Global Information Assurance Certification Paper

Analyze an Unknown Image and Perform Forensic Tool Validation

GIAC Certified Forensic Analyst

Practical Assignment

Version 1.5

Patricia Watson
Systems Forensics, Investigation & Response
Monterey California
July 2004

Table of Contents

List of Figures

List of Tables

Abstract

The purpose of this paper is to complete the practical assignment Version 1.5 at a technically proficient level as part of the Global Information Assurance Certification (GIAC) for Certified Forensic Analyst (GCFA). The practical assignment consists of two parts: (i) the analysis of an unknown image and (ii) forensic tool validation.

The objective of the first part is to analyze a suspicious floppy using a collection of forensics tools and methodologies. As with any forensics investigation, while performing analysis of the unknown image, careful precautions will be taken to ensure integrity of any evidence collected is not jeopardized. Upon completion of the analysis, the end results will be thoroughly summarized. In addition, any relevant legal implications will be discussed.

The goal of the second part is to perform a forensic tool validation to determine if the chosen tool can be used during a forensics investigation. The justification of the tool consists of obtaining repeatable and reproducible results. Further, if the tool is forensically sound, it should not jeopardize evidence integrity. *Hurricane Search*, a text search tool is the program chosen for this validation.

Document Conventions

In this practical assignment, certain words are represented in different fonts and typefaces. The types of words that are represented this way include the following:

- **`Command`** - Operating system commands are represented in this font style. This style indicates a command that is entered at a command prompt or shell.
- ***`Filename`*** - Filenames, paths, and directory names are represented in this style.
- *`Program names`* - The results of a command and other computer output are in this style
- URL - Web URL's are shown in this style.
- *"Quotation"* - A citation or quotation from a book or web site is in this style.

Part 1 – Analyze an Unknown Image

Synopsis of Case

The primary purpose of Part 1 is to analyze a floppy disk that has been seized from Robert John Leszczynski, Jr., the lead process control engineer responsible for the production of a fuel cell battery at Ballard Industries. Although Ballard Industries is the proprietor of the fuel cell battery, it has been recently discovered that a competitor, Rift, Inc., has been producing the same fuel cell battery. During an internal investigation into the apparent loss of proprietary information, security records indicated that a floppy disc had been seized from Robert Leszczynski. Mr. David Keen, the Security Administrator at Ballard Industries, requested a forensic analysis of the floppy disc.

Upon completion of the forensics analysis, all findings will be reported to Mr. David Keen, Security Administrator at Ballard Industries. As the forensics analyst on this case, I must ensure during the course of this analysis that the forensic techniques applied to the floppy disc do not corrupt the digital evidence being analyzed. A mathematical function, known as a hash value, which acts like an electronic fingerprint, was used for integrity confirmation and timestamping throughout the investigation. An algorithm, **`MD5Sum`**, was utilized to calculate the hash value of the image and any files obtained during the investigation to ensure evidence integrity. **`MD5Sum`** produces a 128-bit hash value of the file or image, which is effectively impossible for current computing devices to produce the same hash value of a different file. Further, legal protocol requires proper chain of custody procedures be followed and documented. As such, a digital image of the seized floppy was obtained and the original floppy provided by Mr. Keen has been locked in a secured location to prevent evidence tampering and to maintain an irrefutable chain of custody. Figure 1 illustrates the Chain of Custody form for this forensic analysis.

Chain of Custody Form		
CaseID		**Description**
LeszcynskiV1.5		SANS GCFA version 1.5
Released By	David Keen, Security Administrator Ballard Industries	
Received By	Patricia Watson, Forensic Analyst, GCFA	
TagNumber	**Description of Evidence**	**Image/File Name**
fl-260404-RJL1	3.5 inch TDK floppy disk	fl-260404-RLJ.img.gz
MD5 Hash Value of Image/File Name if Available		
d7641eb4da871d980adbe4d371eda2ad		
Date and Time Item was Seized		**Location**
26 April 2004 4:45pm MST		R&D Labs at Ballard Industries
Name/Names of Individual(s) the Custody Item(s) was/were Obtained From		
Robert Leszcynski		
Name/Names of Individual(s) who Collected the Custody Item(s)		
Staff security guard (name not available)		

Figure 1 – Chain of Custody Form

Preparation Details

To further ensure integrity and validity of evidence analysis, a dedicated forensic workstation was used to perform the forensic analysis. The forensic workstation consists of a *Linux* machine as the main host which contains a *VMware* hosting *Windows XP* operating system. *VMware* is a software package which allows users the capability of running multiple operating systems in one machine simultaneously. By using *VMware*, the forensic analyst has greater functionality throughout the analysis, verification, and examination of the digital evidence given that the investigator can readily access more than one operating system concurrently.

The analysis of the image was performed using a *Linux* computer which consists of the following:

- Operating System: GNU/Linux Gentoo 2.6.5
- Hard Drive Capacity: 40 GB
- Processor: Pentium 4 2.40 GHz
- Memory: 1 GB
- Forensic Tools: Autopsy Forensic Browser 2.0

The program identification and verification was performed using the *Windows XP VMware* which consists of the following:

- Operating System: *Microsoft Windows XP Professional*

Version 2002, Service Pack 1
- Hard Drive Capacity: 5 GB
- Processor: Pentium 4 2.40 GHz
- Memory: 256 MB
- Tools: *WinHex 11.8, Google, HashCalc 2.01, Cygwin*

Prior to digital imaging, to make certain evidence was not corrupted by any external factors, a dedicated hard drive partition was created and sanitized using the *Linux* command **disk dump (dd)**. Confirmation of the sanitized partition, **/dev/hda8/**, is shown in Figure 2.

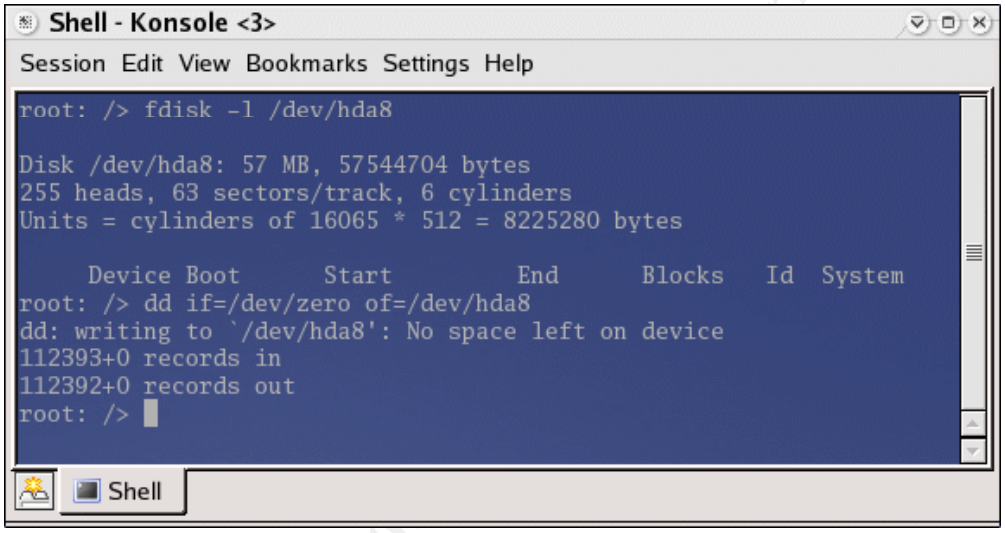


Figure 2 - Screen shot indicating partition ***/dev/hda8*** has been sanitized.

In a computer forensic investigation, it is crucial to leave the original evidence intact from any forensic techniques used during the investigation. For this reason, an image of the original floppy was performed. Once the working partition was sanitized, a digital image of the original floppy was copied and assigned to ***/dev/hda8*** by using the *Linux* command **dcfldd**. The sequence of commands used to copy the digital image to the hard drive and the resulting output is shown in Figure 3.

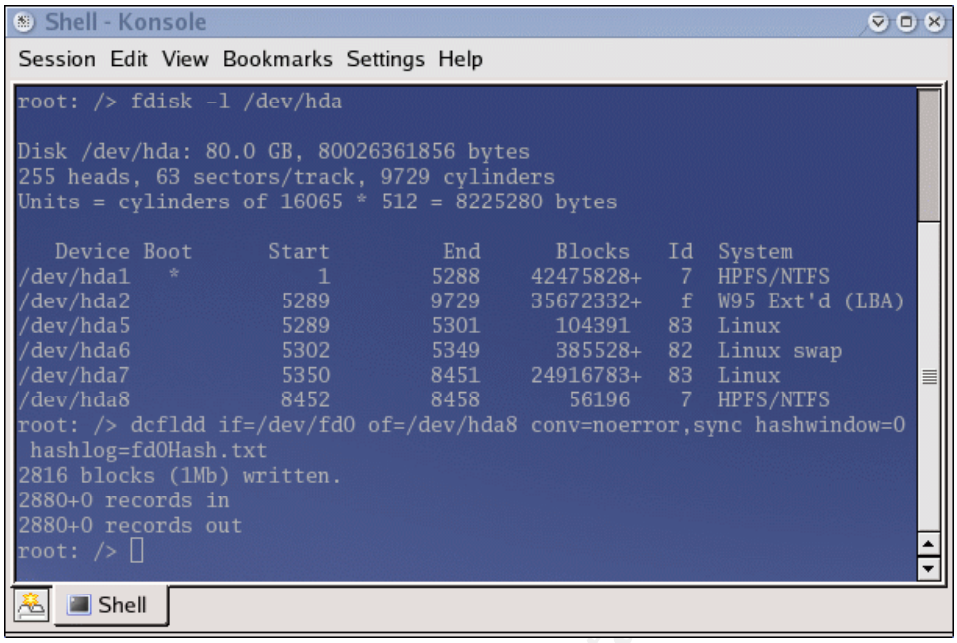


Figure 3 - Screen shot of the steps used to create an image of the seized floppy disc on to the hard drive partition, ***/dev/hda8***.

To verify the digital image was the exact copy of the original floppy, a hash of the image was obtained using **MD5Sum**. Because **MD5sum** is a hash value of the contents of a file or image, and it is repeatable and non-reversible, the unique resulting hash value can be use to validate the integrity when comparing the copy of the image from the original floppy.

Forensic Details

Once the integrity of the image was verified, *Autopsy Forensic Browser Version 2.00* included in the *Sleuthkit Forensic suite* was used to analyze the image and begin evidence collection. *Autopsy* is an Open Source *HTML*-based graphical interfaced to *Linux* command line tools (Figure 4).

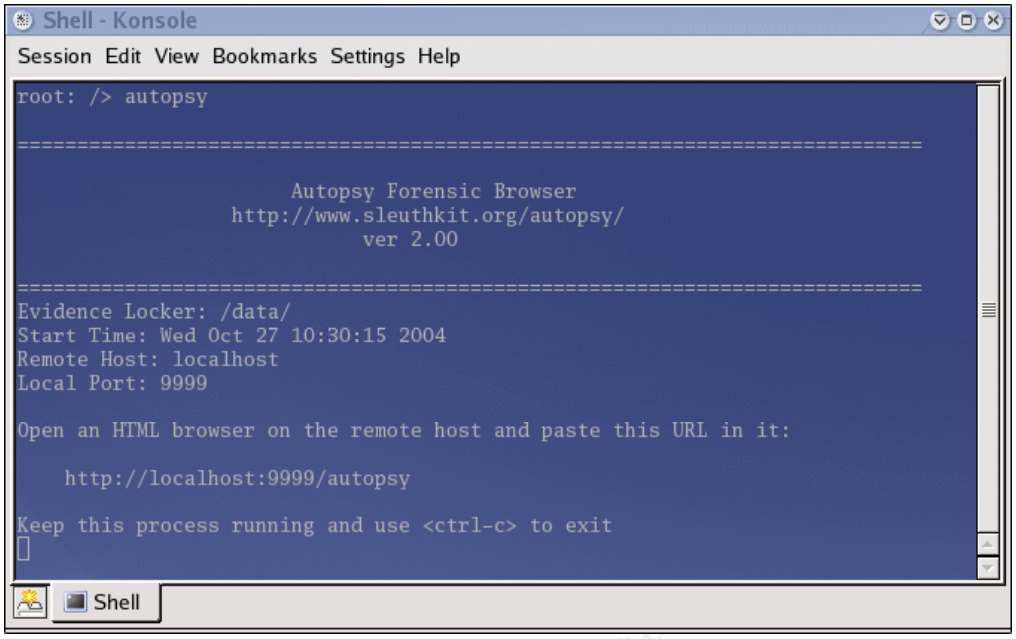


Figure 4 – Screen shot of *Autopsy* Version 2.0

File Analysis on *Autopsy* reveals the image of the floppy disc contains the following files:

- ***_ndex.htm***
- ***Acceptable_Encryption_Policy.doc***
- ***CamShell.dll***
- ***Information_Sensitivity_Policy.doc***
- ***Internal_Lab_Security_Policy.doc***
- ***Internal_Lab_Security_Policy1.doc***
- ***Password_Policy.doc***
- ***Remote_Access_Policy.doc***

Using *Autopsy*, an **MD5Sum** hash value was obtained for each file contained in the image. This step is performed to validate the integrity of each file and prove that none of the files were compromised during the analysis (Figure 5).

