Compression unit size	306	0
Padding	308	00 00 00 00
Allocated size	312	61,440
Real size	320	58,493
Initialized size	328	58,493
\$DATA	336	
Data run	336	

Figure 9: First file MFT analysis shows first and last VCN, allocated vs. real size, and non-resident flags.

▼ \$DATA	336	
▼ Data run	336	
Size	336	0x21
Cluster count	337	15
First cluster	338	16,384

Figure 10: First file MFT analysis shows cluster count and first cluster of file data.

Length of the attribute	160	98	001092768	62 00 00 00 18 00 01 00	05 00 00 00 00 00 05 00	b.....
Offset to the attribute ...	164	0x18	001092784	23 91 46 C4 D6 0B DB 01	23 91 46 C4 D6 0B DB 01	#.FA0.U.#.FA0.U.
Indexed flag	166	1	001092800	23 91 46 C4 D6 0B DB 01	23 91 46 C4 D6 0B DB 01	#.FA0.U.#.FA0.U.
Padding	167	0	001092816	00 F0 00 00 00 00 00 00	7D E4 00 00 00 00 00 00	.0.....)a.....
▼ \$FILE_NAME	168		001092832	20 00 00 00 2D 00 00 00	10 00 69 00 6E 00 6E 00	...i.n.o.
Parent directory file...	168	5	001092848	6F 00 63 00 65 00 6E 00	74 00 5F 00 63 00 61 00	o.c.e.n.t..c.a.
Parent directory se...	174	5	001092864	74 00 2E 00 6A 00 70 00	67 00 00 00 00 00 00 00	t...p.g.....
File created (UTC)	176		001092880	80 00 00 00 48 00 00 00	01 00 40 00 00 00 02 00	...H....@.....
File modified (UTC)	184		001092896	00 00 00 00 00 00 00 00	0E 00 00 00 00 00 00 00	@.....0.....
Record changed (U...	192		001092912	40 00 00 00 00 00 00 00	00 F0 00 00 00 00 00 00	@.....0.....
Last access time (U...	200		001092928	7D E4 00 00 00 00 00 00	7D E4 00 00 00 00 00 00)a.....)a.....
Allocated size	208	61,440	001092944	21 0F 00 40 00 00 00 00	D0 00 00 00 20 00 00 00	!..0....p.....
Real size	216	58,493	001092960	00 00 18 00 00 00 03 00	08 00 00 00 18 00 00 00<...X...
> File attributes	224	20 00 00 00	001092976	2D 00 00 00 3C 00 00 00	E0 00 00 00 58 00 00 00<...X...
(used by EAs and r...	228	45	001092992	00 00 18 00 00 00 04 00	3C 00 00 00 18 00 00 00<...X...
File name length	232	16	001093008	14 00 00 00 00 06 04 00	24 4C 58 55 49 44 00 E8\$LXUID.è
File name namespace	233	0	001093024	03 00 00 00 14 00 00 00	00 06 04 00 24 4C 58 47\$LXG
File name	234		001093040	49 44 00 E8 03 00 00 00	14 00 00 00 06 04 00	ID.è.....
> Attribute \$B0	272		001093056	24 4C 58 4D 4F 44 00 B4	01 00 00 00 00 00 00	\$LXMOD.....
> Attribute \$D0	344					
> Attribute \$E0	376					
End marker	464	0xFFFFFFFF				

Figure 11: First file MFT analysis shows file name in hexadecimal notation.

3.5 Second File MFT Record Analysis

The second file MFT record was similarly analyzed, confirming details about the second file's location and various information regarding it. The following figures depict the process of retrieving the necessary information, regarding the second file, from the Active Disk Editor utility.

☒ Go to Sector

Sector: (0 - 292,967)

or

Cluster:

Use 0x prefix for hexadecimal values

Figure 12: Skip command for second file MFT record in Active Disk Editor.

