ISSN No. 0976-5697

Volume 5, No. 4, April 2014 (Special Issue)

International Journal of Advanced Research in Computer Science

RESEARCH PAPER

Available Online at www.ijarcs.info

Identifying the Places of Data Hiding and Ways of Digital Evidence Collection in Different Types of File Systems

Manish H Bhagwani
Chhattissgarh Instt. Of Tech
Rajnandgaon (CG, India)
manishbhagwani1@rediffmail.com

Rajiv V Dharaskar
MPGI Group of Institutions,
Nanded (MS, India)
rvdharaskar@yahoo.com

V. M. Thakare
Sant Gadge Baba Amravati Univesity,
Amravati (MS, India)
vilthakare@yahoo.co.in

Abstract-There are many data hiding techniques available to hide data in various file systems. Digital analyst needs to identify the places, recover the hidden data along with the evidences which can be helpful for investigation. Different file systems are studied for the purpose and procedures for analysis of data are also described.

Keywords-File systems, Digital evidence, Data hiding, Investigation, Digital Analyst

I. INTRODUCTION

Brian Carrier [1] defines *digital investigation* as a process where we develop and test hypotheses that answer questions about digital events. This is done using the scientific method where we develop a hypothesis using evidence that we find and then test the hypothesis by looking for additional evidence that shows the hypothesis is impossible. *Digital evidence* is a digital object that contains reliable information that supports or refutes a hypothesis. Evidence has both legal and investigative uses.

This paper is organized as follows: section 2 describes the ways of finding the existence of deleted file in the file system and ways of recovering file. Section 3 deals with data hiding techniques in NTFS file system and ways of analyzing data to detect and recover the hidden data with evidence.

II. DELETED FILE

Once a file was deleted, the first letter of the file name was lost (overwritten with the hexadecimal character E5h) and the areas of the File Allocation Table which used to point to the storage areas on the disk occupied by the file were set to zero, indicating that those allocation units were available for reuse. According to Geoff H. Fellows [2], files which are placed in the Recycle Bin are not deleted files.

They are current files on the system, which are merely in a state preparatory to being deleted.

Index	Deleted	Path
1	22/03/05 10:05:47	Q:\Images\cooltext8608517.jpg
2	22/03/05 10:05:49	Q:\Images\cf_rabbit.jpg
3	22/03/05 10:05:54	Q:\Images\14-juillet-p.jpg
4	22/03/05 10:06:00	Q:\Images\AMERI214.WMF
5	22/03/05 10:06:03	Q:\Images\DAIRY029.gif

Decoded INFO2 Records

Figure 1: Contents of INFO2 records in Recycle Bin folders

The existence of the data in the INFO2 file in the Recycle Bin folders (INFO on some systems) means it is possible for the analyst to acquire evidence about the date and time of the deletion of the file as well as what its original full path was on the disk. Of course, once the file is cleared from the Recycle Bin it becomes a deleted FAT file like any other and the INFO2 records are cleared, but even then, because the INFO2 file has an ordered structure of records of 280 bytes in length (or 800 under Windows 2000, XP etc.) it is sometimes still possible in these circumstances to recover deleted INFO2 file data and hence valuable evidence.

Under the New Technology File System (NTFS) the evidence gets better still.

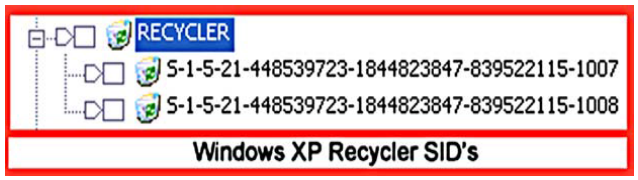

Figure 2: Windows XP Recycler SID's

One reason for this is the fact that the NTFS Recycle Bin is divided up into user Security ID's (SID's), and each user's deleted files are placed into the folder relating to his or her SID. The login names associated with these two SID's can be identified from the Windows Registry files. Of course, each of the SID folders in the Recycle Bin has its very own INFO2 file, and so the details of file deletion in relation to each user logon name is available as well.

