Analysis Of Deleted Data In NTFS Filesystem

Mr. Dhruv Prajapati[1], Mr. Anisetti Anjaneyulu[2], Mr. Nirav Patel[3]

[1]Digital Forensics Analyst e-SF Labs LTD

[3]Digital Forensics Analyst e-SF Labs LTD

Abstract *- The most common and must process for the Digital analyst is recovery of deleted data. There are number of operating system in the market but windows is the most used operating system now a days so that is also true that NTFS is the most used Filesystem in current days. This paper is the methodology of how to recovery hierarchical file structure data from the NTFS Filesystem with the help of MFT data entry.*

I. INTRODUCTION

In the of Digital Forensics the Deleted data recovery is the most valuable process of any analysis case. In the current Digital scenario the disk size is rapidly increases by time by time and also the usage and data of the any user is increased and in the most common case user cant simply delete the older data and then used the disk to store new data and also the most common anti forensics technique for any hacker or any of the crime suspect guy is to format the disk or delete any suspicious data. For the recovery of that data there are lots tool available in the market. This paper shows the best technique for how to recover data from NTFS file system and what is the concept behind it.This paper contains information on MFT file structure and what type of data MFT file store and how it is very use full for Forensic analyst to retrieve deleted files.

In the NTFS file system the best thing is analyst can retrieved the file with the hierarchic structure at what point it was deleted so that the whole structure of hierarchical can be retrieved.

In this paper Autopsy tool is used for the analysis. The NTFS file system maintains an index of all files/directories that belong to a directory called the $I30 attribute. Every directory in the file system contains an $I30 attribute that must be maintained whenever there are changes to the files/directories that belong to it. The $I30 index records are re-arranged accordingly as soon as files or folders are removed from the directory. However, re-arranging of the index records may leave remnants of the deleted file/folder entry within the slack space. Similar to Master File Table (MFT) entries in NTFS, index entries within the B-tree are not completely removed when file deletion occurs. Instead, they are marked as deleted using a corresponding $BITMAP attribute. This can be useful in forensics analysis for identifyig files that may have existed on the drive earlier.

II. ANALYSIS PROCESS

The NTFS file system maintains an index of all files/directories that belong to a directory called the $I30 attribute. Every directory in the file system contains an $I30 attribute that must be maintained whenever there are changes to the files/directories that belong to it. The $I30 index records are re-arranged accordingly as soon as files or folders are removed from the directory. However, re-arranging of the index records may leave remnants of the deleted file/folder entry within the slack space. Similar to Master File Table (MFT) entries in NTFS, index entries within the B-tree are not completely removed when file deletion occurs. Instead, they are marked as deleted using a corresponding $BITMAP attribute. This can be useful in forensics analysis for identifying files that may have existed on the drive earlier.

NTFS directory index entries utilize a $FILE_NAME attribute type to store file information within the index. This is the same attribute employed by the MFT and hence it provides a treasure trove of information about the file:

- Full filename
- Parent directory (useful if you recover a $I30 file in free space and do not know its origin)
- File size
- Creation Time
- Modification Time
- MFT Change Time
- Access Time

The Sleuth Kit (TSK) does an excellent job with Index Attributes.

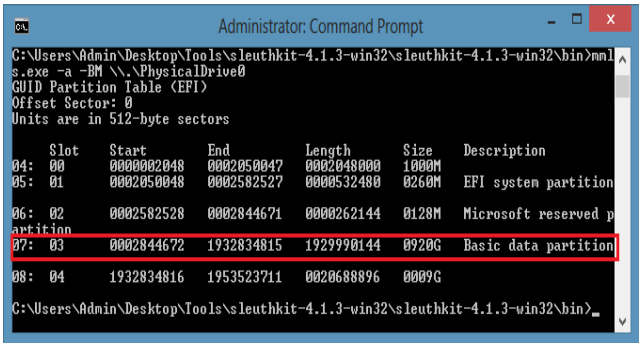


Fig.1 mmls (Sleuthkit)

mmls: Displays the layout of a disk, including the unallocated spaces.