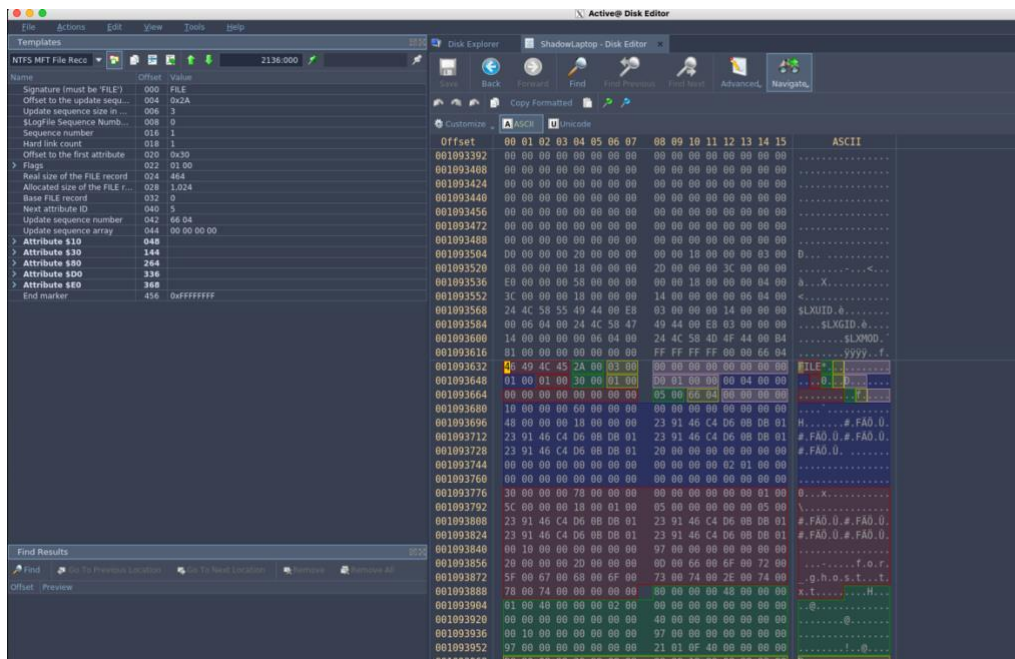


Figure 13: Second file MFT Analysis with Active Disk Editor.

Flags	022	01 00
In use	:0	1
Directory	:1	0

Figure 14: Second file MFT Analysis shows that it is in use.

Attribute \$80	264	
Attribute type	264	0x80
Length (including head...	268	72
Non-resident flag	272	1
Name length	273	0
Name offset	274	0x40
Flags	276	00 00
Attribute ID	278	2
First VCN	280	0
Last VCN	288	0
Data runs offset	296	0x40
Compression unit size	298	0
Padding	300	00 00 00 00
Allocated size	304	4,096
Real size	312	151
Initialized size	320	151
\$DATA	328	

Figure 15: Second file MFT analysis shows first and last VCN, allocated vs. real size, and non-resident flags.

Name	Offset	Value	Save	Back	Forward	Find	Find Previous	Find Next	Advanced	Navigate
Signature (must be 'FILE')	000	FILE								
Offset to the update sequ...	004	0x2A								
Update sequence size in ...	006	3								
\$logfile Sequence Numb...	008	0								
Sequence number	016	1								
Hard link count	018	1								
Offset to the first attribute	020	0x30								
Flags	022	01 00								
Real size of the FILE record	024	464								
Allocated size of the FILE r...	028	1,024								
Base FILE record	032	0								
Next attribute ID	040	5								
Update sequence number	042	66 04								
Update sequence array	044	00 00 00 00								
> Attribute \$10	048									
> Attribute \$30	144									
> Attribute \$80	264									
> Attribute \$D0	336									
> Attribute \$E0	368									
End marker	456	0xFFFFFFFF								