When a file is deleted on an NTFS volume two main events occur as far as the file system is concerned. Firstly, the MFT record for the file is marked as relating to a deleted file. Secondly, a system file called $Bitmap is updated to show that the allocation units occupied by the file are available for reuse by the system. Nothing in the MFT record's data (or other) attributes is disturbed or altered (unless the MFT record comes to be overwritten itself).

CONFERENCE PAPER
Two day National Conference on Innovation and Advancement in Computing
Organized by: Department of IT, GITAM UNIVERSITY Hyderabad (A.P.) India
Schedule: 28-29 March 2014

Even where the data has been partially overwritten on an NTFS volume, but the MFT record still survives, the analyst may still be able to recover and to produce useful evidence.

In case mobile phones, many handsets use variants of the FAT file system, originally created by Microsoft for the IBM PC, to maintain media files such as pictures and video clips in NAND flash memory. The differences between the implementations on a handset and on a PC make the recovery of deleted files from the handset more difficult.

Offset	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	
00028672	43	5F	00	75	00	6B	00	00	00	FF	FF	0F	00	21	FF	FF	C_.u.k...ÿÿ..!ÿÿ
00028688	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	00	00	FF	FF	FF	FF	ÿÿÿÿÿÿÿÿÿÿ..ÿÿÿÿ
00028704	02	50	00	61	00	63	00	6B	00	61	00	0F	00	21	67	00	.P.a.c.k.a...!g.
00028720	65	00	20	00	6F	00	32	00	70	00	00	00	6F	00	73	00	e. .o.2.p...o.s.
00028736	01	55	00	73	00	65	00	72	00	20	00	0F	00	21	43	00	.U.s.e.r. ...!C.
00028752	6F	00	6E	00	74	00	65	00	6E	00	00	00	74	00	20	00	o.n.t.e.n...t. .
00028768	55	53	45	52	43	4F	7E	31	20	20	20	63	00	00	00	00	USERCO~1 c....
00028784	00	00	00	00	00	00	85	4E	79	31	25	01	30	00	00	00	.....Ny1u.0...

Figure 3: Directory entry (Nokia 6230) with ”initial” bytes (orange), attributes (green), SC (yellow), modification-time (pink), modification-date (brown) and file-size (blue): Before deletion

If the entry in fig. 3 were to be deleted (as in fig. 4), the initial byte in every block making up the directory entry would be overwritten with 0xE5.

Offset	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	
00028672	E5	5F	00	75	00	6B	00	00	00	FF	FF	0F	00	21	FF	FF	å_.u.k...ÿÿ..!ÿÿ
00028688	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	00	00	FF	FF	FF	FF	ÿÿÿÿÿÿÿÿÿÿ..ÿÿÿÿ
00028704	E5	50	00	61	00	63	00	6B	00	61	00	0F	00	21	67	00	åP.a.c.k.a...!g.
00028720	65	00	20	00	6F	00	32	00	70	00	00	00	6F	00	73	00	e. .o.2.p...o.s.
00028736	E5	55	00	73	00	65	00	72	00	20	00	0F	00	21	43	00	åU.s.e.r. ...!C.
00028752	6F	00	6E	00	74	00	65	00	6E	00	00	00	74	00	20	00	o.n.t.e.n...t. .
00028768	E5	53	45	52	43	4F	7E	31	20	20	20	63	00	00	00	00	åSERCO~1 c....
00028784	00	00	00	00	00	00	85	4E	79	31	00	00	30	00	00	00	.....Ny1u.0...