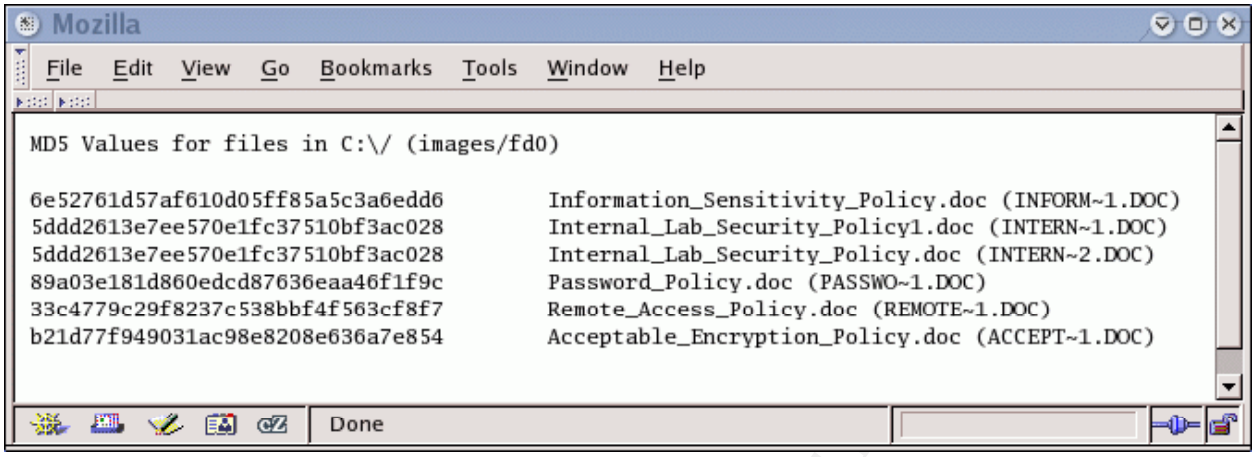


Figure 5 – Screen shot of `MD5` hash values for each file contained in image

In a computer forensic analysis, the window of time in which the incident transpired is very important. By using a combination of *`Autopsy's`* File Analysis, Image Details, Meta Data, and File Activity Time Line, the information summarized in Table 1 was obtained. Floppy disks are formatted based on the FAT12 file system, which is the oldest flavor of the FAT family. Each file generated in a FAT file system stores up to three date codes known as Accessed Time, Created Time, and Written Time. During a forensic analysis, retrieved timestamps from the image being analyzed can be used as digital "tracks" of activities that transpired during a given period of time. However, timestamps have limitations which should be taken into consideration to avoid misinterpretation of events. One of the biggest limitations of timestamps is that they can be easily modified. On a *`Microsoft Windows`* platform, a simple task such as listing the contents of a file changes the file's access timestamp. Further, with the use of free tools and commands which are readily available on the internet, adversaries can modify timestamps to cover their tracks. For example, the timestamps of a file can be modified using the **`utimes()`** C library function. While timestamps can be useful in constructing a precursory timeline, they can't be trusted as conclusive evidence.

Table 1 – Summary of image details

Deleted	File Name	Date Written	Date Accessed	Date Created	Size (in bytes)	Owner	Group	All	UID	GID	META	Starting Sector	Total Sectors	Ending Secor
Y	_ndex.htm	04/23/04	04/26/04	04/26/04	727				0	0	28	33	1.42	34.42
	Acceptable_Encryption_Policy.doc	04/23/04	04/26/04	04/26/04	22528	r;w;x	r	r	root	root	27	1341	44.00	1385
Y	CamShell.dll	02/03/01	04/26/04	04/26/04	36864				0	0	5	33	72.00	105
	Information_Sensitivity_Policy.doc	04/23/04	04/26/04	04/26/04	42496	r;w;x	r	r	root	root	9	105 1631	83.00	187 1631
	Internal_Lab_Security_Policy.doc	04/22/04	04/26/04	04/26/04	33423	r;w;x	r	r	root	root	17	251	65.28	316.3
	Internal_Lab_Security_Policy1.doc	04/22/04	04/26/04	04/26/04	32256	r;w;x	r	r	root	root	13	188	63.00	251
	Password_Policy.doc	04/23/04	04/26/04	04/26/04	307935	r;w;x	r	r	root	root	20	317	601.44	918.4
	Remote_Access_Policy.doc	04/23/04	04/26/04	04/26/04	215895	r;w;x	r	r	root	root	23	919	421.67	1341

Based on the last written time, the chronological order of the image details are

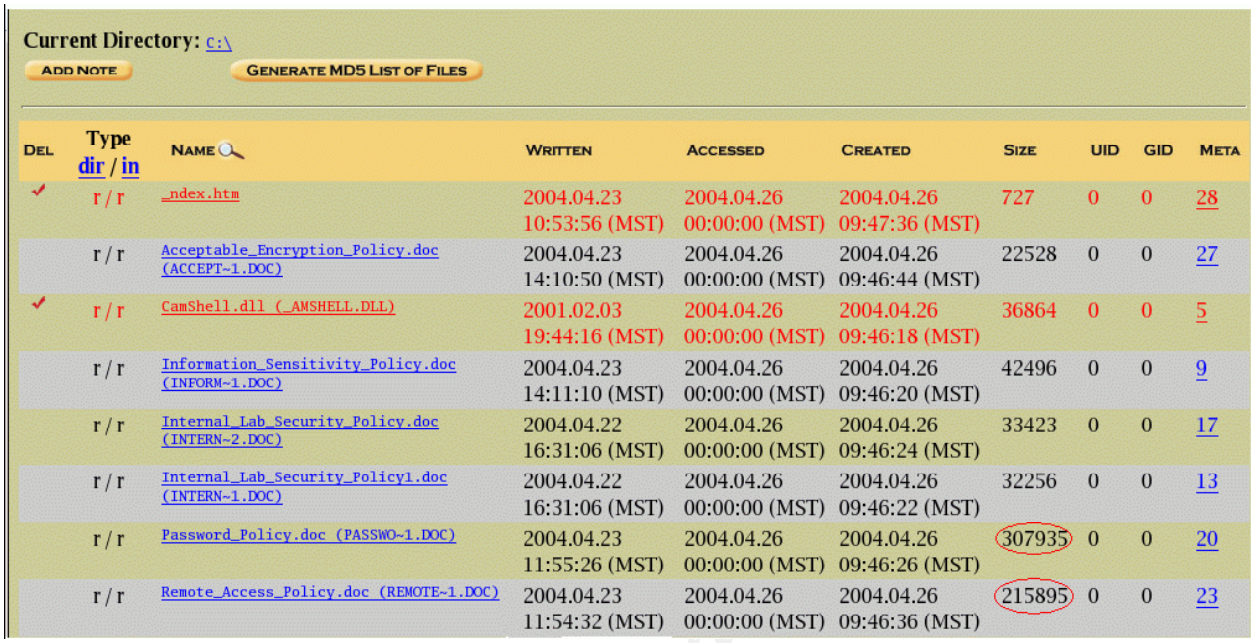
as follows:

- ***CamShell.dll*** is 36,864k, was last written on February 3rd of 2001, created on April 26, 2004 and last accessed on April 26, 2004.
- Both ***Internal_Lab_Security_Policy.doc*** and ***Internal_Lab_Security_Policy1.doc*** were last written on April 22, 2004, created and last accessed on April 26, 2004.
- ***Internal_Lab_Security_Policy.doc*** is 33,423 KB and ***Internal_Lab_Security_Policy1.doc*** is 32,256 KB in size***._ndex.htm*** size 727 KB, ***Acceptable_Encryption_Policy.doc*** size 22,528 KB, ***Information_Sensitivity_Policy.doc*** size 42,496 KB, ***Password_Policy.doc*** size 307,935 KB, and ***Remote_Access_Policy.doc*** size 32,256 KB were all last written on April 23, 2004, created and last accessed on April 26, 2004.

It is important to note that all non-deleted files on this image have *read* (r), *write* (w), and *execute* (x) owner permissions. Further, all files have *read* (r) *group* and *all* (all other users including guest users) permissions. Finally, all non-deleted files have ***root*** (Administrator) as file owner which gives users system administrator privileges including read, write and execute permissions, which defeat the read-only permissions set for group and all. Files with administrator privileges can be easily exploited by hackers or dishonest employees, because they can easily be remotely accessed and manipulated when networked either on a local are network (LAN) or to the Internet.

Deleted Files

In computer forensics, thorough analysis of deleted files contained within the image in question is an integral part of the investigation, particularly since they can be an indication of foul play. In general, recovering deleted files during a computer forensics investigation is fairly easy, especially when using forensic tools, such as, *Autopsy*. With *Autopsy*, deleted files are displayed in red with a check mark under the deleted column. As shown in Figure 6, both ***_ndex.htm*** and ***CamShell.dll*** are deleted files.



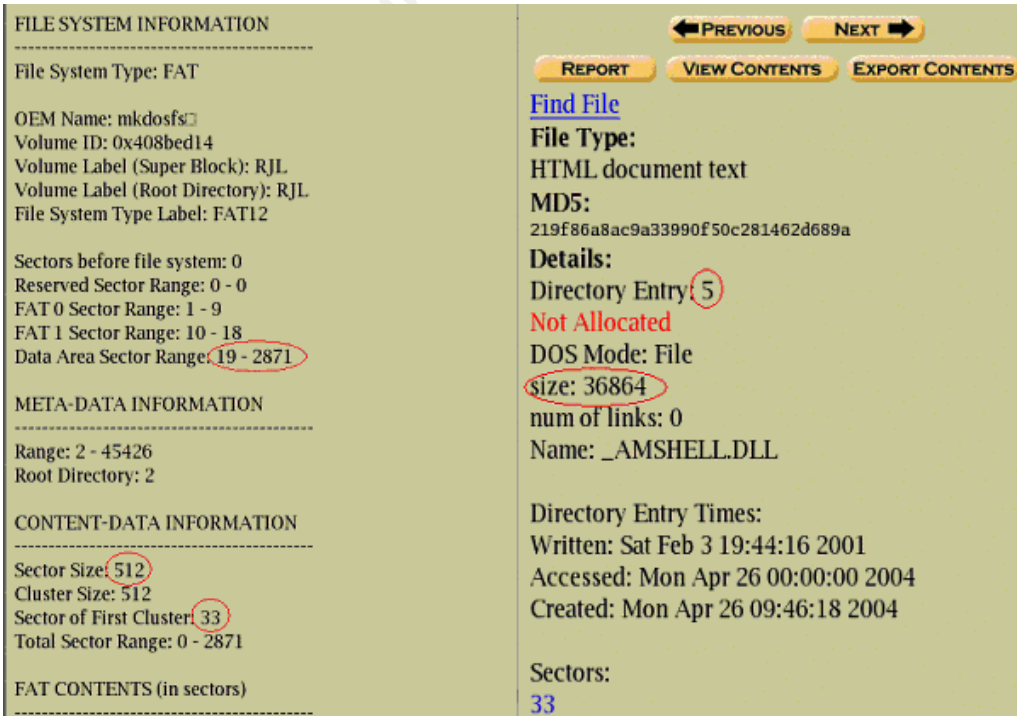
Current Directory: C:\

ADD NOTE | GENERATE MD5 LIST OF FILES

DEL	Type dir / in	NAME	WRITTEN	ACCESSED	CREATED	SIZE	UID	GID	META
✓	r / r	_ndex.htm	2004.04.23 10:53:56 (MST)	2004.04.26 00:00:00 (MST)	2004.04.26 09:47:36 (MST)	727	0	0	28
	r / r	Acceptable_Encryption_Policy.doc (ACCEPT~1.DOC)	2004.04.23 14:10:50 (MST)	2004.04.26 00:00:00 (MST)	2004.04.26 09:46:44 (MST)	22528	0	0	27
✓	r / r	CamShell.dll (_AMSHELL.DLL)	2001.02.03 19:44:16 (MST)	2004.04.26 00:00:00 (MST)	2004.04.26 09:46:18 (MST)	36864	0	0	5
	r / r	Information_Sensitivity_Policy.doc (INFORM~1.DOC)	2004.04.23 14:11:10 (MST)	2004.04.26 00:00:00 (MST)	2004.04.26 09:46:20 (MST)	42496	0	0	9
	r / r	Internal_Lab_Security_Policy.doc (INTERN~2.DOC)	2004.04.22 16:31:06 (MST)	2004.04.26 00:00:00 (MST)	2004.04.26 09:46:24 (MST)	33423	0	0	17
	r / r	Internal_Lab_Security_Policy1.doc (INTERN~1.DOC)	2004.04.22 16:31:06 (MST)	2004.04.26 00:00:00 (MST)	2004.04.26 09:46:22 (MST)	32256	0	0	13
	r / r	Password_Policy.doc (PASSWO~1.DOC)	2004.04.23 11:55:26 (MST)	2004.04.26 00:00:00 (MST)	2004.04.26 09:46:26 (MST)	307935	0	0	20
	r / r	Remote_Access_Policy.doc (REMOTE~1.DOC)	2004.04.23 11:54:32 (MST)	2004.04.26 00:00:00 (MST)	2004.04.26 09:46:36 (MST)	215895	0	0	23

Figure 6 – Screen shot of *Autopsy's* **File Analysis** of the image

The Image Details from *Autopsy*, shown in Figure 7, indicate cluster and sector sizes of 512 bytes, respectively. In addition, with the use of the *Autopsy* Meta Data option, the beginning sector for each corresponding file on the image was determined (Figure 7).



