International Journal of Science and Research (IJSR)
ISSN (Online): 2319-7064
Index Copernicus Value (2013): 6.14 | Impact Factor (2015): 6.391

Recovering Deleted Files from NTFS

Rincy Roy Oommen[1], Princy Sugathan[2]

[1, 2]Cochin University of Science and Technology, College of Engineering Kallooppara, Kerala, India

Abstract: ***Recovering lost and deleted information is one of the main part in Digital Forensics. Data recovery is a process which finds and recovers data, in which there may be some risks happens, for no all situations can be defined or arranged previously. Data recovery also retrieves lost, deleted, unusable or inaccessible data that lost for various reasons. In computer forensics, the main source of evidence is the data which is stored in the file. The file system is used to manage all files present on the disk. A suspect can remove evidence by deleting evidence containing files. So, it is important for forensic investigator to get back the deleted evidences. This paper described the structure of the NTFS file system and proposed a method to recover deleted files from the system by analysing the MFT entry and also detects the directory from which the file was deleted.***

Keywords: Forensics, Data Recovery, File System, NTFS, MFT Entry

1. Introduction

A file system is used as the methods and data structures that an operating system uses to keep track of files on a disk or partition. The Windows NT file system (NTFS) provides a better combination of performance, reliability, and compatibility not found in the FAT file system. The NTFS file system includes some security features required for the high-end personal computers and file servers in a corporate environment.

Recently, cyber crimes have been increasing drastically. After conducting a crime, the suspect tries to delete the evidence containing files present in the computer. So data recovery needs more attention in digital forensics. Data recovery is a process of finding and recovering lost, deleted, inaccessible or unusable data lost for various reasons. Data recovery can be classified into two categories: software data recovery and hardware data recovery. Software data loss tends to be the deletion of files and the overwriting of data. Hardware data loss is occurred due to physical problems such as a hard drive fault or losing a thumb drive. If the data on the disk is overwritten, then the old data cannot be recovered. There are two file recovery methods are used for files that have not been overwritten, which includes:

- Recovering files through analysis of the information about files and folders.
- Recovering files using search for known file types.

In NTFS, data recovery of deleted files such as non-resident files, compressed files and encrypted files are much difficult. Also detecting the directory from which the deleted files originated adds to its complexity. This paper proposes to develop a tool that incorporates different file recovery processes as a single framework and detects the directory of the deleted files by analysing MFT entries of NTFS.

1.1 NTFS Concepts

The NTFS file system which supports data access control and ownership privileges for the integrity of critical data .NTFS is the only file system that allows the users to assign permissions to individual files. The NTFS file system has a simple and a very powerful design. Basically, everything on the NTFS volume is a file and everything in a file is designed as an attribute, from the data attribute to the file name attribute through the security attribute. The following figure illustrates the NTFS volume layout when formatting has finished.

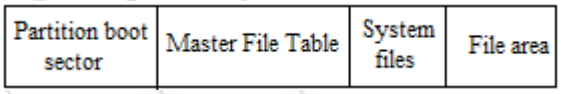

Figure 1: Formatted NTFS Volume

1.2 NTFS File System Structure

NTFS file system creates a File Record for each user data file. All these File Records are stored in a special table named as Master File Table (MFT). The size of MFT entry is 1024 bytes. The Master File Table location is available from partition boot sector (512 bytes), which is the first sector of the formatted partition of NTFS. Partition boot sector is different than disk boot sector which is first sector of entire disk that contains information to find start address of partition. The offset 48-55 (8 bytes) in the partition boot sector represents the location of the starting cluster address of the MFT. To get the physical address of NTFS partition relative to the start of the partition by multiplying it with bytes per sector (located at offset 11-12) and sectors per cluster (located at offset 13).

MFT Start Address = Start Cluster of MFT * Bytes per Sector * Sectors per cluster.

1.3 NTFS Master File Table (MFT)

The heart of NTFS is MFT. NTFS reserves the first 16 records of the table for storing special information. If the first MFT record is damaged, NTFS reads the second one to find the MFT mirror file, whose first record is indistinguishable to the first record of the MFT. In the boot sector the locations of the data segments for both the MFT and its mirror files are recorded . The figure provides a simplified illustration of the MFT structure:

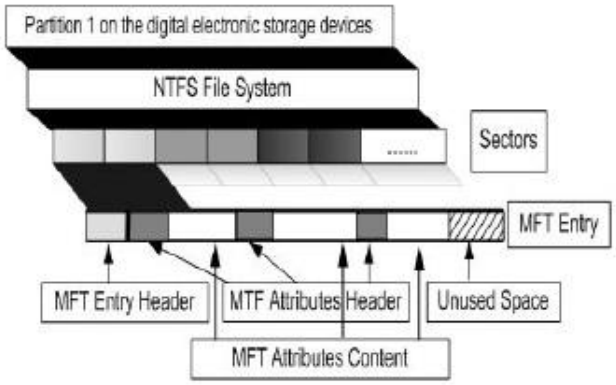

Figure 2 : MFT Layout Structure

The attributes of a file are written to the certain amount of the allocated space provided by the MFT for each file record. Small files and directories (typically 512 bytes or smaller) can resides entirely in the master file table record. Large directories are organized into B-trees, having records with pointers to external clusters involves directory entries that could not be fit into the MFT structure. Directory records are housed within the master file table just like file records which holds index information.