Offset	00	01	02	03	04	05	06	07	08	09	10	11	12	13	14	15	ASCII
001094560	E0	00	00	00	58	00	00	00	00	18	00	00	00	04	00	00	ä...X.....
001094576	3C	00	00	00	18	00	00	00	00	14	00	00	00	06	04	00	<.....
001094592	24	4C	58	55	49	44	00	E8	03	00	00	00	14	00	00	00	\$LXUID.è.....
001094608	00	06	04	00	24	4C	58	47	49	44	00	E8	03	00	00	00	...\$LXGID.è.....
001094624	14	00	00	00	00	06	04	00	24	4C	58	4D	4F	44	00	04\$LXMOD."
001094640	81	00	00	00	00	00	00	00	FF	FF	FF	FF	00	00	66	04yyyy..f.
001094656	06	49	4C	45	2A	00	03	00	00	00	00	00	00	00	00	00	FILE: [.....]
001094672	01	00	01	00	30	00	01	00	00	01	00	00	00	04	00	00	...0.0.....
001094688	00	00	00	00	00	00	00	00	05	00	66	04	00	00	00	00[.....]
001094704	10	00	00	00	60	00	00	00	00	00	00	00	00	00	00	00
001094720	48	00	00	00	18	00	00	00	23	91	46	C4	D6	0B	D8	01	H.....#.FAO.U.
001094736	23	91	46	C4	D6	0B	D8	01	23	91	46	C4	D6	0B	D8	01	#.FAO.U.#.FAO.U.
001094752	23	91	46	C4	D6	0B	D8	01	20	00	00	00	00	00	00	00	#.FAO.U.
001094768	00	00	00	00	00	00	00	00	00	00	00	00	02	01	00	00
001094784	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
001094800	30	00	00	00	78	00	00	00	00	00	00	00	00	00	01	00	B...X.....
001094816	5E	00	00	00	18	00	01	00	05	00	00	00	00	00	05	00	^.....
001094832	23	91	46	C4	D6	0B	D8	01	23	91	46	C4	D6	0B	D8	01	#.FAO.U.#.FAO.U.
001094848	23	91	46	C4	D6	0B	D8	01	23	91	46	C4	D6	0B	D8	01	#.FAO.U.#.FAO.U.
001094864	00	10	00	00	00	00	00	00	0A	00	00	00	00	00	00	00
001094880	20	00	00	00	20	00	00	00	0E	00	70	00	61	00	73	00p.a.s.
001094896	73	00	70	00	60	00	72	00	61	00	73	00	65	00	2E	00	s.p.h.r.a.s.e.
001094912	74	00	78	00	74	00	00	00	80	00	00	00	48	00	00	00	t.x.t.....H.
001094928	01	00	40	00	00	00	02	00	00	00	00	00	00	00	00	00	..@.....
001094944	00	00	00	00	00	00	00	00	40	00	00	00	00	00	00	00@.....
001094960	00	10	00	00	00	00	00	00	0A	00	00	00	00	00	00	00
001094976	0A	00	00	00	00	00	00	00	21	01	10	40	00	00	00	00t.0.....
001094992	00	00	00	00	20	00	00	00	00	00	18	00	00	00	03	00	B.....
001095008	08	00	00	00	18	00	00	00	20	00	00	00	3C	00	00	00<.....
001095024	E0	00	00	00	58	00	00	00	00	00	18	00	00	00	04	00	B...X.....
001095040	3C	00	00	00	18	00	00	00	14	00	00	00	00	06	04	00	<.....
001095056	24	4C	58	55	49	44	00	E8	03	00	00	00	14	00	00	00	\$LXUID.è.....
001095072	00	06	04	00	24	4C	58	47	49	44	00	E8	03	00	00	00	...\$LXGID.è.....
001095088	14	00	00	00	00	06	04	00	24	4C	58	4D	4F	44	00	04\$LXMOD."
001095104	81	00	00	00	00	00	00	00	FF	FF	FF	FF	00	00	00	00yyyy....
001095120	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00

Figure 19: Third file MFT Analysis with Active Disk Editor.

Flags	022	01 00
In use	:0	1
Directory	:1	0

Figure 20: Third file MFT Analysis shows that it is in use

Attribute \$80	264	
Attribute type	264	0x80
Length (including head...	268	72
Non-resident flag	272	1
Name length	273	0
Name offset	274	0x40
Flags	276	00 00
Attribute ID	278	2
First VCN	280	0
Last VCN	288	0
Data runs offset	296	0x40
Compression unit size	298	0
Padding	300	00 00 00 00
Allocated size	304	4,096
Real size	312	10
Initialized size	320	10
\$DATA	328	

Figure 21: Third file MFT analysis shows first and last VCN, allocated vs. real size, and non-resident flags.

▼ \$DATA	328	
▼ Data run	328	
Size	328	0x21
Cluster count	329	1
First cluster	330	16,400

Figure 22: Third file MFT analysis shows cluster count and first cluster of file data.

