Forensic Analysis of Persistent Data Storages Analyzing NTFS Formatted Drives

H. G. OZGUR[1] and S. SAHIN[1]

[1]Izmir Institute of Technology, Izmir/Turkey, huseyinozgur@iyte.edu.tr
[1]Izmir Institute of Technology, Izmir/Turkey, serapsahin@iyte.edu.tr

***Abstract* - Computational forensic is a far-reaching field for criminal and civil laws in this age. Aim of this project is to develop an educational tool to analyze persistent storage devices to find possible evidences. Evidences are found as metadata information on these drives. These metadata information are collected in a comma separated value-CSV format document. This CSV file can be imported to a database and can be easily analyzed. By the classical forensic investigator tools mostly can analyze only one media at a time, but by this way more than one digital evidence can be comparatively analyzed. Also, deleted files may hold evidences, so recovering of deleted files is an additional topic for this study. The structured codes and related documents have shared on github as an open project; to give a chance to extension of this study by new projects and we hope that the product of this study can be useful tool for computational forensic courses.**

***Keywords* - Computational forensic, Hard-drive Analysis, Metadata, Recovering Files Deleted**

I. INTRODUCTION

The digital age defines a new virtual world which includes many type of activities, operations and relations among many type of individuals. The individuals quite likely use digital devices for some reason. These devices can be selected as target or source for the criminal activities and they must be defined and verified by digital evidences according to laws and regulations. Digital forensic is a very specific area which focuses on required specific analyzing tool and their usage principles and methodologies to discover evidences. The digital evidence or electronic evidence; according to Eochan [6] can be any information stored or transmitted in digital form which can be used as a proof material to a court case. The very important point is that the authenticity of the digital evidence must be protected and should be provable.

If an investigator wants to find evidences which are in digital form, she starts with analyzing of hard-drives, because digital evidences are most commonly held in static storage units. Analyzing of static storage units such as hard drives, memory sticks etc. is basis of digital forensic. In this project, a tool is developed to analyze the persistent data storages, and first focus is to find all patterns which may constitute evidence for a case. Even evidences are deleted, or corrupted, or modified, they usually leave a trace on these units and its file system structured records. Due that reason the second focus of this project is to recover of deleted files in storage unit.

Analyzing hard-disks to search evidences is required to have technical and conceptual understanding. The layers of a storage device from application to its hardware must be known. A hard-drive abstraction involves four parts which are physical layer, partitions on device, used file system, and its semantic layer. Physical layer reflects device itself and it is split into one or more partition. Each partition needs a bridge (abstraction layers to have a connection) between files on system hardware and user's application. File system provides this bridge. Semantic layer has different design structures according to different file systems.

The following Figure 1 shows overall steps of finding evidences on hard-drives:

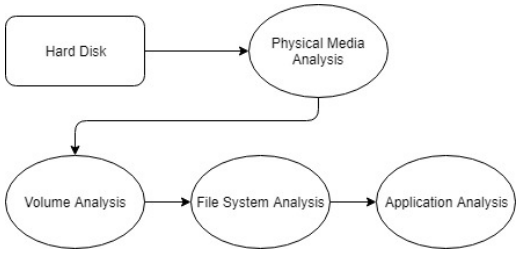

Figure 1 Steps of finding evidence on hard-drives [1]

- In the physical media analysis, the locations of the disk partitions are determined, and file formats of the partitions are detected.
- Each file system has its own structure. For NTFS formatted drives, beginning of the master file table (MFT) records is in the volume analysis part, and some information about the volume are obtained, then sector by sector analysis is started.
- In the file system analysis part, metadata in MFT record are extracted. All records of MFT as metadata are written on comma separated value - CSV document. In this project, also deleted files are recovered. Then, each file is analyzed in analysis phase.

The second chapter of this article includes the background information about the physical media analysis methodology. Third chapter explains the volume analysis of NTFS structure. Fourth chapter presents the developed solution tool for file system analysis over previous steps. At last the conclusion chapter explains the success results of this study and future works to enlarge the efficiency domain of this tool.

II. PHYSICAL MEDIA ANALYSIS

At the beginning of the hard-drive analysis, master boot record (MBR) is analyzed. MBR is located always in the first sector of booting device. Hard-drive partition information starts at the offset 0x1BE in MBR table and its size is 16 bytes. There are four partition table entries in MBR. Last two bytes of MBR sector are 0x55 and 0xAA.