-a denotes Show Allocated Volumes

-B denotes print the rounded length in bytes

-M denotes Hide Metadata volumes

\\.\PhysicalDrive0 is a disk which is evaluated here to get volume details

In fig 1, take slot #03 into account as the Index attribute of a directory we wanted to analyze is of slot #03 and 2844672 is the start sector of selected slot which is basic data partition.

Now select a partition from which we want to find MFT Entry of a file.

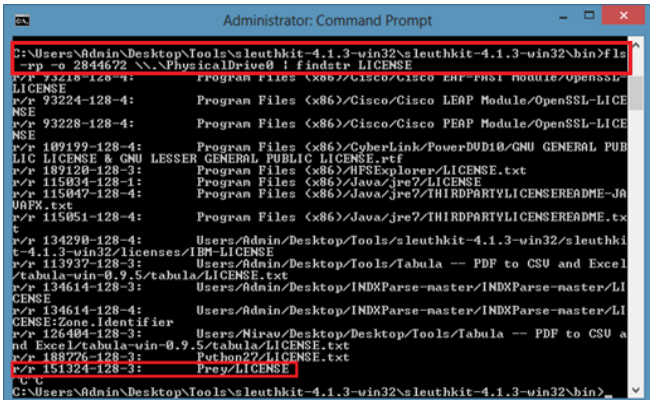


Fig.2 fls (Sleuthkit)

fls : Lists allocated and deleted file names in a directory.

-r denotes recurse on directory entries

-p denotes Display full path for each file

-o denotes imgoffset means Offset into image file (in sectors)

\\.\PhysicalDrive0 is a disk which is evaluated here to get volume details

findstr filename means it searches for given filename starting from given -o imgoffset

r/r denotes file entry in $MFT file

LICENSE is a file located at MFT entry 151324-128-3 in $MFT file which is under the system at path "Prey/LICENSE".

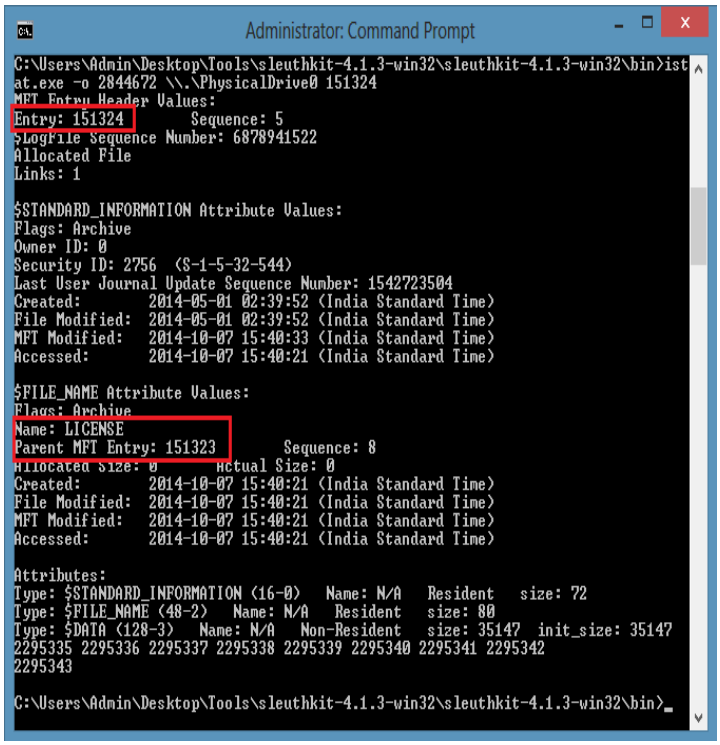


Fig.3 istat (Sleuthkit)

istat : Displays the statistics and details about a given metadata structure in an easy to read format.

Here istat command displays details about given filename at MFT Entry 151324 in $MFT file and also displays parent MFT entry which is 151323.