File name	234	001094720	40 00 00 00 18 00 00 00	23 91 46 C4 D6 08 DB 01	N...U...#..FAO.U..
Attribute type	264	001094736	23 91 46 C4 D6 08 DB 01	23 91 46 C4 D6 08 DB 01	#..FAO.U...#..FAO.U..
Length (including head...)	268	001094752	23 91 46 C4 D6 08 DB 01	20 00 00 00 00 00 00 00	#..FAO.U...#..FAO.U..
Non-resident flag	272	001094768	00 00 00 00 00 00 00 00	00 00 00 00 02 01 00 00
Name length	273	001094784	00 00 00 00 00 00 00 00	00 00 00 00 00 00 00 00
Name offset	274	001094800	30 00 00 00 78 00 00 00	00 00 00 00 00 00 01 00	0...X.....
Flags	276	001094816	5E 00 00 00 18 00 01 00	05 00 00 00 00 00 05 00	^.....
Attribute ID	278	001094832	23 91 46 C4 D6 08 DB 01	23 91 46 C4 D6 08 DB 01	#..FAO.U...#..FAO.U..
First VCN	280	001094848	23 91 46 C4 D6 08 DB 01	23 91 46 C4 D6 08 DB 01	#..FAO.U...#..FAO.U..
Last VCN	288	001094864	00 10 00 00 00 00 00 00	0A 00 00 00 00 00 00 00
Data runs offset	296	001094880	20 00 00 00 2D 00 00 00	0E 00 70 00 61 00 73 00p.a.s.s.
Compression unit size	298	001094896	73 00 78 00 68 00 72 00	61 00 73 00 65 00 2E 00	s.p.h.r.a.s.e.s.
Padding	300	001094912	74 00 78 00 74 00 00 00	80 00 00 00 48 00 00 00	t.x.t.....H...
Allocated size	304	001094928	01 00 40 00 00 00 02 00	00 00 00 00 00 00 00 00	..@.....
Real size	312	001094944	00 00 00 00 00 00 00 00	40 00 00 00 00 00 00 00@.....
Initialized size	320	001094960	00 10 00 00 00 00 00 00	04 00 00 00 00 00 00 00@.....
\$DATA	328				

Figure 23: Third file MFT analysis shows file name in hexadecimal notation.

3.7 File Extraction with “dd” command

With all of the information gathered, we can create a general mapping of the NTFS partition to help extract and find the files.

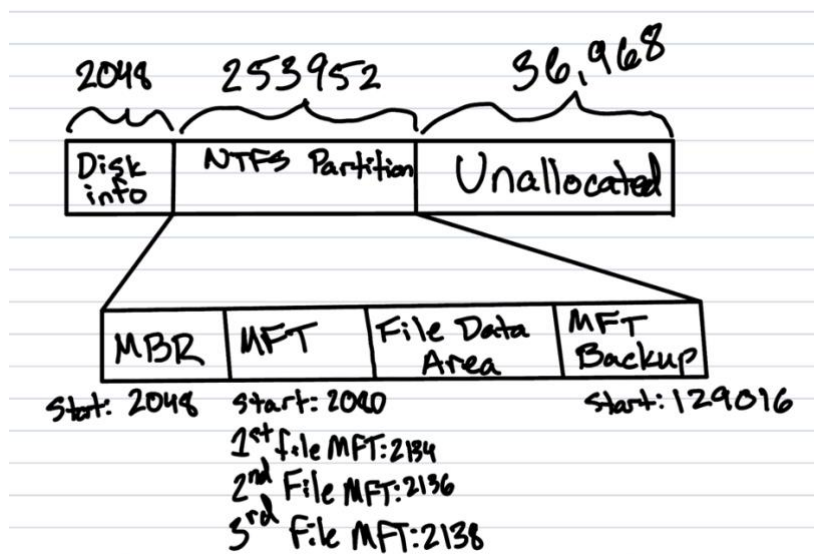


Figure 24: Mapping of NTFS partition.

Using the byte offsets obtained from the MFT record analysis, the dd command was employed to extract files from the disk image. The extracted files were:

- innocent_cat.jpg

```
user18@siftworkstation: ~/Documents/Project2
$ dd if=ShadowLaptop.dd of=innocent_cat.jpg bs=512 skip=133120 count=120
120+0 records in
120+0 records out
61440 bytes (61 kB, 60 KiB) copied, 0.0508547 s, 1.2 MB/s
```

Figure 25: "dd" command to extract innocent_cat.jpg.



Figure 26: Recovered contents of innocent_cat.jpg.

- for_ghost.txt

```
user18@siftworkstation: ~/Documents/Project2
$ dd if=ShadowLaptop.dd of=for_ghost.txt bs=512 skip=133240 count=8
8+0 records in
8+0 records out
4096 bytes (4.1 kB, 4.0 KiB) copied, 0.000341738 s, 12.0 MB/s
```

Figure 27: "dd" command to extract for_ghost.txt.

```

user18@siftworkstation: ~/Documents/Project2
$ cat for_ghost.txt
Ghost,

Your stego skills might come in handy. I left the file on my laptop,
pretty sure you can figure it out. You know where the laptop is.

Shadow1
user18@siftworkstation: ~/Documents/Project2

```

Figure 28: Recovered contents of for_ghost.txt.