FILE SYSTEM INFORMATION

File System Type: FAT

OEM Name: mkdosfs
Volume ID: 0x408bed14
Volume Label (Super Block): RJL
Volume Label (Root Directory): RJL
File System Type Label: FAT12

Sectors before file system: 0
Reserved Sector Range: 0 - 0
FAT 0 Sector Range: 1 - 9
FAT 1 Sector Range: 10 - 18
Data Area Sector Range: 19 - 2871

META-DATA INFORMATION

Range: 2 - 45426
Root Directory: 2

CONTENT-DATA INFORMATION

Sector Size: 512
Cluster Size: 512
Sector of First Cluster: 33
Total Sector Range: 0 - 2871

FAT CONTENTS (in sectors)

PREVIOUS | NEXT

REPORT | VIEW CONTENTS | EXPORT CONTENTS

Find File

File Type:
HTML document text

MD5:
219f86a8ac9a33990f50c281462d689a

Details:
Directory Entry: 5
Not Allocated
DOS Mode: File
size: 36864
num of links: 0
Name: _AMSHELL.DLL

Directory Entry Times:
Written: Sat Feb 3 19:44:16 2001
Accessed: Mon Apr 26 00:00:00 2004
Created: Mon Apr 26 09:46:18 2004

Sectors:
33

Figure 7 – Screen shot of *Autopsy's* Image Details

Based on this initial examination, all relevant information from ***_ndex.htm*** and ***CamShell.dll*** are summarized in Table 1.

Table 2 - Information extracted from ***_ndex.htm*** and ***CamShell.dll***

File Name	Date Written	Date Accessed	Date Created	File Size	Meta	Starting Sectors	Total Sectors	Ending Sectors
_ndex.htm	4/23/04	4/23/04	4/26/04	727	28	33	1.42	34.42
CamShell.dll	2/03/01	4/26/04	4/26/04	36,864	5	33	72	105

The total number of sectors per file was determined by dividing the sector size (512 bytes) by the file size. The ending sector for each file was obtained by taking the beginning sector and adding the total sectors. Because both deleted files begin on sector 33, it is concluded that ***_ndex.htm*** resides over the first 727 bytes of ***CamShell.dll***. The **written date** and the *Meta* suggest that ***CamShell.dll*** was the first file created on this image. The file was then deleted, which resulted in the un-allocation of the allocated space for that file on the image. As a result, ***_ndex.htm*** was allocated on the first 727 bytes of the 36,864 KB of ***CamShell.dll***. An example of the resulting overlap is shown in Figure 8.

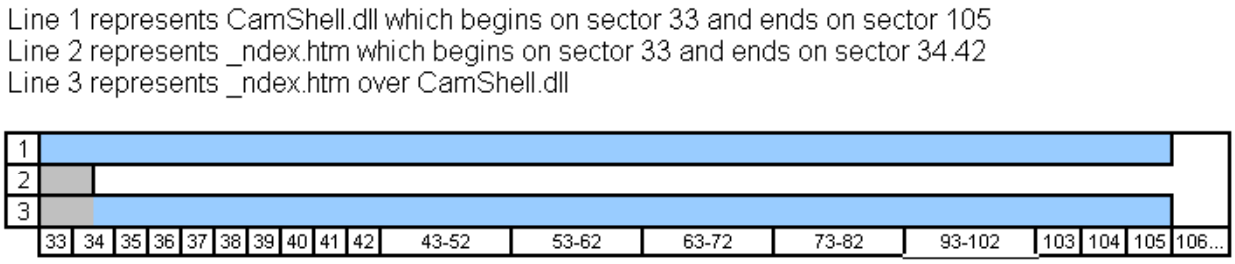


Figure 8 – Example of overlapping files

In order to examine the contents of sectors 33 through 105, the beginning sector and the total sectors were revealed using `Autopsy`. The far left column in Figure 9 shows the corresponding offset bytes, the middle column contains the hexadecimal (HEX) representation of the file contents, and the far right column displays the file contents in American Standard Code for Information Interchange (ASCII).

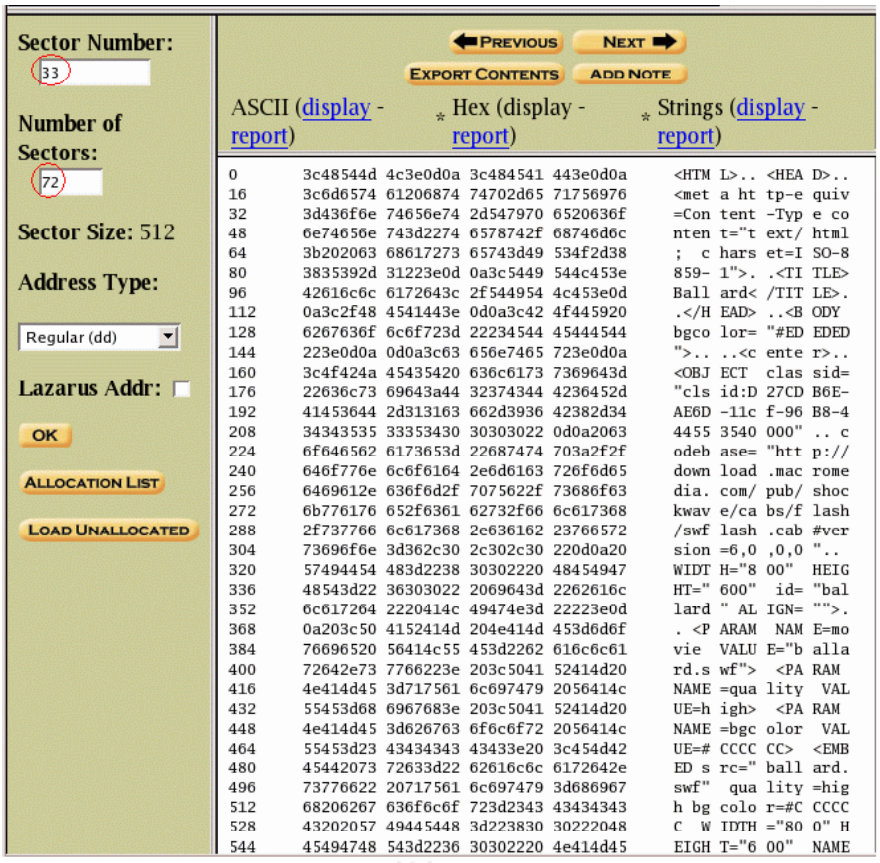


Figure 9 – Screen shot of *Autopsy's* hexadecimal display of sector 33 through sector 105

As previously stated, the first 727 bytes contain the contents of ***_ndex.htm*** and the remaining 36,137 KB contains the residue of ***CamShell.dll***. Based on the information extracted from latent data (Figure 10), it can be concluded that Mr. Leszczynski downloaded the software package known as *Camouflage* (Version 1.0.4) from http://www.camouflage.freeserve.co.uk.

```
29248   01004300 6f006d00 6d006500 6e007400     ..C. o.m. m.e. n.t.
29264   73000000 68007400 74007000 3a002f00     s... h.t. t.p. :./.
29280   2f007700 77007700 2e006300 61006d00     /.w. w.w. ..c. a.m.
29296   6f007500 66006c00 61006700 65002e00     o.u. f.l. a.g. e...
29312   66007200 65006500 73006500 72007600     f.r. e.e. s.e. r.v.
29328   65002e00 63006f00 2e007500 6b000000     e... c.o. ..u. k...
29344   54003200 01004300 6f006d00 70006100     T.2. ..C. o.m. p.a.
29360   6e007900 4e006100 6d006500 00000000     n.y. N.a. m.e. ....
29376   54007700 69007300 74006500 64002000     T.w. i.s. t.e. d. .
29392   50006500 61007200 20005000 72006f00     P.e. a.r.  .P. r.o.
29408   64007500 63007400 69006f00 6e007300     d.u. c.t. i.o. n.s.
29424   00000000 b0008800 01004600 69006c00     .... .... ..F. i.l.
29440   65004400 65007300 63007200 69007000     e.D. e.s. c.r. i.p.
29456   74006900 6f006e00 00000000 4b006500     t.i. o.n. .... K.e.
29472   65007000 73002000 66006900 6c006500     e.p. s. . f.i. l.e.
29488   73002000 63006f00 6e007400 61006900     s. . c.o. n.t. a.i.
29504   6e006900 6e006700 20007300 65006e00     n.i. n.g.  .s. e.n.
29520   73006900 74006900 76006500 20006900     s.i. t.i. v.e.  .i.
29536   6e006600 6f007200 6d006100 74006900     n.f. o.r. m.a. t.i.
29552   6f006e00 20007300 61006600 65002000     o.n.  .s. a.f. e. .
29568   66007200 6f006d00 20007000 72007900     f.r. o.m.  .p. r.y.
29584   69006e00 67002000 65007900 65007300     i.n. g. . e.y. e.s.
29600   2e000000 cc00a800 01004c00 65006700     .... .... ..L. e.g.
29616   61006c00 43006f00 70007900 72006900     a.l. C.o. p.y. r.i.
29632   67006800 74000000 43006f00 70007900     g.h. t... C.o. p.y.
29648   72006900 67006800 74002000 28006300     r.i. g.h. t. . (.c.
29664   29002000 32003000 30003000 2d003200     ). . 2.0. 0.0. -.2.
29680   30003000 31002000 62007900 20005400     0.0. 1. . b.y.  .T.
29696   77006900 73007400 65006400 20005000     w.i. s.t. e.d.  .P.
29712   65006100 72002000 50007200 6f006400     e.a. r. . P.r. o.d.
29728   75006300 74006900 6f006e00 73002c00     u.c. t.i. o.n. s.,.
29744   20004100 6c006c00 20007200 69006700      .A. l.l.  .r. i.g.
29760   68007400 73002000 72006500 73006500     h.t. s. . r.e. s.e.
29776   72007600 65006400 20007700 6f007200     r.v. e.d.  .w. o.r.
29792   6c006400 77006900 64006500 2e000000     l.d. w.i. d.e. ....
29808   38001600 01005000 72006f00 64007500     8... ..P. r.o. d.u.
29824   63007400 4e006100 6d006500 00000000     c.t. N.a. m.e. ....
29840   43006100 6d006f00 75006600 6c006100     C.a. m.o. u.f. l.a.
29856   67006500 00000000 34001400 01004600     g.e. .... 4... ..F.
29872   69006c00 65005600 65007200 73006900     i.l. e.V. e.r. s.i.
29888   6f006e00 00000000 31002e00 30003100     o.n. .... 1... 0.1.
29904   2e003000 30003000 31000000 38001400     ..0. 0.0. 1... 8...
29920   01005000 72006f00 64007500 63007400     ..P. r.o. d.u. c.t.
29936   56006500 72007300 69006f00 6e000000     V.e. r.s. i.o. n...
29952   31002e00 30003100 2e003000 30003000     1... 0.1. ..0. 0.0.
29968   31000000 34001200 01004900 6e007400     1... 4... ..I. n.t.
29984   65007200 6e006100 6c004e00 61006d00     e.r. n.a. l.N. a.m.
30000   65000000 43006100 6d005300 68006500     e... C.a. m.S. h.e.
30016   6c006c00 00000000 44001a00 01004f00     l.l. .... D... ..O.
30032   72006900 67006900 6e006100 6c004600     r.i. g.i. n.a. l.F.
30048   69006c00 65006e00 61006d00 65000000     i.l. e.n. a.m. e...
30064   43006100 6d005300 68006500 6c006c00     C.a. m.S. h.e. l.l.
30080   2e006400 6c006c00 00000000 2c000200     ..d. l.l. .... ,...
```

Figure 10 - Hexadecimal display of sector 33 through 105 offset 29264 through 30080

Searching for `Camouflage`

A search on *`Google`* displayed a few versions of *`Camouflage`*, however the version used by Mr. Leszczynski was found in http://camouflage.unfiction.com (Figure 11).