Figure 4 shows output from the TSK **istat** tool for a "Prey" directory. Near the bottom of the output we see the NTFS attribute list.

istat command displays 0x10 (STANDARD_INFORMATION ATTRIBUTE), 0x30

(FILE_NAME ATTRIBUTE), 0x40 (OBJECT_ID ATTRIBUTE) values for given MFT

Entry.

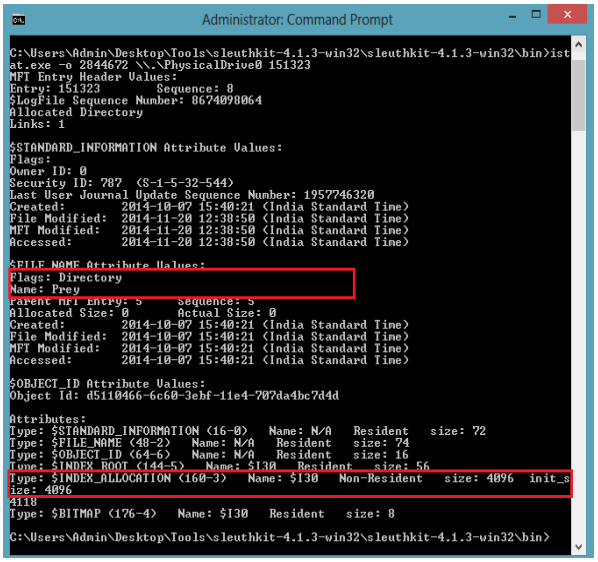


Fig.4 istat (Sleuthkit)

istat command displays $INDEX_ALLOCATION of a directory which is (160-3) for given MFT Entry.

To export the $I30 attribute from this directory, we use the icat tool from TSK and give it the MFT entry number of the directory along with the identifier for the $INDEX_ALLOCATION attribute, which in this case is "*160-3*". This output is redirected into a file named, $I30 which contains file metadata such as physical size, logical size, modified time, accessed time, changed time, created time etc.

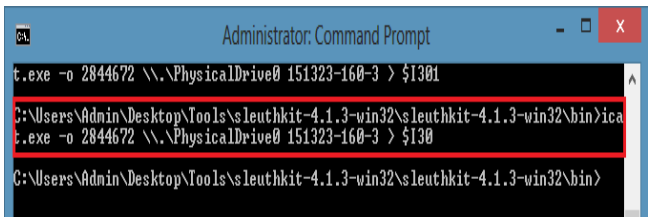


Fig.5 icat (Sleuthkit)

Now we need to parse this INDEX file into easy readable format hence INDXParse.py script is used to convert it into csv format.

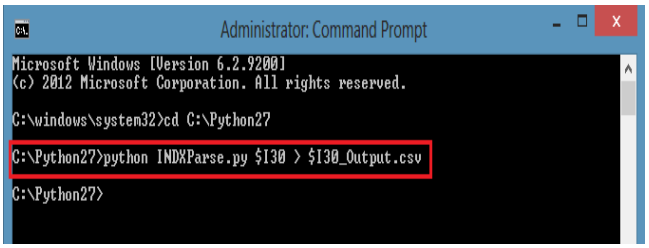
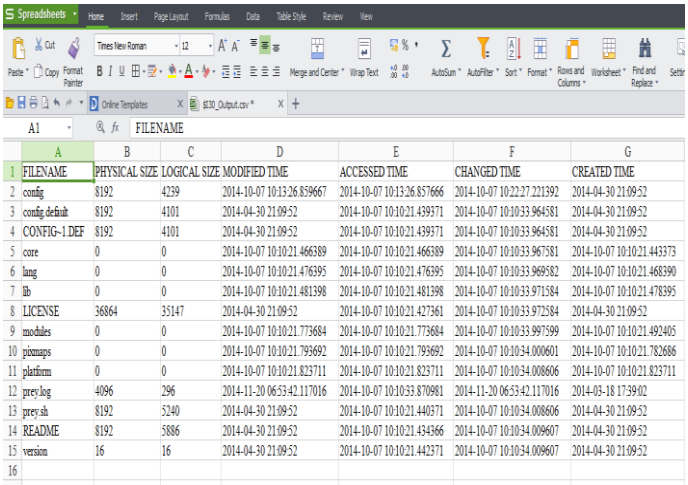


Fig.6 INDXParse.py

In above figure 6, INDXParse.py is a python file used to convert $I30 into csv format.