- passphrase.txt

```

user18@siftworkstation: ~/Documents/Project2
$ dd if=ShadowLaptop.dd of=passphrase.txt bs=512 skip=133248 count=8
8+0 records in
8+0 records out
4096 bytes (4.1 kB, 4.0 KiB) copied, 0.000387001 s, 10.6 MB/s
user18@siftworkstation: ~/Documents/Project2

```

Figure 29: “dd” command to extract passphrase.txt.

```

user18@siftworkstation: ~/Documents/Project2
$ cat passphrase.txt
MTIzNA==
user18@siftworkstation: ~/Documents/Project2

```

Figure 30: Recovered contents of passphrase.txt.

Upon extraction of the files, further analysis of the “innocent_cat.jpg” file was needed to confirm whether or not any files were embedded in the image using steganography. The passphrase recovered from “passphrase.txt” was encoded in Base64, which decoded to “1234”. This was then used, along with Steghide, to analyze and extract the embedded file, named “secret.txt”.

```

user18@siftworkstation: ~/Documents/Project2
$ steghide info innocent_cat.jpg
"innocent_cat.jpg":
  format: jpeg
  capacity: 3.4 KB
Try to get information about embedded data ? (y/n) y
Enter passphrase:
  embedded file "secret.txt":
    size: 135.0 Byte
    encrypted: rijndael-128, cbc
    compressed: yes

```

Figure 31: Steghide info command using decoded password on innocent_cat.jpg.

```
user18@siftworkstation: ~/Documents/Project2
$ steghide extract -sf innocent_cat.jpg -xf secret.txt
Enter passphrase:
wrote extracted data to "secret.txt".
```

Figure 32: : Steghide extract command using decoded password on innocent_cat.jpg.

```
$ cat secret.txt
We will meet at my place to discuss the plan, here's my address: 7th Brown Ave, Apt #4.
BTW, my real name is Sean.

Shadow1
2024-08-16
```

Figure 33: Recovered contents of secret.txt.

4 Results and Discussion

As seen from the steps taken above, there were 3 files extracted from the disk image, one of which had an additional embedded file within it. When analyzing the NTFS Boot Sector with the Active Disk Editor in **Figure 3**, the following information about the partition was determined:

- Bytes/Sector = 512
- Sectors/Cluster = 8
- Sectors Before Partition = 2048
- MFT Cluster Start = 4
- MFT Record Size = 1024

The extracted files, as well as their attributes and offsets, are listed below:

- innocent_cat.jpg

- File Name: innocent_cat
- File Extension: jpg
- Attributes: \$10, \$30, \$80, \$D0, \$E0
- In Use (yes/no): Yes (1 – Active)
- Non-Resident (yes/no): Yes
- Allocated Size: 61440 bytes
- Real Size: 58493 bytes
- Starting Cluster: 16384
- Starting Byte Offset: 133120 (starting sector offset) * 512 = 68,157,440
- Ending Byte Offset: (133120 + 120 (allocated file size in sectors)) * 512 = 68,218,880
- Embedded File Name: secret
- Embedded File Extension: txt
- Embedded File Size: 135 bytes
- for_ghost.txt
 - File Name: for_ghost
 - File Extension: txt
 - Attributes: \$10, \$30, \$80, \$D0, \$E0
 - In Use (yes/no): Yes (1 – Active)
 - Non-Resident (yes/no): Yes
 - Allocated Size: 4096 bytes
 - Real Size: 151 bytes
 - Starting Cluster: 16399
 - Starting Byte Offset: 133240 (starting sector offset) * 512 = 68,218,880
 - Ending Byte Offset: (133240 + 8 (allocated file size in sectors)) * 512 = 68,222,976

- passphrase.txt
 - File Name: passphrase
 - File Extension: txt
 - Attributes: \$10, \$30, \$80, \$D0, \$E0
 - In Use (yes/no): Yes (1 – Active)
 - Non-Resident (yes/no): Yes
 - Allocated Size: 4096 bytes
 - Real Size: 10 bytes
 - Starting Cluster: 16400
 - Starting Byte Offset: 133248 (starting sector offset) * 512 = 68,222,976
 - Ending Byte Offset: (133248 + 8 (allocated file size in sectors)) * 512 = 68,227,072