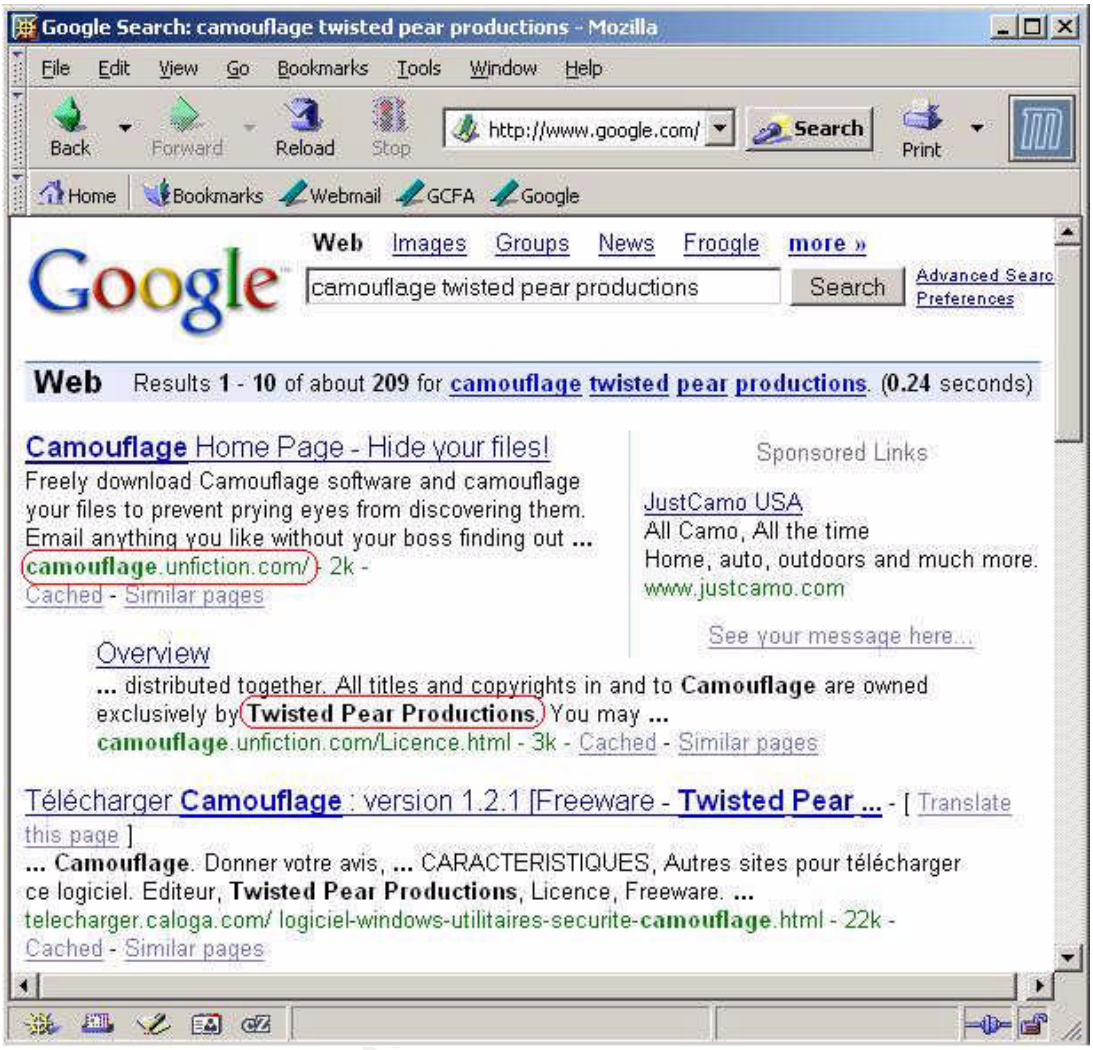


Figure 11 – Screen shot of *Google's* search for *Camouflage*

It is important to note that a search on http://www.camouflage.freeserve.co.uk as well as on http://www.twistedpear.freeserve.co.uk shows these host names no longer in exist. Upon further search with ***Whois* lookups**, name resolution is not possible for either site. However, the *Camouflage* software obtained through http://camouflage.unfiction.com indicates this software is exclusively owned by Twisted Pear Productions. Further analysis of the program's source code indicates the ***CamShell.dll*** of *Camouflage* is the same as the ***CamShell.dll*** recovered from the image. Both ***CamShell.dll*** files were viewed on the universal hexadecimal editor *WinHex 11.8*. A free evaluation version of *WinHex* can be obtained from http://www.x-ways.net/winhex/forensics.html. A synchronized view of the first few offsets of each file is illustrated in Figure 12.

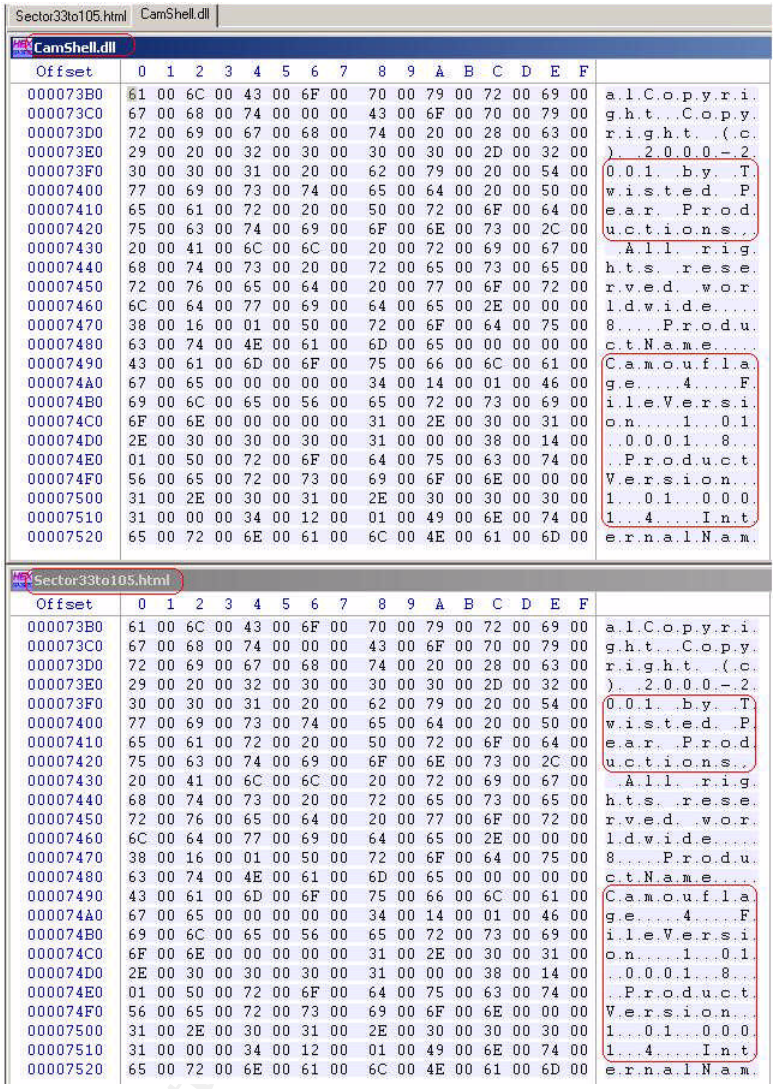


Figure 12 – Screen print of synchronized view of ***CamShell.dll*** on *WinHex 11.8*

In addition, using *WinHex*, a block containing the source code for ***CamShell.dll*** was defined starting on offset 1000 and ending on offset 8730 on both files to capture a hash value of both ***CamShell.dll*** files using **MD5Sum**. By doing so, this integrity check further verifies the correct product version of *camouflage* has been obtained (Figure 13).

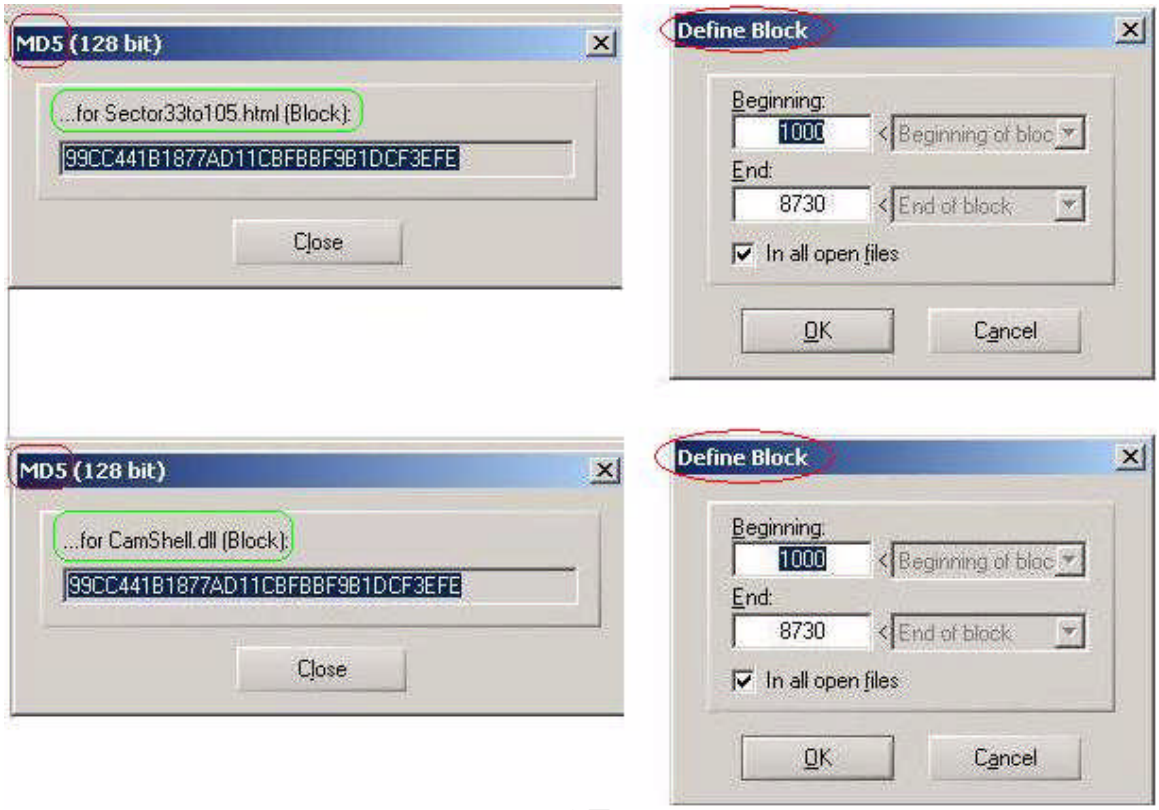


Figure 13 – Screen shot of `MD5` hash value of both ***`CamShell.dll`*** files

Brief Overview of Camouflage

`Camouflage`, a user-friendly steganography software, allows users to hide or "camouflage" virtually any type of file by appending it to another file. The camouflaged file then looks and functions like the second file. For instance, if an adversary wants to conceal a *`Microsoft Access Database`*, he can camouflage the *`Microsoft Access Database`* in a *`Microsoft Word`* document. The *`Microsoft Word`* document would have no visible differences other than it did before the *`Database`* was camouflaged into the document. However, since *`Camouflage`* uses a technique which appends the file, the new file containing the appended camouflaged file increases in size. For example, if the *`Database`* is 184 KB and the *`Word`* document is 30 KB, the new *`Word`* document containing the camouflaged *`Database`* would then be at least 214 KB. A steganography software, like *`Camouflage`*, can be very useful when attempting to conceal a document. For instance, a disgruntled employee can easily conceal a proprietary document on a removable media which can then be disseminated to competitors or adversaries without the knowledge of the rightful owner. Therefore, during a forensics analysis the analyst needs to be cognizant of all aspects of each file, particularly when something as simple as the size of

a document can be a significant clue to foul play.

Suspicious Files

This discussion brings us to the next two files of interest: ***Password_Policy.doc*** and ***Remote_Access_Policy.doc***. Due to the unusually large file sizes, circled in red in Figure 6, it is suspected these files contain camouflaged files. All that is required to reveal potentially camouflaged files is a simple right click on the suspected file followed by the selection of the un-camouflage option. To protect camouflaged files from unauthorized access, *Camouflage* prompts for a password, regardless of whether the file is camouflaged or not. Accordingly, a string search on *Autopsy* was performed in an attempt to gain a clue on possible passwords (Figure 14). However, a password guess can be drawn out and often unsuccessful. Instead, the decision was made to perform a *Google* search to try and find out how *Camouflage* encrypts its passwords. After all, most Steganography freeware is known to use weak encryption systems.

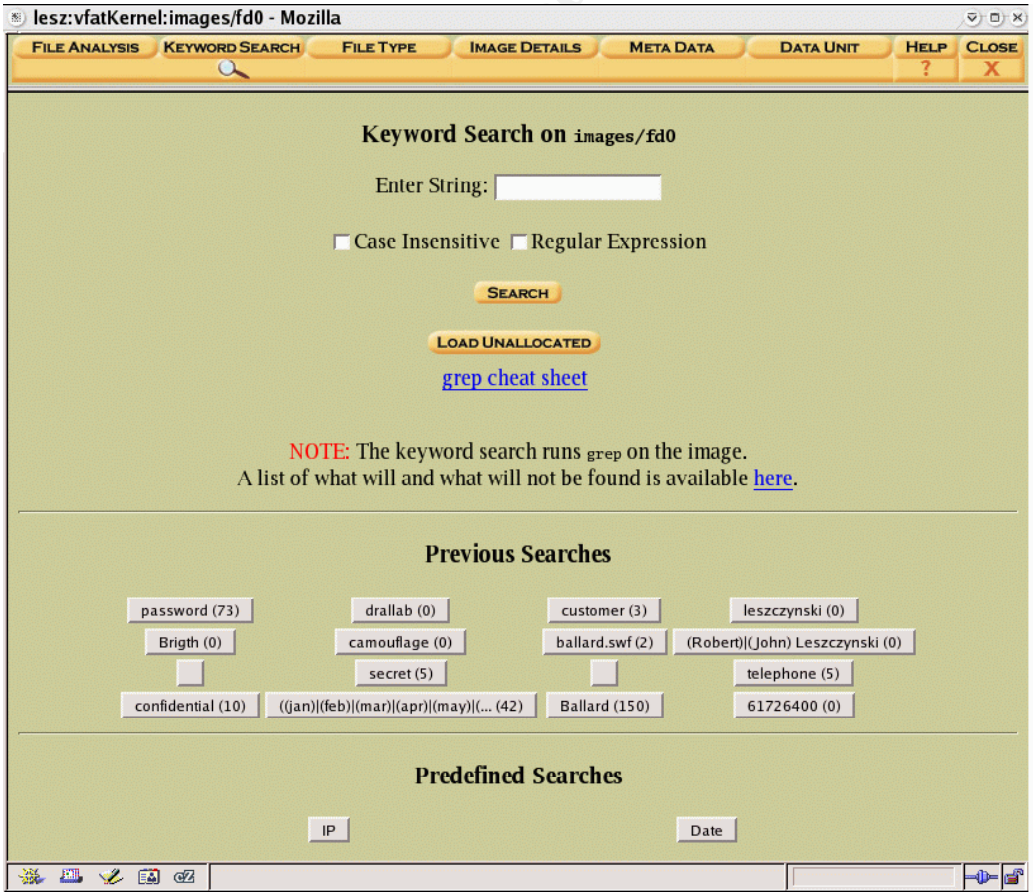


Figure 14 – Screen shot of *Autopsy's* **Keyword Search**

The first *Google* search attempt resulted in over 200 hits. The first listing, a blackhat.com presentation on Steganography seemed promising (Figure 15).