Structure of the partition table entry explained in [2], [3], here:

* Important bytes of MBR for forensic analysis

- Byte 1*: 0x80 → active(bootable), 0x00 → inactive (not bootable)
- Byte 2, 3, 4: Location of the beginning of the partition
- Byte 5*: Type of the partition (e.g.0x07 is NTFS)
- Byte 6, 7, 8: Location of the end of the partition
- Bytes 9, 10, 11, 12*: Logical block addressing of start sector in little endian format, and it is measured in blocks
- Bytes 13, 14, 15, 16: Number of sectors in the partition

III. VOLUME ANALYSIS

A. *NTFS Structure*

When a hard-drive is formatted as NTFS, the space which involves the first 16 sectors is known as partition boot sector ($Boot) which includes boot metadata file. It includes bootstrap code at the first 15 sectors, and then continues with initial program loader (IPL) of the boot sector. Also, it contains a copy of boot sector at the last sector to make system more reliable.

The Master File Table (MFT) holds records about each file and directory on the NTFS formatted hard drive. There are 2 mirrors of MFT, if one of them is damaged, then the other one is used to recover. The location of Master File Table in a NTFS volume is not in the predefined clusters. In the boot sector, there are two fields which are BIOS parameter block - BPB and Extended BPB [7]. These fields describe the location information of MFT. In this way, if there is a problem occurred in MFT's own sector, operation can be transferred to the mirror of MFT.

NTFS Boot Sector	Master File Table	File System Data	Master File Table Copy

Figure 3 Structure of NTFS drives

The above Figure 3 indicates the structure of NTFS formatted drive. And following table 1 shows the layout of MBR. Boot sector begins with jump instruction which is EB 52 90. Then it is followed by Original Equipment Manufacturer - OEM ID which is used to identify the name and version number of the operating system that formatted the volume.

Table 1 Structure of Master Boot Record (MBR) [4]

Offset	**Length of Field**	**Meaning**
0x00	3 Bytes	Jump Instruction
0x03	8 Bytes	OEM ID
0x0B	25 Bytes	BPB
0x24	48 Bytes	Extended BPB
0x54	426 Bytes	Bootstrap Code
0x1FE	2 Bytes	End of the Sector Marker

IV. FILE SYSTEM ANALYSIS

A. *NTFS Master File Table Analysis*

MFT table stores a record for each file or folder which is on an NTFS volume [8]. Also, MFT stores records about itself. Each record is 1KB.

These records consist of a header and a few attributes as you can see in Figure 5.

MFT Entry Header	Attribute	Attribute	Attribute	Unused space

Figure 5 Structure of an MFT record

The following table 2 represents the layout of an entry of MFT. Some important metadata given in this table are:

- Magic number indicates the beginning of the MFT record. Its value is always "46 49 4C 46" which is represented in hexadecimal form.
- LSN changes when the record is modified.
- Sequence number indicates that how many times this MFT record has been reused.
- Hard link count indicates the number of entries that is referencing this record.

Table 2 Layout of an MFT record

Offset	**Size**	**Description**
0x00 – 0x03	4	Magic Number: 'FILE'
0x04 – 0x05	2	Offset to the update sequence
0x06 – 0x07	2	Number of entries in fixup array
0x08 – 0x0f	8	$LogFile Sequence Number(LSN)
0x10 – 0x11	2	Sequence Number
0x12 – 0x13	2	Hard Link Count
0x14 – 0x15	2	Offset to first attribute
0x16 – 0x17	2	Flags: 0x00: Unallocated space, free to be over written, 0x01: record in use, 0x02: directory
0x18 – 0x1b	4	Used size of MFT entry
0x1c – 0x1f	4	Allocated size of MFT entry
0x20 – 0x27	8	File reference to the base FILE record
0x28 – 0x29	2	Next attribute ID
0x30 – 0x1000		Attributes and fixup value

1) *MFT Attributes*

Table 3 shows possible attributes which can be included in an MFT record. Each attribute has different structure and consist of different metadata about the file.