The information found above was all extracted or derived from the Active Disk Editor and Steghide analysis for each file record in **Section 3**. In **Figure 25**, we see that the contents of “innocent_cat.jpg” is simply an image of a cat. In **Figure 27**, the contents of “for_ghost.txt” we see that Shadow1 left a message for Ghost alluding to the use of steganography in the “innocent_cat.jpg file”. **Figure 28**, the contents of “passphrase.txt”, includes a passphrase “MTIzNA==” which was found to be encoded in base64, and decodes to “1234”. This passphrase was then used to extract the embedded file, “secret.txt”, from “innocent_cat.jpg”. Within “secret.txt”, we find critical information regarding the hacking plot, where Shadow1 reveals his real name to be Sean, and his home address, “7th Brown Ave, Apt #4”.

5 Conclusions and recommendations

The forensic analysis of the NTFS partition on the “ShadowLaptop.dd” disk image successfully recovered critical files that provided insights into the communications and plans of a hacker known as “Shadow”. The key findings revealed hidden messages embedded in an

image file, as well as a passphrase that unlocked these hidden contents. Notably, the investigation identified shadow's real name (Sean) and address, which could be pivotal in locating and apprehending the hacker. It is recommended that law enforcement continues to investigate Shadow's associates and potential safe houses, particularly focusing on the address found in "secret.txt". The recovered files should be securely stored and preserved for potential legal proceedings, ensuring that the chain of custody is maintained.

6 Acknowledgements

I would like to express my sincere gratitude to Dr. Farah Kandah, whose insightful lectures and guidance were instrumental in helping me navigate the process of file recovery and digital forensics.

Appendix A: File Recovery and Offset Information

The table below summarizes the information gathered during the forensic analysis of the NTFS partition, including file names, extensions, real and allocated sizes, starting and ending sectors, as well as the offsets for each file. This data was crucial in the recovery of files from the disk image.

Table 1: Excel spreadsheet containing detailed information regarding the disk image analysis.

General NTFS Values													
Description	Value	Structure	Start Location	Size									
Bytes/Sec	512	MBR	0x0	2									
Sec/Cluster	8	MBR	0x0	1									
Reserved Sectors	0	MBR	0x0	2									
Sectors Before Partition	2048	MBR	0x1C	4									
SMFT Cluster Start	4	MBR	0x30	8									
SMFT Mirr Cluster Start	16384	MBR	0x38	8									
# System SMFT Records	22	MFT											
SMFT Record Size	1024	MFT											
NTFS Data Structure Locations													
Allocated (Sectors)	Start												
Sectors to Partition	2048												
SMFT Mirr Start	16384												
SMFT Cluster Start	4												
SMFT System Records	54												
File #1 SMFT Record	2												
File #2 SMFT Record	2												
File #3 SMFT Record	2												
NTFS SMFT Record Information													
Filename	Ext	Attributes	In Use (Header)	Non-Resident (0x80)	Allocated Size (x80)	Real Size (x80)	1st Cluster (x80 - 2)	1st Sector	1st Sector + Disk Offset	# Clusters (x80)	# Sectors	First VCN (x80)	Last VCN (x80)
innocent_cat	.jpg	\$STANDARD_INFORMATION (x10) \$FILENAME (x30) \$DATA (x80) \$DO, \$EO	Yes (1 - Active)	Yes	61440	58483	16384	131072	133120	15	120	0	14
for_ghost	.txt	\$STANDARD_INFORMATION (x10) \$FILENAME (x30) \$DATA (x80) \$DO, \$EO	Yes (1 - Active)	Yes	4096	151	16399	131192	133240	1	8	0	0
passphrase	.txt	\$STANDARD_INFORMATION (x10) \$FILENAME (x30) \$DATA (x80) \$DO, \$EO	Yes (1 - Active)	Yes	4096	10	16400	131200	133248	1	8	0	0
Confirmation Command													
hexdump ShadowLaptop.dd -s \$((133120/512)) -n \$((120/512))													
hexdump ShadowLaptop.dd -s \$((133240/512)) -n \$((8/512))													
hexdump ShadowLaptop.dd -s \$((133248/512)) -n \$((8/512))													
Recovery Command													
dd if=ShadowLaptop.dd of=innocent_cat.jpg bs=512 skip=133120 count=120													
dd if=ShadowLaptop.dd of=for_ghost.txt bs=512 skip=133240 count=8													
dd if=ShadowLaptop.dd of=passphrase.txt bs=512 skip=133248 count=8													