Figure 15 – Screen shot of *Google* search for *Camouflage* password key recovery

The Camouflage Key

The link, http://www.blackhat.com/presentations/bh-usa-04/bh-us-04-raggo/bh-us-04-raggo-up.pdf, corresponds to a presentation entitled *"Steganography, Steganalysis, & Cryptanalysis"* by Michael T. Raggo, Principal Security Consultant at VeriSign. The document contains a detailed explanation on how to locate and decipher the key in *Camouflage* to retrieve the password (Figure 16). *Camouflage* uses a common encryption algorithm called **XOR**. The encrypted password can be **XOR-ed** with the key to obtain the plain-text password.

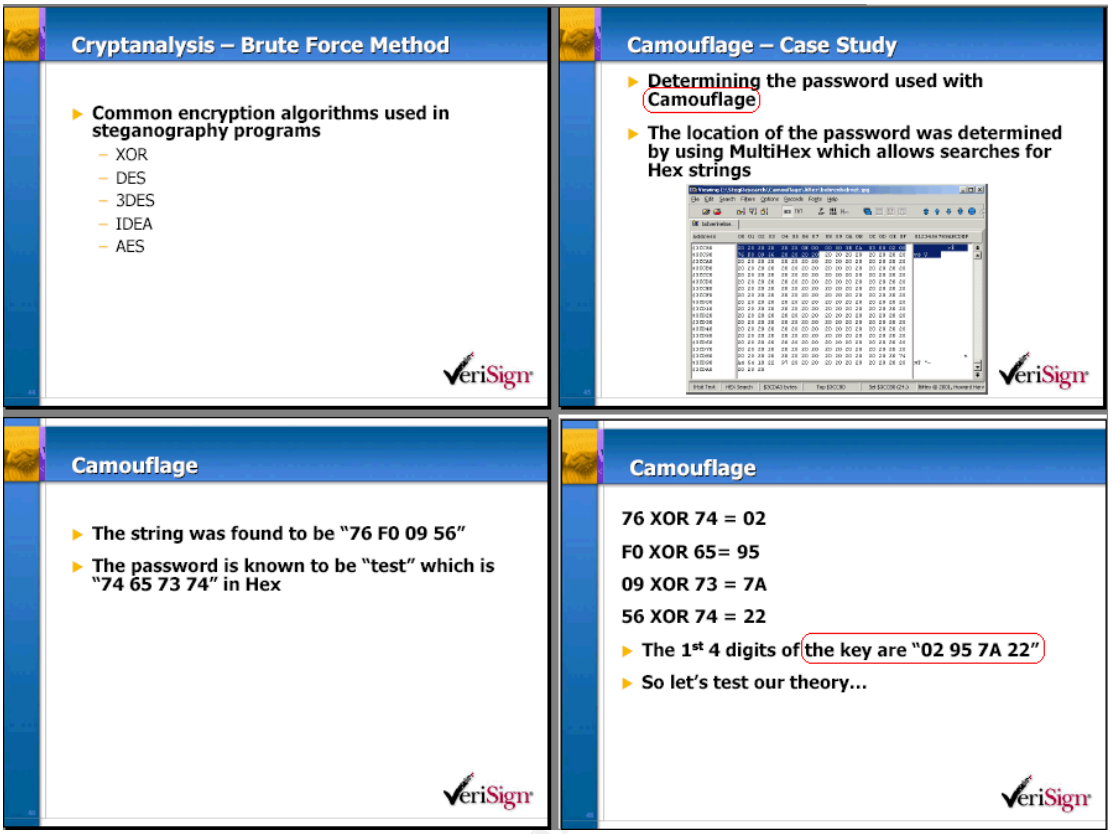


Figure 16 - Un-camouflaging Camouflage's password

Decrypting Camouflage Password

By viewing ***Password_Policy.doc*** and ***Remote_Access_Policy.doc*** in *WinHex*, the strings corresponding to the encrypted password were determined. Using the hexadecimal calculator in *WinHex*, the encrypted password was **XOR-ed** to the key, **02 95 7A 22**. This is the same key documented in the "*Steganography, Steganalysis, & Cryptanalysis"* presentation as shown in Figure 16. A hexadecimal table obtained from http://www.asciitable.com/ was used to translate the resulting HEX string into ASCII (Figure 17). The first four bytes obtained from the ***Remote_Access_Policy.doc*** and ***Password_Policy.doc*** files spelled out **"Remo"** and **"Pass"** respectively. After obtaining the first four characters, minimal efforts were required to successfully guess the passwords. The *Camouflage* passwords needed to access these files are **"Remote"** and **"Password"** the first word of the file names. Ironically, the ***Password_Policy.doc*** document provided by Ballard Industries which was found in the image of the floppy disc stresses the importance of using strong passwords.

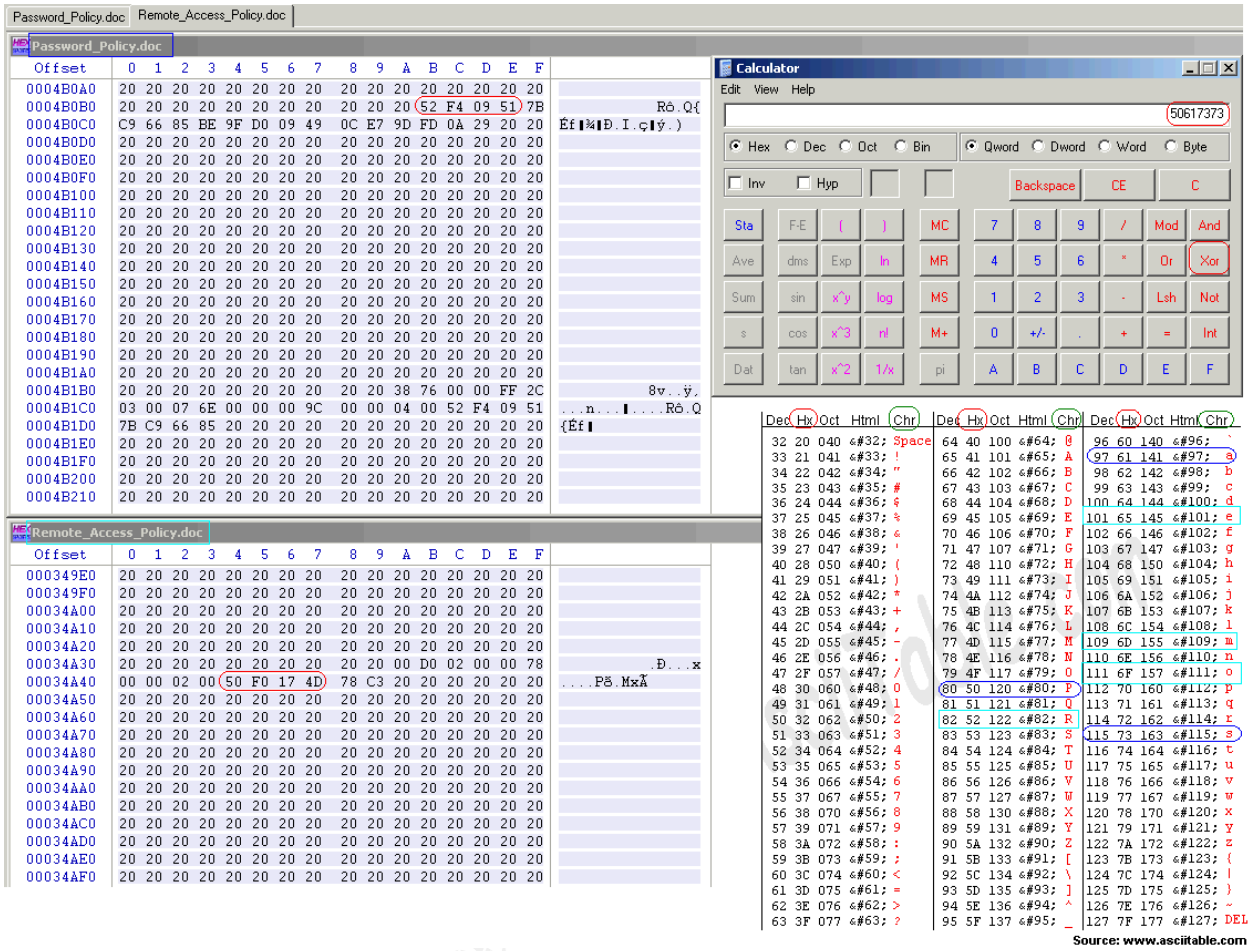


Figure 17 – Screen shot illustrating the process of taking the encrypted password which is then **`XOR-ed`** with key to reveal clear text password

Table 3 summarizes the steps taken to decrypt the encrypted password in *`Camouflage`* for each file in question:

Table 3 - Summary of Password Decryption

File Name	Encrypted Password (in Hexadecimal)	Encryption Algorithm	Key	Decrypted Password (in Hexadecimal)	Clear Text
Password_Policy.doc	52 F4 09 51	XOR	02 95 7A 22	50 61 73 73	P a s s
Remote_Access_Policy.doc	50 F0 17 4D	XOR	02 95 7A 22	52 65 6D 6F	R e m o

"Un-Camouflage" Suspicious Files

Once the password for each file was successfully decrypted, the camouflaged files were retrieved. ***`Password_Policy.doc`*** is 307,935 KB, which is a

comparatively large size for a *Microsoft Word* document. With the use of *Camouflage*, it was revealed that the file contained three hidden images:

- ***PEM-fuel-cell-large.jpg***, size 28 KB
- ***Hydrocarbon fuel cell page2.jpg***, size 203 KB
- ***Pem_fuelcell.gif***, size 30 KB

Note that the original ***Password_Policy.doc*** is only 39 KB (Figure 18), which is a typical size for a *Microsoft Word* document. Note also that the created, modified, and accessed time for all files in Figure 18 is April 23, 2004. This is the last written date shown in *Autopsy*'s File Analysis, Image Details, and File Activity Time Line (Table 1).

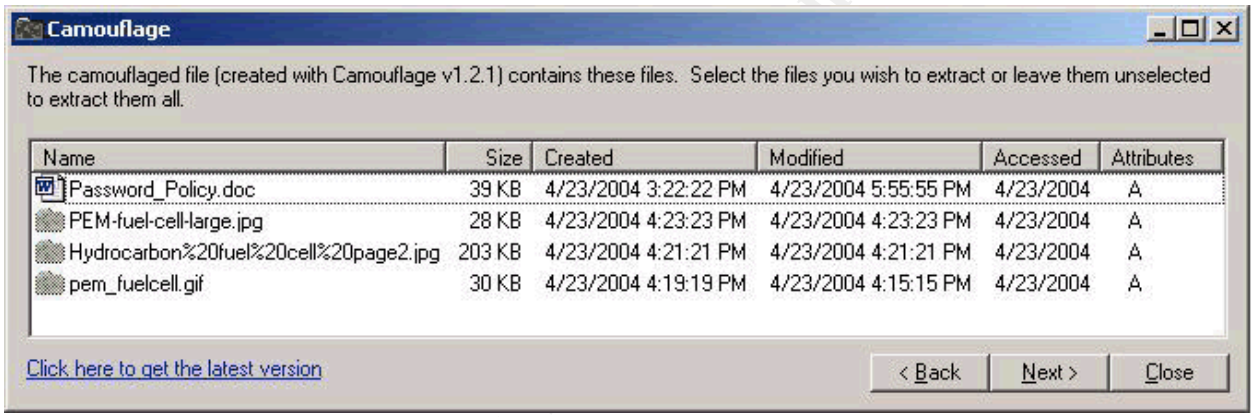


Figure 18 – Screen shot of camouflaged files within ***Password_Policy.doc***

The hidden images contain detailed information on fuel cells being produced by Ballard Industries. The file ***PEM-fuel-cell-large.jpg*** details the design of Ballard Industries proprietary product, the *PEM Fuel Cell* (Figure 19). The next camouflaged file, ***Hydrocarbon fuel cell page2.jpg*** is a scanned document containing detailed information on anode and electrolyte material sets and design implications which could enhance the commercial viability of hydrocarbon-based fuel cells (Figure 20). Further, the file ***Pem_fuelcell.gif*** contains a detailed schematic of the electric circuit of the proprietary hydrocarbon-based fuel cell (Figure 21). These hidden images possess valuable proprietary information which could bring about the loss of technological know-how if appropriated by competitors of the firm. Sensitive information such as trade secrets play a fundamental role to the success of a company. Provided that the hidden files contain what appear to be Ballard Industries trade secrets, it is suspected the files are part of an inherent computer crime on the part of Mr. Leszczynski.

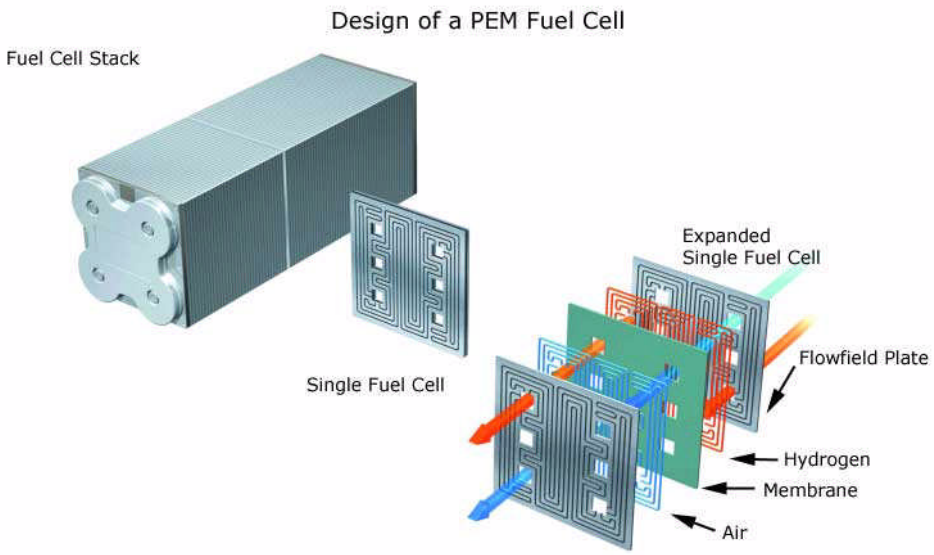


Figure 19 – Screen shot of camouflaged file: ***PEM-fuel-cell-large.jpg***

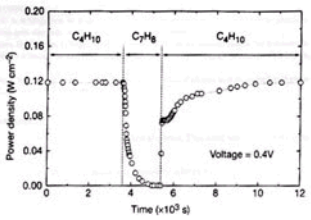


Figure 3 Effect of switching fuel type on the cell with the Cu-ceria composite anode at 973 K. The power density of the cell is shown as a function of time. The fuel was switched from *n*-butane (C_4H_{10}) to toluene (C_7H_8), and back to *n*-butane.