Table 3 MFT record attributes

Type	OS	Name
0x10		$STANDARD_INFORMATION
0x20		$ATTRIBUTE_LIST
0x30		$FILE_NAME
0x40	NT	$VOLUME_VERSION
0x40	2K	$OBJECT_ID
0x50		$SECURITY_DESCRIPTOR
0x60		$VOLUME_NAME
0x70		$VOLUME_INFORMATION
0x80		$DATA
0x90		$INDEX_ROOT
0xA0		$INDEX_ALLOCATION
0xB0		$BITMAP
0xC0	NT	$SYMBOLIC_LINK
0xC0	2K	$REPARSE_POINT
0xD0		$EA_INFORMATION
0xE0		$EA
0xF0	NT	$PROPERTY_SET
0x100	2K	$LOGGED_UTILITY_STREAM

For instance, the structure of $STANDARD_INFORMATION attribute is presented in table 4:

Table 4 The structure of $STANDARD_INFORMATION attribute

Offset	Size	Description
~	~	Standard Attribute Header
0x00	8	C Time – File Creation
0x08	8	A Time – File Altered
0x10	8	M Time – MFT Changed
0x18	8	R Time – File Read
0x20	4	**DOS File Permissions**
0x24	4	Maximum Number of Versions
0x28	4	Version Number
0x2C	4	Class ID

All the MFT attribute structures can be found in [9].

The other important point is about the size of the file content which is addressed by MFT record. The files are either resident or nonresident according to its size. When a file content can fit within the same cluster of MFT file record, it is called resident attribute. Otherwise, it is called non-resident attribute and it can spread on different clusters.

For example, $DATA attribute contains the either location of the content of file or content of the file itself. If it is non-resident, then content of the file is stored in different location in the disk. There are different addressing methods for fragmented, un-fragmented, and compressed files. To obtain file content, data-run is used to address file content. The reader can reach more detailed information on data-run from [5].

Almost all attributes start with Standard Attribute Header. This header includes important information about attributes. You can see the layout of the header in the table 5 given below:

Table 5 Standard Attribute Header

Offset	Size	Description
0x00	4	Attribute Type
0x04	4	Length (including header)
0x08	1	Non-resident flag
0x09	1	Name length
0x0A	2	Offset to the Name
0x0C	2	Flags
0x0E	2	Attribute ID
0x10	4	Length of the Attribute
0x14	2	Offset to the Attribute
0x16	1	Indexed flag
0x17	1	Padding
0x018	2N	The Attribute's Name
2N+0x18	L	The Attribute

Some important metadata which part of the header are:

- Attribute Type: For example, if its value is 0x10, then it refers to $Standard_Information attribute.
- Length: It gives us total length of attribute.
- Non-resident flag: If its value is 0x00, then given attribute is resident. If it is 0x01, then the attribute is non-resident. MFT record size is fit for resident attributes. If size is not enough, then the attribute is located somewhere else in the volume.
- Offset to the Attribute: This indicates the location of the attribute.

2) Forensics Tool Developed in Scope of the Project

Forensic tool development for persistent data storages starts with reading the first sector of the physical media. In that step, program checks whether MBR (Master Boot Record) is included or not. If MBR is included, then program looks for NTFS formatted partition information in MBR, in that way, location of NTFS boot sector is found, and analysis continues in that location. If MBR is not included, then program checks whether the first sector is NTFS boot sector or not. If it is NTFS boot sector, analysis continues in the first sector.

NTFS boot sector includes some metadata about the volume. These metadata are extracted to a CSV file. Also, NTFS boot sector is used to locate beginning address of the MFT (Master File Table). Program continues its execution in at that location.

Most of the work is done in MFT records. Each MFT record is analyzed to obtain metadata of each file or folder which is stored in the storage. Some Metadata of each record are extracted to a CSV file. Each line of the CSV file includes metadata of a folder or a file. One of the metadata indicates whether the file is deleted or not. If the file deleted, and there is not an overwrite on its location, then program tries to recover the deleted file. But the recovery process is implemented for the file contents which have been stored in continuous clusters.

3) Similar Products and Novelty of the Project

There are several similar products. But most of them are not open source. To make an expressive comparison, some open source products has been selected.

This project can be compared with "Analyze MFT[1]" solution. The software projects can be compared according to the principles of software engineering such as documentation, learnability, changeability, interoperability etc. Table 7 shows the differences when these two projects compare with these principles.

Table 7 Differences of Analyze MFT, and the Project

	Comparison of Analyze MFT and Our Project
Documentation	Detailed documents are provided by our project but Analyze MFT has not been well documented. It just includes how a user can implement it on her system.
Learnability	Our project comes with both detailed documentation, and well commented codes. Learning it is easy for both programmers, and Investigator.
Changeability	Any programmer can change our code after reading its documents and reading its code easily. Making any change on Analyze MFT is hard, because learning it takes time.
Evolvability	Evolvability takes time for both applications. Both analyze NTFS formatted drives. Expanding their scope requires analyzing structure of different file formats. Pros of our program that it provides sector reader class, and csv documentation class. These two classes can be used to analyze any other file formats. Also, after reading documentation, it gives an idea about the processes to analyze a disk.
Interoperability	Our application comes with some features which are sector reading, recovering deleted files, csv documenting. These features can be used separately inside any other application, and the other applications can use our program to get metadata of MFT records. Analyze MFT can be used CSV documentation, anomaly detection etc. inside an application.