1.4 MFT Entry Attributes And Headers

The file system NTFS views each file (or folder) as a set of file attributes. File attributes provides some information such as file's name, its security information, and even its data and these information are stored in MFT entry as series of attributes. Each attribute has its own internal structure and specific purposes. Each attribute is identified by using an attribute code or an attribute name. Every file does not contain each of the attribute listed in the table. Different attributes are added in File Record depending on the file type. NTFS creates metadata files to manage the entire file system which contains the administrative data. To differentiate NTFS metadata files begin with ”$” sign.

The header structure of MFT helps in analyzing the MFT ntry and its attributes. In the MFT Header structure, Offset to first attribute is specified which is relative to start of MFT Entry and also used to start parsing attributes which are present in MFT Entry. Flags are used to pinpoint whether the entry is for file or folder and also helps to identify whether the file/folder is deleted or not. Each Attribute present in MFT Entry also has a well defined header associated with it, which holds important details such as name, type, Id, length and some other information. For large files, it only holds the location information. If the data size is small, then the data is stored in MFT Entry itself. The attribute list is given in Table below:

Table 1: MFT Entry Attributes With Their Type Identifier

Type Identifier (Decimal	*Type Identifier (Hexadecimal*	*Attribute Name*
16	0x10	$STANDARD_INFORMATION
32	0x20	$ATTRIBUTE_LIST
48	0x30	$FILE_NAME
64	0x40	$VOLUME_VERSION
64	0x40	$OBJECT_ID
80	0x50	$SECURITY_DESCRIPTOR
96	0x60	$VOLUME_NAME
112	0x70	$VOLUME_INFORMATION
128	0x80	$DATA
144	0x90	$INDEX_ROOT
160	0xA0	$INDEX_ALLOCATION
176	0xB0	$BITMAP
192	0xC0	$REPARSE_POINT
192	0xD0	$EA_INFORMATION
208	0xE0	$EA
224	0xF0	$PROPERTY_SET
256	0x100	$LOGGED_UTILITY_ STREAM
---	0xFFFFFFFF	END OF ATTRIBUTES

The details of some important MFT entry attributes are as follows:

$STANDARD INFORMATION Attribute: The type identifier of 16 (0x10H) represents this attribute. It stores temporal information such as flags, owner, security ID and the most important details about timestamps which includes File Creation Time, File Access Time, File Modification Time and MFT Modification Time. File times of $STANDARD INFORMATION are not displayed to the user.

$FILE NAME Attribute: This attribute has the type identifier of 48 (0x30H). It has a definite structure and it contains the file name in Unicode, the size and temporal information as well. Parent directory file reference contains MFT Entry address of parent folder. The existed timestamps in the $FILE NAME attribute are shown to user when the file properties are viewed.

$DATA Attribute: The type identifier of this attribute is 128 (0x80H). Each file has a default $DATA attribute which does not have any special name, additional $DATA attributes must have a name. The folders does not have any $DATA attribute in MFT Entry. $DATA attribute holds files data in a special format named as raw format. $BITMAP Attribute: The $BITMAP metadata file is placed at Entry 6 in MFT and the information about the allocation status of disk clusters are located on its data attribute. The data present in this attribute is organized into 1-byte values and each bit of a data byte represents one cluster. The cluster numbers are interpreted in reverse order. For example, Consider byte with binary values 10000100. Here the LSB value 0 denotes that cluster 0 is not allocated. The allocated status of cluster 8 is denoted by its MSB, which is allocated as it has value 1.

2. Proposed System

The proposed system can be given as a method to consolidate the procedures to recover deleted files such as resident, non-resident files, encrypted files, compressed files etc. It can also be used to detect the origin directory of deleted files. This helps in descending the time consuming efforts of an investigator for depending upon different softwares suited for different types of file recovery. The proposed procedure for deleted file recovery is as follows:

1. Scan the NTFS partitions (drives) and search for the MFT Entries.
2. Identify the MFT entries of the files.
3. Search for the signature value of the file and process its necessary attributes for file recovery .
4. After analysis, find the deleted files and folders through the flag bytes and check its $DATA attribute.
5. If the file is resident, contents are present in the MFT itself. Then copy the contents to the external location for recovery.
6. If the file is non-resident, contents are present in the external clusters. Then parse the runlist.
7. Check allocation status of clusters present in the runlist using $BITMAP attribute.
   (a) If the allocation status of the clusters are 0: complete recovery of files are possible.
   (b) If the allocation status of some clusters are 1: partial recovery of files are possible.
   (c) If the allocation status of all the clusters are 1: recovery of files are not possible.