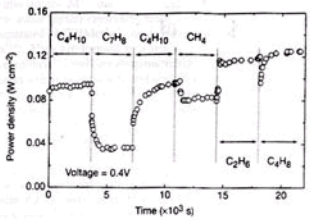


Figure 4 Effect of switching fuel type on the cell with the Cu-(doped ceria) composite anode at 973 K. The power density is shown as a function of time. The fuels were: *n*-butane (C_4H_{10}), toluene (C_7H_8), *n*-butane, methane (CH_4), ethane (C_2H_6), and 1-butene (C_4H_8).

higher temperature. Visual inspection of a cell after two days in *n*-butane at 1,073 K showed that the anode itself remained free of the tar deposits that covered the alumina walls.

Although it is possible that the power generated from *n*-butane fuels resulted from oxidation of H_2—formed by gas-phase reactions of *n*-butane that produce hydrocarbons with a lower C:H ratio—other evidence shows that this is not the case. First, experiments were conducted in which the cell was charged with *n*-butane and then operated in a batch mode without flow. After 30 minutes of batch operation with the cell short-circuited, GC analysis showed that all of the *n*-butane in the cell had been converted completely to CO_2 and water. (Negligible amounts of CO_2 were formed in a similar experiment with an open circuit.) Second, analysis of the CO_2 formed under steady-state flow conditions, shown in Fig. 2, demonstrates that the rate of CO_2 formation increased linearly with the current density. (It was not possible for us to quantify the amount of water formed in our system.) Figure 2 includes data for both *n*-butane at 973 K, and methane at 973 K and 1,073 K. The lines in the figure were calculated assuming complete oxidation of methane (the dashed line) and *n*-butane (the solid line) to CO_2 and water according to reactions (1) and (2):

$$CH_4 + 4O^{2-} \rightarrow CO_2 + 2H_2O + 8e^- \quad (1)$$

$$C_4H_{10} + 13O^{2-} \rightarrow 4CO_2 + 5H_2O + 26e^- \quad (2)$$

With methane, only trace levels of CO were observed along with CO_2, so that the agreement between the data points and the calculation demonstrates consistency in the measurements and no leaks in the cell. With *n*-butane, simultaneous, gas-phase, free-radical reactions to give hydrocarbons with various C:H ratios make quantification more difficult; however, the data still suggest that complete oxidation is the primary reaction. Furthermore, the batch experiments show that the secondary products formed by gas-phase reactions are ultimately oxidized as well. Taken together, these results demonstrate the direct, electrocatalytic oxidation of a higher hydrocarbon in a SOFC.

Along with our observation of stable power generation with *n*-butane for 48 hours, Fig. 3 further demonstrates the stability of the composite anodes against coke formation. Aromatic molecules, such as toluene, are expected to be precursors to the formation of graphitic coke deposits. In Fig. 3, the power density was measured at 973 K and 0.4 V while the fuel was switched from dry *n*-butane, to 0.033 bar of toluene in He for 30 minutes, and back to dry *n*-butane. The data show that the performance decreased rapidly in the presence of toluene. Upon switching back to dry *n*-butane, however, the current density returned to 0.12 W cm^{-2} after one hour. Because the return was not instantaneous, it appears that carbon formation occurred during exposure to toluene, but that the anode is self-cleaning. We note that the electrochemical oxidation of soot has been reported by others[11].

The data in Fig. 4 show that further improvements in cell performance can be achieved. For these experiments, samaria-doped ceria was substituted for ceria in the anode, and the current densities were measured at a potential of 0.4 V at 973 K. The power densities for H_2 and *n*-butane in this particular cell were approximately 20% lower than for the first cell, which is within the range of our ability to reproduce cells. However, the power densities achieved for some other fuels were significantly higher. In particular, stable power generation was now observed for toluene. Similarly, Fig. 4 shows that methane, ethane and 1-butene could be used as fuels to produce electrical energy. The data show transients for some of the fuels, which are at least partially due to switching.

The role of samaria in enhancing the results for toluene and some of the other hydrocarbons is uncertain. While samaria is used to enhance mixed (ionic and electronic) conductivity in ceria and could increase the active, three-phase boundary in the anode, samaria is also an active catalyst[12]. Other improvements in the performance of SOFCs are possible. For example, the composite anodes could be easily attached to the cathode-supported, thin-film electrolytes that have been used by others to achieve very high power densities[3]. In addition to raising the power density, thinner electrolytes may also allow lower operating temperatures.

Additional research is clearly necessary for commercial development of fuel cells which generate electrical power directly from hydrocarbons; however, the work described here suggests that SOFCs have an intriguing future as portable, electric generators and possibly even as energy sources for transportation. The simplicity afforded by not having to reform the hydrocarbon fuels is a significant advantage of these cells. □

Received 13 September 1999; accepted 26 January 2000.

1. Steele, B. C. H. Running on natural gas. *Nature* **400**, 620–621 (1999).
2. Service, R. F. Bringing fuel cells down to earth. *Science* **285**, 682–685 (1999).
3. Perry Murray, E., Tsai, T. & Barnett, S. A. A direct-methane fuel cell with a ceria-based anode. *Nature* **400**, 649–651 (1999).
4. Putna, E. S., Stubenrauch, J., Vohs, J. M. & Gorte, R. J. Ceria-based anodes for the direct oxidation of methane in solid oxide fuel cells. *Langmuir* **11**, 4832–4837 (1995).
5. Park, S., Craciun, R., Vohs, J. M. & Gorte, R. J. Direct oxidation of hydrocarbons in a solid oxide fuel cell: I. methane oxidation. *J. Electrochem. Soc.* **146**, 3603–3605 (1999).
6. Steele, B. C. H., Kelly, I., Middleton, P. H. & Rudkin, R. Oxidation of methane in solid-state electrochemical reactors. *Solid State Ionics.* **28**, 1547–1552 (1988).
7. Lloyd, A. C. The power plant in your basement. *Sci. Am.* **281**(1), 80–86 (1999).

Figure 20 – Screen shot of camouflaged file: ***Hydrocarbon fuel cell page2.jpg***

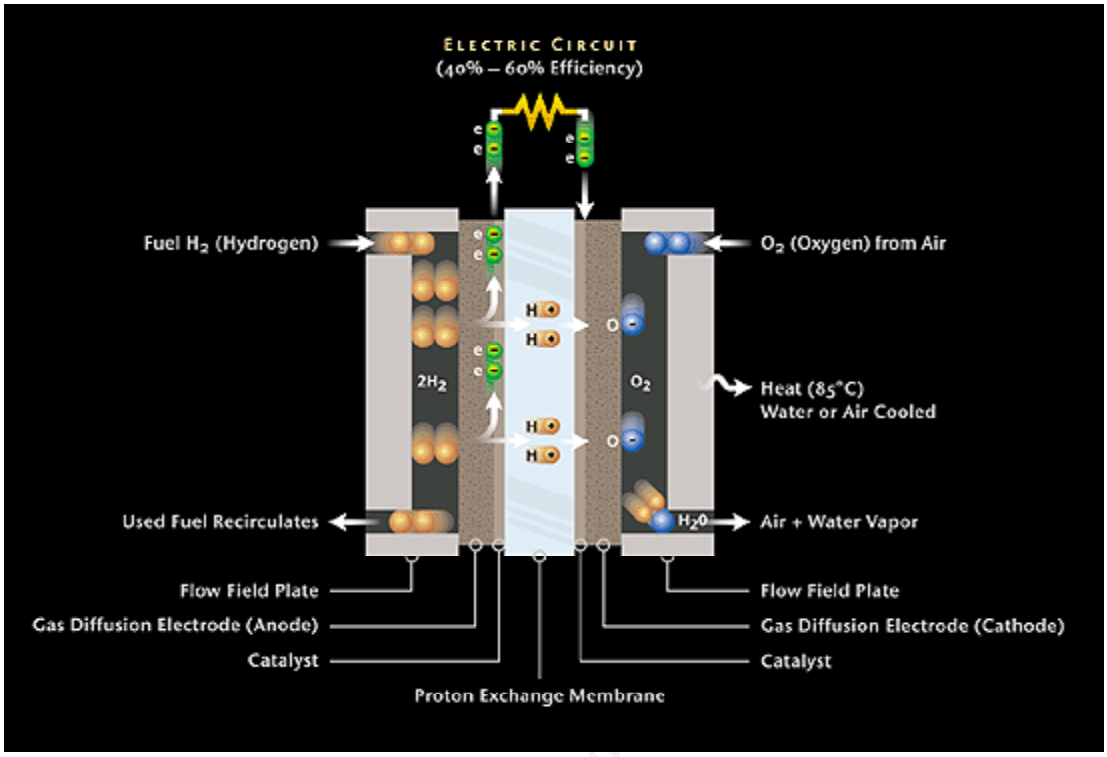


Figure 21 – Screen shot of camouflaged file: ***PEM_fuelcell.gif***

The next file in question, ***Remote_Access_Policy.doc***, which is 32,256 KB, contains one camouflaged file:

- ***CAT.mdb*** which is 180 KB

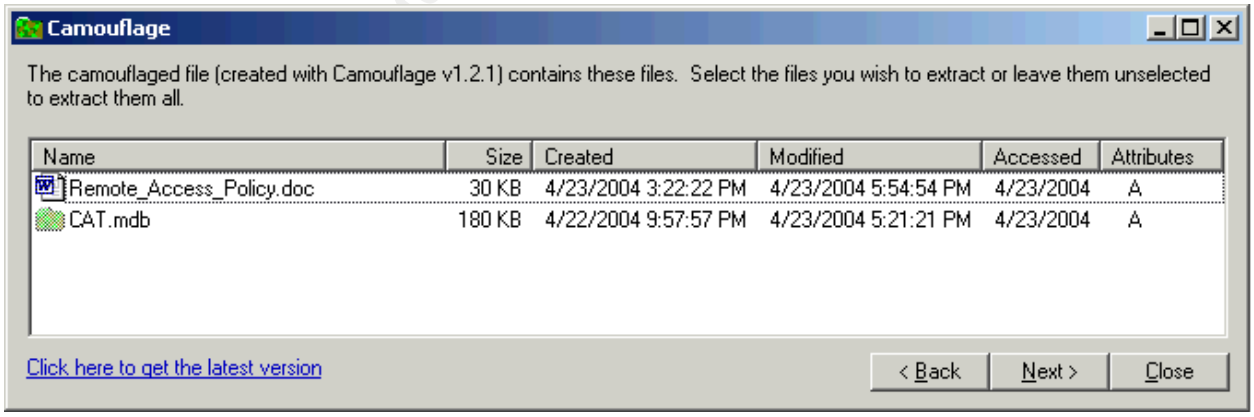
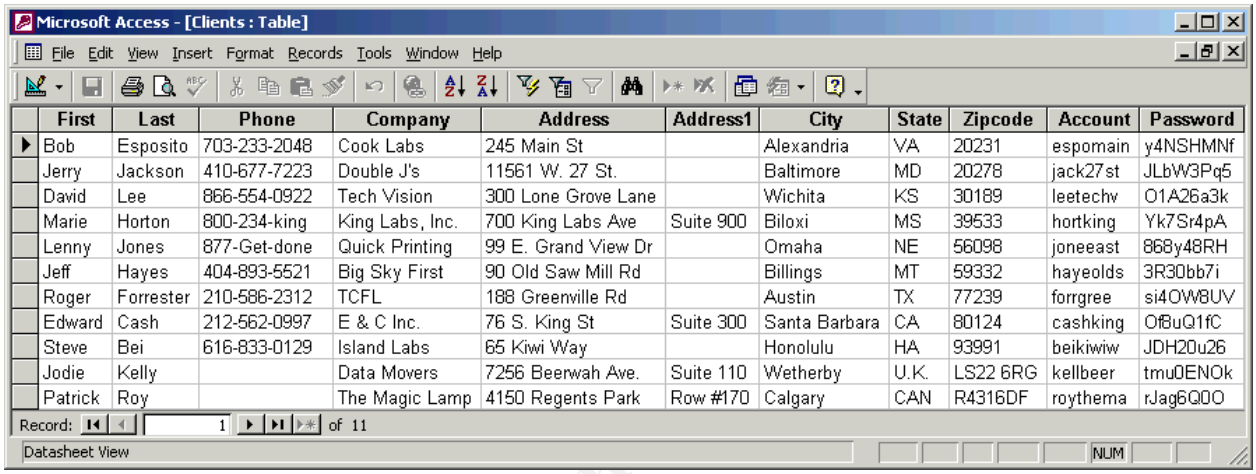


Figure 22 – Screen shot of `camouflaged` files within ***Remote_Access_Policy.doc***

Once again, as illustrated in Figure 22, the original ***Remote_Access_Policy.doc*** is only 30 KB in size, which is significantly smaller than the resulting camouflaged file. The camouflaged file, ***CAT.mdb*** is a `Microsoft Access` database which appears to be a customer database

containing detailed information on Ballard's clients including the clients' full name, company name, contact information, account name and even the account password in plain text as illustrated in Figure 23. The original ***Remote_Access_Policy.doc*** was created, modified, and accessed on the same date as the last written date previously shown in Table 1. The camouflaged ***CAT.mdb*** file was created on April 22, 2004, modified and accessed on April 23, 2004.