Table 7 has been created according to our usage experiences of Analyze MFT.

The Analyze MFT which has been developed very similar in the scope of our project. But, our application also provides recovering of deleted files. Codes of MFT project has not been commented well, so it is hard-to understand the code. But, almost each line of our code includes an explanation part. The Analyze MFT has not enough documentations and our project is published with its documents as well.

There are also some other applications like Autopsy, but they have been developed as open product for end users, and they include different functionalities. For example, Autopsy can be used for hash filtering, web artifacts extraction etc. It is provided with usage training, and commercial support for forensics analysts. Our project can be used by developers and forensics end users, but main purpose is that users can easily understand the processes of analyzing a digital source. After understanding, anyone can develop more comprehensive forensic analysis environment with our project.

4) Improvable Aspects and Future Work

Improvable aspects of the project:

- Recovering fragmented files
- Different kind of MFT attributes can be analyzed in detail, for example creation times or read times of files can be exported and relation between them can be observed
- CSV documentation content can be extended
- An GUI can be designed
- An option to recover deleted data files can be added
- An option to view desired sectors as hexadecimal can be added

Future work:

- Different file systems like FAT, ext2, ext3, ext4 etc. can be analyzed
- Non-persistent storages can be analyzed, different requirements and procedures are required to collect evidence from a volatile memory

V. CONCLUSION

Scope of the analyzing persistent data storages is a beginning objective on NTFS file systems, but it could involve a wide range of file formats. Each file format has its own structure. Finding documentation about any kind of file format can be hard. Most of the documentations include only limited part of the structure of the desired file format.

Values of the project:

- Analyzing storages to find evidences is a sensitive process. First, the evidence properties of the collected data should be provable. Second, the secrecy or privacy issue of analyzed data should be guaranteed. Data can be belonging to government or persons. Due this reason, forensic analysis job is mostly done in national boundaries by governmental agencies. This project can be starting point to develop national forensic analysis tool.
- The current forensics tools are imported, and this causes high costs to buy product and its training for users. Developing national product cuts off these costs.

[1] https://pypi.python.org/pypi/analyzeMFT/2.0.19

- As a nation we would like to have more forensic professionals. Some of the universities and computer engineering departments include related courses. This national tool can be used training purposes in lectures and used as support laboratory material.

- This open tool can be enhanced by anyone to develop different kind of forensic facilities.

The most valuable point in this project is to provide environment for investigators to create associations inside different kind of collected data. Because the developed program gives a CSV file as an output. This CSV file can be imported to a relational database, and investigators can create necessary queries to see relations among different metadata records on single or many of different storage units.

All in all, forensic science is the topic which certainly must be worked on it. The necessary lectures are given by different educational organizations. The purpose of this project is to create a lecture material and as a future objective to start an open project to develop a national free of charge forensic storage analyzing tool.

VI. REFERENCES

[1] File System Forensic Analysis - By Brian Carrier

[2] http://www.bydavy.com/2012/01/lets-decrypt-a-master-boot-record/

[3] https://www.youtube.com/watch?v=BG1gQ4Ta79M

[4] http://www.cse.scu.edu/~tschwarz/coen252_07Fall/Lectures/NTFS.html

[5] http://homepage.cs.uri.edu/~thenry/csc487/video/66_NTFS_Data_Runs.pdf

[6] Casey, Eoghan (2004). Digital Evidence and Computer Crime, Second Edition. Elsevier. ISBN0-12-163104-4

[7] Hardware White Paper, FAT32 File System Specification, General Overwiev of On-Disk Format, Version 1.03, Dec. 6, 2000, Microsoft Corporation, https://staff.washington.edu/dittrich/misc/fatgen103.pdf

[8] The NTFS File System, https://technet.microsoft.com/en-us/library/cc976808.aspx

[9] Richard Russon Yuval Fledel, NTFS Documentation, http://inform.pucp.edu.pe/~inf232/Ntfs/ntfsdoc.pdf