The resulting file can be opened and filtered in Excel. File names, file size, and four timestamps are displayed in the output shown in below figure 7.



	A	B	C	D	E	F	G
1	FILENAME	PHYSICAL SIZE	LOGICAL SIZE	MODIFIED TIME	ACCESSED TIME	CHANGED TIME	CREATED TIME
2	config	8192	4239	2014-10-07 10:13:26.859667	2014-10-07 10:13:26.857666	2014-10-07 10:22:27.221392	2014-04-30 21:09:52
3	config.default	8192	4101	2014-04-30 21:09:52	2014-10-07 10:10:21.439371	2014-10-07 10:10:33.964581	2014-04-30 21:09:52
4	CONFIG-1.DEF	8192	4101	2014-04-30 21:09:52	2014-10-07 10:10:21.439371	2014-10-07 10:10:33.964581	2014-04-30 21:09:52
5	core	0	0	2014-10-07 10:10:21.466389	2014-10-07 10:10:21.466389	2014-10-07 10:10:33.967581	2014-10-07 10:10:21.443373
6	lang	0	0	2014-10-07 10:10:21.476395	2014-10-07 10:10:21.476395	2014-10-07 10:10:33.969582	2014-10-07 10:10:21.468390
7	lib	0	0	2014-10-07 10:10:21.481398	2014-10-07 10:10:21.481398	2014-10-07 10:10:33.971584	2014-10-07 10:10:21.478395
8	LICENSE	36864	35147	2014-04-30 21:09:52	2014-10-07 10:10:21.427361	2014-10-07 10:10:33.972584	2014-04-30 21:09:52
9	modules	0	0	2014-10-07 10:10:21.773684	2014-10-07 10:10:21.773684	2014-10-07 10:10:33.997599	2014-10-07 10:10:21.492405
10	pixmaps	0	0	2014-10-07 10:10:21.793692	2014-10-07 10:10:21.793692	2014-10-07 10:10:34.000601	2014-10-07 10:10:21.782686
11	platform	0	0	2014-10-07 10:10:21.823711	2014-10-07 10:10:21.823711	2014-10-07 10:10:34.008606	2014-10-07 10:10:21.823711
12	prey.log	4096	296	2014-11-20 06:53:42.117016	2014-10-07 10:10:33.870981	2014-11-20 06:53:42.117016	2014-03-18 17:39:02
13	prey.sh	8192	5240	2014-04-30 21:09:52	2014-10-07 10:10:21.440371	2014-10-07 10:10:34.008606	2014-04-30 21:09:52
14	README	8192	5886	2014-04-30 21:09:52	2014-10-07 10:10:21.434366	2014-10-07 10:10:34.009607	2014-04-30 21:09:52
15	version	16	16	2014-04-30 21:09:52	2014-10-07 10:10:21.442371	2014-10-07 10:10:34.009607	2014-04-30 21:09:52
16							

Fig.7 $I30 into csv format

III. CONCLUSION

This technique is useful to make a proper data recovery tool. It is a method of complete data recovery process of NTFS file with the proper file name. In all other data recovery techniques file will recover in not a folder wise.

REFERENCES

[1] *File System Forensic Analysis*, Brian Carrier

[2]INDXParser.py by Willi Ballenthin

[3]http://digital-forensics.sans.org/blog/2011/09/20/ntfs-i30-index-attributes-evidence-of-deleted-and-overwritten-files

[4]http://www.osforensics.com/faqs-and-tutorials/how-to-scan-ntfs-i30-entries-deleted-files.html

[5]https://www.mandiant.com/blog/striking-gold-incident-response-ntfs-indx-buffers-part-1-extracting-indx/