Figure. 4: Directory entry (Nokia 6230) with ”initial” bytes (orange), attributes (green), SC (yellow), modification-time (pink), modification-date (brown) and file-size (blue): After deletion, with overwritten initial byte and SC

A. Fragmented Files:

Garfinkel [3] presented statistics about the incidence of file fragmentation on actual file systems recovered from used hard drives purchased on the secondary market. The kinds of fragmentation patterns observed on those drives are representative of fragmentation patterns found in drives of forensic interest. Analysis of Garfinkel corpus was performed using Carrier’s Sleuth Kit [4] and a file walking program that was specially written for this project.

III. NTFS FILE SYSTEM

Ewa Huebner (et. al.2006) [5] discussed the methods of hiding data in the NTFS file system. Analysis techniques which can be applied to detect and recover data hidden using each of these methods are also discussed.

A. Hiding Data In $Data Attribute:

Most of the metadata files, except the directories and the extension metadata files already contain the $DATA attribute. It is possible to append some hidden data to these $DATA attributes without disturbing the data already present. To discover whether there is any data hidden in this manner the investigator should check the number of clusters allocated to each $DATA attribute using NTFS File Sector Information Utility Nfi (Microsoft OEM).

B. Hiding Data In $Boot File:

In NTFS the boot record is stored in a metadata file called $Boot. This is the only file with a fixed location; it always starts at the first cluster of the file system. Windows allocates 16 sectors to this file but typically only half of these sectors contain non-zero bits (Carrier[1]). According to the NTFS documentation at the Linux NTFS project (Provos and Honeyman [6]), there are certain unused bytes in the boot sector. However, Windows will not mount the file system if there are any non-zero values in these unused bytes. As a result, this space cannot be used to hide data.
However, the bytes allocated to boot code in the $Boot file of NTFS file system can be used to hide data. Boot code is only essential for a bootable file system to locate files required to boot up Windows (Carrier[1]). The analysis of hidden data in the $Boot file should start by comparing the boot sector and the backup boot sector.
The consistency check would only give an indication of file system manipulation if the checksums differ from each other. If they are the same, there is still a possibility that the modified boot sector was copied to its backup to avoid detection. For this reason the extracted content should still be analyzed with a hex editor.

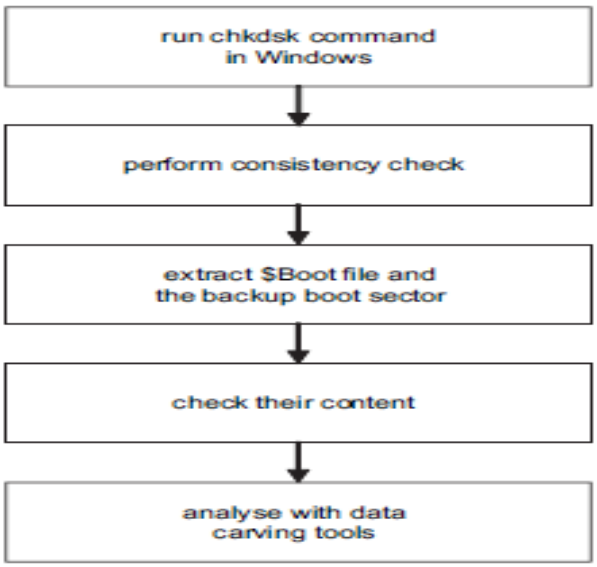

Figure 5 – Analysis of hidden data in $Boot file and the backup boot sector.

C. Hiding Data In $Badclus File:

Modern hard disk controllers handle bad sectors themselves without any involvement of the operating system. According to Schindler et al., [7] typically one of the following two ways is used: slipping (modifying Logical Block Number (LBN) to physical mapping to skip the defective sector) or remapping (reallocating LBN from defective area to a spare sector). In addition the volume managers included with Windows are capable of remapping bad sectors, so called sector sparring, and even recovering data on fault tolerant systems [10]. Fig. 3 shows the flowchart for detection of hidden data in faked bad clusters.