First	Last	Phone	Company	Address	Address1	City	State	Zipcode	Account	Password
Bob	Esposito	703-233-2048	Cook Labs	245 Main St		Alexandria	VA	20231	espomain	y4NSHMNf
Jerry	Jackson	410-677-7223	Double J's	11561 W. 27 St.		Baltimore	MD	20278	jack27st	JLbW3Pq5
David	Lee	866-554-0922	Tech Vision	300 Lone Grove Lane		Wichita	KS	30189	leetechv	O1A26a3k
Marie	Horton	800-234-king	King Labs, Inc.	700 King Labs Ave	Suite 900	Biloxi	MS	39533	hortking	Yk7Sr4pA
Lenny	Jones	877-Get-done	Quick Printing	99 E. Grand View Dr		Omaha	NE	56098	joneeast	868y48RH
Jeff	Hayes	404-893-5521	Big Sky First	90 Old Saw Mill Rd		Billings	MT	59332	hayeolds	3R30bb7i
Roger	Forrester	210-586-2312	TCFL	188 Greenville Rd		Austin	TX	77239	forrgree	si4OW8UV
Edward	Cash	212-562-0997	E & C Inc.	76 S. King St	Suite 300	Santa Barbara	CA	80124	cashking	OfBuQ1fC
Steve	Bei	616-833-0129	Island Labs	65 Kiwi Way		Honolulu	HA	93991	beikiwiw	JDH20u26
Jodie	Kelly		Data Movers	7256 Beerwah Ave.	Suite 110	Wetherby	U.K.	LS22 6RG	kellbeer	tmu0ENOk
Patrick	Roy		The Magic Lamp	4150 Regents Park	Row #170	Calgary	CAN	R4316DF	roythema	rJag6Q0O

Figure 23 – Screen shot of camouflaged customer *Microsoft Database*

The data contained in the database file exhibits sensitive third party confidential information which if not properly protected could lead to a potential monetary loss by Ballard Industries. If competitors of the firm, such as Rift, Inc., were to acquire such sensitive information, the competitor company could try to attain the customers from Ballard Industries in efforts to gain an edge on market share. Further, if it is revealed that Ballard Industries did not properly protect such sensitive information, non-disclosure agreements could be violated. Either case could lead to a prospective monetary loss by Ballard Industries.

The Final Analysis

Based on the digital evidence analyzed during the investigation, it is concluded Mr. Leszczynski took advantage of his position as lead process control engineer at Ballard Industries to acquire and illegally disseminate sensitive proprietary information. Mr. Leszczynski used *Camouflage* to conceal proprietary information, which to the casual observer appeared to be company policy documents. However, with the use of *Camouflage*, the proprietary information was compressed and concealed in the same file as the company policy documents, all of which fit within a 3.5 inch TDK floppy disk. Further, because one of Ballard's major competitors, Rift, Inc., has been receiving orders for fuel cell batteries which were once unique to Ballard, it is my opinion Mr.

Leszczynski has been providing this sensitive proprietary information to Rift. Although the digital forensics evidence is circumstantial, legal action against Mr. Leszczynski is highly justifiable, and in collaboration with conventional investigation techniques, conviction is likely.

Prior reference was made to the fact that all non-deleted files in this image have *read (r), write (w), and execute (x) owner permissions*. In addition all files have ***root*** as file owner. To guard against similar digital crime, it is highly recommended that Mr. Keen and the System Administrators at Ballard consider modifying access permission requirements for users. It is recommended that access permissions and file ownership be based on the nature of the employee's responsibilities and their position in the company. For example, Mr. Leszczynski, as lead process control engineer, does not need access to the company's confidential client confidential information, as that contained in ***CAT.mdb***. Thus, a permission structure enabling users with the appropriate need to know to access sensitive information could be implemented to help minimize the compromise of intellectual property by dishonest employees.

Additionally, the company ought to consider strict rules regarding the ability of employees to download and install programs on their work-stations. Perhaps administrator rights on employee workstations should only be given to designated System Administrators. Therefore, if any employee needs access to certain software, that software would be made available by the System Administrator and not by the employee at will. Finally, Mr. Keen and the company's System Administrators should consider performing random system checks on all workstations for suspicious programs which have not been authorized by a Supervisor or System Administrator, like `Camouflage`.

Legal Implications

As stated by Warren G. Kruse II and Jay G. Heiser in the book *"Computer Forensics Incident Response Essentials,"* there are two types of computer exploitations: *"A computer is used to commit a crime, or the computer itself is the target of a crime."* Fraud, theft of intellectual property and theft of trade secrets are examples in which a computer is used to commit a crime.

Case Example

The theft of trade secrets as detailed in *Title 18, Crimes and Criminal Procedure* (http://www.cybercrime.gov/1832NEW.htm), is a crime and can result in a maximum statutory penalty of not more than $5,000,000, or imprisonment of not more than 10 years, or both. Under the ruling of *United*

States Of America v. Trieu Lam and Thanh Tran, which was filed November 03, 2004, CR 04 20198, CASBN 118321 http://www.usdoj.gov/usao/can/press/assets/applets/2004_11_04_Lam_ind.pdf both defendants pled guilty to theft of trade secret and criminal forfeitures. Mr. Trieu Lam was charged with one count of conspiracy to possess stolen trade secrets, and two counts of theft of trade secrets. Mr. Tran was charged with one count of conspiracy to possess stolen trade secrets. As stated in the indictment, both defendants are facing maximum statutory penalty of 10 years imprisonment and a fine of $250,000. A press release posted at the U.S. Department of Justice website, any sentence following conviction would be dictated by the Federal Sentencing Guidelines and imposed by the discretion of the Court. This is just one example in which it is evident that in the United States, the theft of trade secrets is a serious crime and can lead to serious retributions.

In this case, based on the digital evidence found on the floppy seized from Mr. Leszczynski, it can be alleged that he knowingly and without authorization used a company-owned computer to unlawfully obtain proprietary and confidential information from Ballard Industries through the use of `Camouflage`. However, to further support this case and prove beyond reasonable doubt, it is recommended Mr. Leszczynski's computer be seized and analyzed. In addition, supporting evidence such as router or firewall logs should be analyzed to confirm that no other user had access to Mr. Leszczynski's computer through the company network. To demonstrate that Mr. Leszczynski passed the proprietary and confidential information to Ballard's major competitor, Rift, Inc., conventional investigative techniques would need to be pursued. For instance, it may be possible to obtain a warrant stipulating the search of all computer systems and paperwork at Rift, Inc., based upon an alleged conspirator relationship between Mr. Leszczynski and Rift, Inc. The implication is that Rift Inc knew the information received from Mr. Leszczynski was illegally obtained proprietary information.

Nevertheless, Mr. Leszczynski was caught in the act of attempting to remove a floppy from the R&D labs at Ballard industries which is against company policy. Moreover, Mr. Leszczynski violated Ballard Industries "Information Sensitivity Policy" which clearly addresses the responsibilities of each employee in regards to protecting information of varying sensitivity levels. As stated in the "Information Sensitivity Policy", it is the responsibility of every employee at Ballard Industries to familiarize themselves with the guidelines regarding proper handling and protection of sensitive company information such as trade secrets. Efforts made by Mr. Leszczynski to secretly remove sensitive information from Ballard Industries without proper authorization clearly violate the company's "Information Sensitivity Policy". Based on the digital evidence analyzed during the investigation, it has been proven the floppy contains sensitive proprietary and confidential information which belongs to Ballard Industries. The "Information Sensitivity Policy" clearly states that any employee found in violation of the

policy may be subject to disciplinary action, including termination of employment. As a result of the findings obtained during the investigation, it is recommended Mr. Leszczynski be dismissed from the company.

Additional Information

More Information on Steganography tools and methods can be found at:

- *"Current Steganography Tools and Methods"* by Erin Michaud http://www.giac.org/practical/GSEC/Erin_Michaud_GSEC.pdf
- Presentation by Michael T. Raggo, CISSP *"Steganography, Steganalysis, & Cryptanalysis"* http://www.blackhat.com/presentations/bh-usa-04/bh-us-04-raggo/bh-us-04-raggo-up.pdf

More information on `Camouflage` Software can be found at:

- *`Camouflage`* Home Page http://camouflage.unfiction.com/
- *"The Ease of Steganography and Camouflage"* by John Bartlett http://www.sans.org/rr/whitepapers/vpns/762.php

Information on Title 18 can be found at:

- U.S. Department of Justice Press Release http://www.usdoj.gov/usao/can/press/html/2004_11_04_lam.html
- U.S. Department of Justice http://www.cybercrime.gov/1832NEW.htm

Part 2 – Forensic Tool Validation

The primary purpose of Part 2 is to analyze a tool to determine the value of such tool in performing a forensics investigation. Value entails the usefulness of the tool with respect to maintaining evidence integrity, as well as obtaining repeatable and reproducible results. The tool chosen for this validation is *Hurricane Search 4.07*, formerly known as *WinGREP. Hurricane Search* is a search tool which locates text stored on a computer hard drive including text files, *PDF* documents, compressed zip and binary files.

Scope

Search tools enable users to scour a computer hard drive for a given sequence of characters such as a word or a phrase. The text search tool called *Hurricane Search* performs a variety of tasks that could be of benefit to a digital forensics investigator, such as:

- real time searches on multiple directories that filter out specific subsets and directories
- previews of the search match and several lines around the match
- text search on compressed Zip
- text search on binary files
- extended regular expression support

Using *Hurricane Search*, a string search on numerous document formats was performed to evaluate the programs forensics capabilities. In particular, several areas of interest were emphasized for this analysis: (i) potential corruption of digital evidence when using this tool during a forensic investigation, (ii) verifiable and repeatable results, and (iii) reproducible results.

Tool Description

Hurricane Search was created by Hurricane Software. Hurricane Software provides both software development and professional services. Based in Kansas City, Missouri, Hurricane Software has been in business since 1995. *Hurricane Search* was originally released as *WinGREP* for use by programmers and software developers alike. The highly effective *WinGREP* search tool has evolved to meet the requirements of digital forensic analysts and system security administrators. The latest evolution of Hurricane Software's *WinGREP* search engine is *Hurricane Search 4.07*. Hurricane Software

offers two editions of *Hurricane Search*; the Standard Edition and the Professional Edition. In addition, Hurricane Software provides a fifteen day free trial version of the Professional Edition, *Hurricane Search 4.07 Professional Trial Edition*, which was the edition chosen for this analysis. The features of the *Hurricane Search 4.07 Standard Edition* include:

- multi-file text searches with find and replace capabilities
- export of search results into the following formats: CSV, tab delimited, XML, and standard grep output
- seamless merges with many Integrated Development Environments (IDEs) and Editors
- extended regular and DOS expression syntax
- configurable file mask groups

Hurricane Search 4.07 Standard Edition supports all of the *Microsoft Windows* family platforms. Currently, *Hurricane Search 4.07 Standard Edition* can be purchased at a price of $89 for one user license and $1,600 for up to 25 user licenses from http://www.hurricanesoft.com/cart.jsp (Figure 24).

Hurricane Search 4.0 Standard Edition

Platforms Supported	Windows 95, 98, NT, ME, 2000, XP			
Current Version:	4.07			
PRODUCT	DELIVERY METHOD	MEDIA AVAILABLE	PRICE	
1 User license	Download, Mail	Yes	$89.00	Add To Cart
5 User licenses	Download, Mail	Yes	$400.00	Add To Cart
10 User licenses	Download, Mail	Yes	$700.00	Add To Cart
25 User licenses	Download, Mail	Yes	$1,600.00	Add To Cart

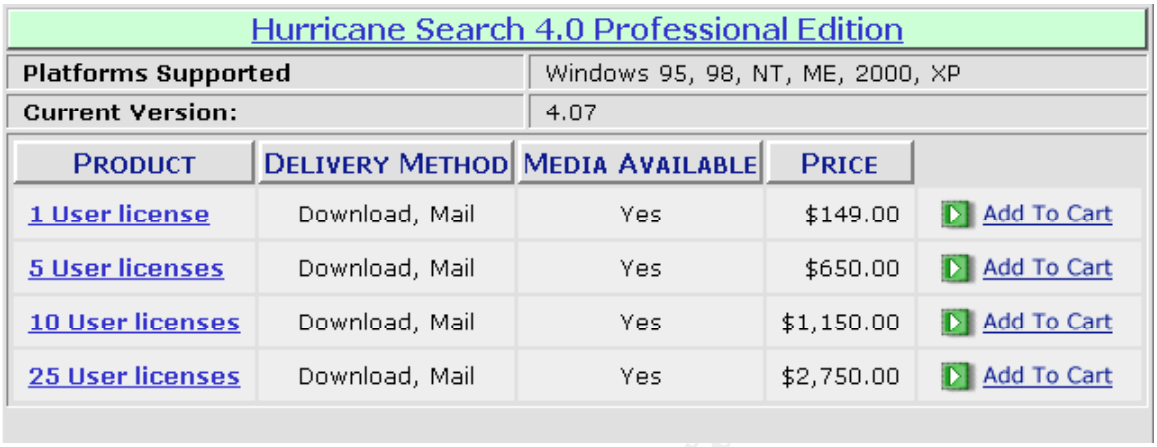
Figure 24 – *Hurricane Search 4.0 Standard Edition*

The features offered in *Hurricane Search 4.07 Professional Edition* include all the features bundled with the Standard Edition as well as the following:

- text searches within *Microsoft Word*, *Adobe PDF* and *Binary* files
- text searches within *Archive* files such as *ZIP* and *Java JAR* files
- with all searches having exclude files and directories capabilities

Hurricane Search 4.0 Professional Edition also supports all of the

Microsoft Windows family platforms. Currently, *Hurricane Search 4.07 Professional Edition* can be purchased at a price of $149 for one user license and $2,750 for up to 25 user licenses from http://www.hurricanesoft.com/cart.jsp (Figure 25).