First scan the NTFS partitions and obtain the MFT. For each user data file, NTFS file system creates a File Record that contained in the MFT. The location of the Master File Table is available in partition boot sector (512 bytes) which is the first sector of the NTFS formatted partition, which can be retrieved using windows API functions.

Then analyse the file's MFT entry and check the flag value details to know about the allocation status of the file. Then process the data attribute header. Data attribute are in two types: resident data attribute and non-resident data attribute. If data attribute is resident, data can be stored in MFT entry itself because of their smaller size. If data attribute is non-resident which stores the information about the data in external clusters which is in special format named as run list. Each MFT record contains different data runs and offsets that hold the actual relevant file content.

RunList are of variable lengths, but it should be considered as one byte. The first byte of runlist is organized into lower 4-bits and upper 4-bits. The lower 4 bits represents the number of bytes to consider in the calculation of length of a single Run. The run is a continuous collection of Cluster Addresses and follows the header byte. The value of upper 4 bits represents the number of bytes used in the calculation of start Offset address of cluster and follows the Length field. To convert the RunList to actual cluster address list, consider the Offset field value as first cluster number and then continue the numbers till it reaches the value specified in Length field.

If $DATA attribute is resident, file content is present in MFT Entry. Just copy it to external location to complete recovery process. If the type of $DATA attribute is non-resident, file's contents are present in external cluster. Check the allocation status (allocated or not) of each cluster present in the Runlist. If all clusters have allocated status as 0, means that the files data is still present on disk. Therefore complete file recovery is possible. If the allocated status of some clusters as 1, means some data become overwritten. So, partial file recovery is possible. Recovery of file is impossible when all the clusters are allocated.

3. Conclusion

NTFS is designed to quickly perform some standard and advanced file operations such as read, write, search and file system recovery on very large hard disks. While conducting an investigation, file recovery is an important step in the investigation process. Deleting files from NTFS computers does not mean that the file is permanently deleted. If the time increases after deletion, the clusters in unallocated space may be overwritten. So, it decreases the chances of successful recovery of files.

This paper proposes to develop a tool that incorporates different processes of file recovery as a single framework which helps in minimizing time to examine the deleted files recovered from the system. It also detects the directory from which the file was deleted.

4. Acknowledgment

We would like to thank, first and foremost, Almighty God, without his support this work would not have been possible. We would also like to thank all the faculty members of college of engineering Kallooppara for their immense support.

References

[1] Sameer H Mahant, B B Meshram *"NTFS Deleted Files Recovery: Forensics View"*, IRACST- International Journal Of Computer Science and Information Technology & Security (IJCSITS), ISSN:2249-9555 Vol.2, No.3, June 2012.
[2] Mr.Dhruv Prajapati, Mr.Anisetti Anjaneyulu,Mr.Nirav Patel,"*Analysis Of Deleted Data In NTFS File System"*, International Journal for Science And Research In Technology (IJSART) volume 1 Issue2–FEBRUARY 2015.
[3] www.NTFS.com. [Online].
[4] Phil Nabity, Brett J L Landry ,"Recovering Deleted And Wiped Files: A Digital Forensic Comparison Of FAT32 And NTFS File Systems Using Evidence Eliminator".
[5] Liu Naiqi, Wang Zhongshan, Hao Yujie, QinKe "*Computer Forensics Research and Implementation Based on NTFS File System"*, ISECS International

Colloquium on Computing, Communication, Control, and Management,2008
[6] Behzad Mahjour Shafiei, Farshid Iranmanesh, Fariborz Iranmanesh*,"Review NTFS Basics"*, Australian Journal of Basic and Applied Sciences, 6(7): 325-338, 2012 ISSN 1991-8178
[7] Jason Medeiros*"NTFS Forensics: A Programmer View Of Raw File system Data Extraction"*,Grayscale Research 2008.

Author Profile

Rincy Roy Oommen obtained the Degree of Bachelor of Technology in Computer Science and Engineering from College of Engineering, Chengannur in 2014. She is now pursuing her master degree in Computer Science with specialization in Cyber Forensics and Information Security at College of Engineering, Kallooppara under Cochin University of Science and Technology.

Princy Sugathan obtained B.Tech in Computer Science and Engineering from College of Engineering Chengannur and M.Tech in Software Engineering from Cochin University. She is currently working as Assistant Professor in College of Engineering, Kallooppara.