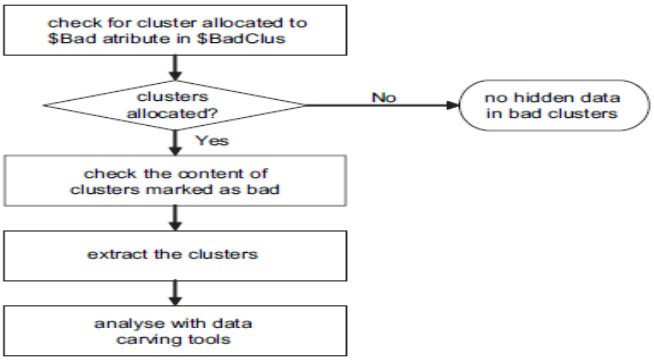

Figure 6: Detection of Hidden Data in Faked Bad Clusters

According to the NTFS documentation at the Linux NTFS project Provos and Honeyman [6], there are certain unused bytes in the boot sector. However, Windows will not mount the file system if there are any non-zero values in these unused bytes [1]. As a result, this space cannot be used to hide data. The analysis of hidden data in the $Boot file should start by comparing the boot sector and the backup boot sector.

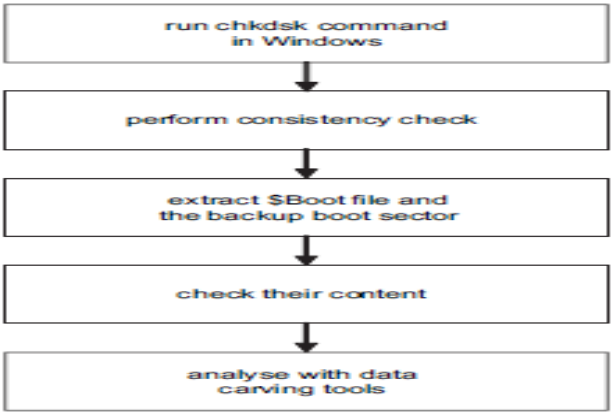

Figure 7 – Analysis of hidden data in $Boot file and the backup boot sector.

IV. CONCLUSION

Data hiding methods in different file systems help to understand the ways in which culprits hide their data. The methods of digital diagnosis are suggested here which will help analyst to produce evidences.

V. REFERENCES

[1]. Brian Carrier, "File System Forensic Analysis", Addison Wesley Professional [Book], March 17, 2005, ISBN: 0-32-126817-2.

[2]. Geoff H. Fellows, "The joys of complexity and the deleted file", Digital Investigation (Elsevier), Vol. 2, pp: 89-93, February 2005.

[3]. Garfinkel Simson, "Forensic feature extraction and cross-drive analysis", Digit Investigation (Elsevier), http://www.dfrws.org/2006/proceedings/10-Garfinkel.pdf, pp: 71-81, August, 2006.

[4]. Carrier Brian, "The Sleuth Kit & Autopsy: forensics tools for Linux and other Unixes", http://www.sleuthkit.org, 2005.

[5]. Ewa Huebner, Derek Bem, Cheong Kai Wee, "Data hiding in the NTFS file system", Digital Investigation (Elsevier), pp. 211-226, March 2006.

[6]. Provos N, Honeyman P, "Detecting steganographic content on the internet", pp. 13.

[7]. Schindler J, Griffin JL, Lumb CR, Ganger GR, "Track-aligned extents: matching access patterns to disk drive characteristics", Conference on file and storage technologies (FAST), Monterey, CA, USA, pp. 16, 2002.

[8]. Russinovich ME, Solomon DA, "Microsoft Windows Internals", Redmond: Microsoft Press[Book], 2005.

CONFERENCE PAPER
Two day National Conference on Innovation and Advancement in Computing
Organized by: Department of IT, GITAM UNIVERSITY Hyderabad (A.P.) India
Schedule: 28-29 March 2014