Hurricane Search 4.0 Professional Edition

Platforms Supported	Windows 95, 98, NT, ME, 2000, XP
Current Version:	4.07

PRODUCT	DELIVERY METHOD	MEDIA AVAILABLE	PRICE	
1 User license	Download, Mail	Yes	$149.00	Add To Cart
5 User licenses	Download, Mail	Yes	$650.00	Add To Cart
10 User licenses	Download, Mail	Yes	$1,150.00	Add To Cart
25 User licenses	Download, Mail	Yes	$2,750.00	Add To Cart

Figure 25 – *Hurricane Search Professional Edition*

In lieu of purchasing the Professional Edition for $149, the free trial version of *Hurricane Search 4.07 Professional Trial Edition* was used to evaluate the effectiveness of this search tool in performing a forensics investigation. The trial version has all of the features of the Professional Edition, except that it expires fifteen days after installation. This fifteen day trial period offers a system administrator or a forensics analyst ample time to evaluate the tool's potential prior to deciding on purchasing the product. The free trial version of *Hurricane Search 4.07 Professional Trial Edition* was downloaded from http://www.hurricanesoft.com/download.jsp (Figure 26).

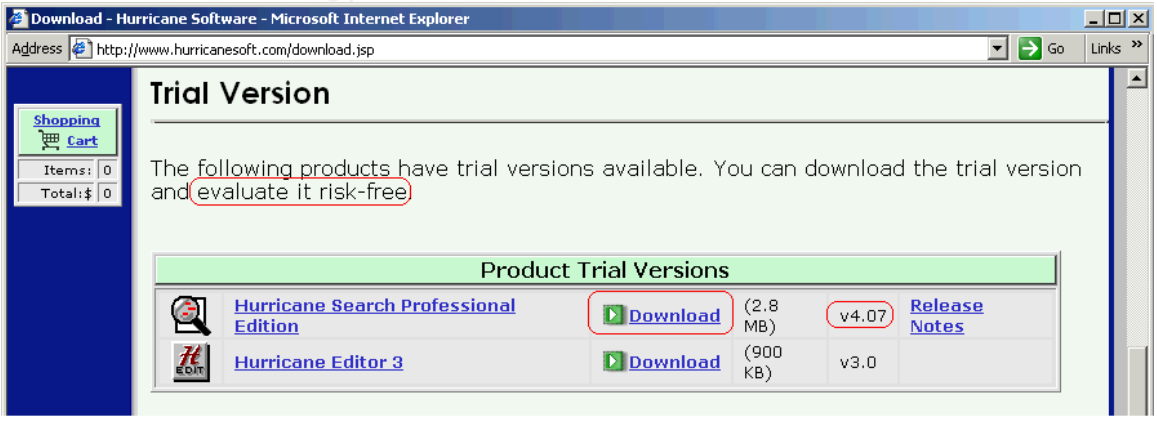


Figure 26 – *Hurricane Search Professional Free Trial Version*

Hurricane Search is derived from the *UNIX* command **`Global Regular`**

Expression Print (GREP). Hurricane Software has transferred the powerful search capabilities of **GREP** into the *Microsoft Windows* platform to provide users a fast and user-friendly stand-alone search tool. During a forensics investigation, a key word search is a fundamental step while performing an analysis of an image. By gathering key information during the analysis, the investigator can ascertain potential digital evidence clues. However depending on the size of the image in question, key word searches can be very time consuming and lengthy. *Hurricane Search* allows users to perform complex searches of regular expressions stored in a computer hard drive. A regular expression is a pattern of words or characters that can match various text strings or set of words or characters. Performing searches of regular expressions allows forensics analysts to expand the keyword search to be used during the investigation. For example, a search on **"passwor[ds]"** matches all lines with either "password" or "passwords." This feature can be especially useful when searching a large image or hard drive since text searches can be time consuming.

Perhaps one of the most significant advantages *Hurricane Search* provides forensics investigators is that the multifaceted search tool is designed for *Microsoft Windows* platforms. *Microsoft Windows* remains the most popular operating system for home offices to the network infrastructure of large corporations, but it is less secure and more vulnerable than other non-mainstream systems, such as *Linux Red Hat*. Hence, having a powerful tool, such as **GREP**, in a *Microsoft Windows* environment, as offered by *Hurricane Search*, can be of great benefit to computer forensics analysts. The **graphical user interface** (GUI) within *Hurricane Search* maximizes productivity by enabling the user to perform several tasks, such as, simultaneous searches of complex regular expressions within multiple directories with a simple click of the mouse. In addition, *Hurricane Search* offers users the capability of exporting and saving search results which can be revisited throughout the forensic investigation.

One drawback of *Hurricane Search* is that the program must be installed on a system for it to run. This can be a huge inconvenience when performing a live-system analysis, because it is crucial to avoid introducing any external data to the system in question in order to preserve evidence. However, if the forensic analyst is using a dedicated investigation machine under a controlled environment, it is good practice to have an assortment of forensics tools that can be judiciously applied during an analysis. Continuous learning and training of forensics tools and methodologies is critical for forensics analyst in order to stay current with the ever evolving cyber world.

System Files and Libraries

When executing *Hurricane Search*, several system libraries and files are accessed when the tool is executed. As such, if this tool were to be used on a live-system during an incident response, the time stamps on the system libraries and files accessed during the execution of the tool would be disturbed. During a forensics analysis, a change of time stamps on any system file could result in potential corruption of digital evidence. Using *OllyDbg*, which is a free 32-bit assembler level analyzing debugger for *Microsoft Windows* platforms (Figure 27), the system libraries and files accessed by *Hurricane Search* were revealed. Certain common system libraries and files include:

- ***kernel32.dll*** is the 32-bit dynamic link library in the *Microsoft Windows* operating system kernel which handles memory management.
- ***gdi32.dll*** are Graphics Device Interface (GDI) functions for device outputs (drawing and font management).
- ***user32.dll*** are *Microsoft Windows* management functions for message handling, timers, menus, and communications.
- ***shell32.dll*** is a library containing *Microsoft Windows* Shell Application Programming Interface (API) functions which are used when opening web pages and files.
- ***Secur32.dll*** is a library which contains *Microsoft Windows* security functions

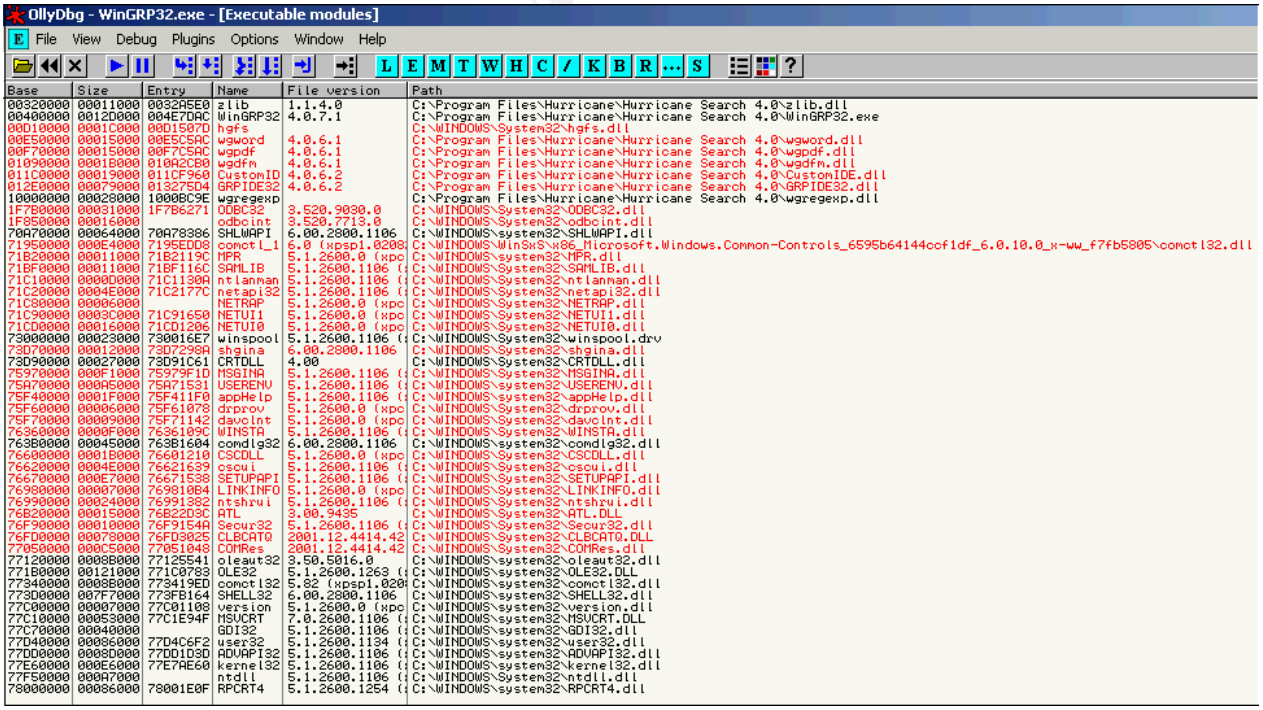


Figure 27 – Screen shot of system libraries and files accessed by *Hurricane Search*

After analyzing the executable file of *Hurricane Search* using *OllyDbg*, it

was determined that the program is not compiled statically. A statically compiled program has all libraries and files needed to run within the program's binary code. Since *Hurricane Search* relies on certain *Microsoft Windows* system files and libraries in order to properly execute, the conclusion was made that the tool is not compiled statically. Although analyzing the program's source code is outside the scope of this paper since *Hurricane Search* is a commercial application and the source code is not readily available, a simple test was performed to verify if the tool can be used in an evidentiary sound way. To do so, first an **md5Sum** hash of the *dll* files and libraries included with the executable and the *Microsoft Windows* files and libraries used by *Hurricane Search* was taken using *Cygwin* and the hash values were appended to a file called ***HSPHashes.txt*** to ensure that the integrity of such files is not corrupted during testing (Figure 28).

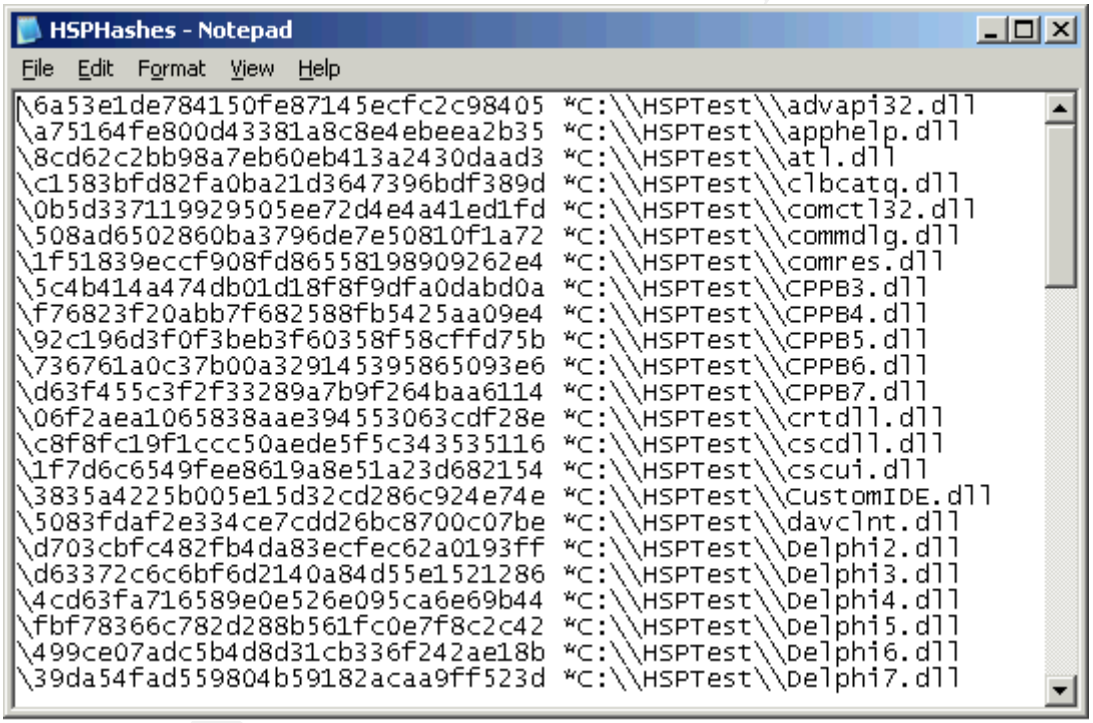


Figure 28 – Screen shot of **HSPHashes.txt** file

Subsequently, an additional *Microsoft Windows XP VMware* image was created called **HurricaneTest** (Figure 29). A copy of the free version of *Hurricane Search 4.07 Professional Trial Edition* and a copy of *OllyDbg* was installed on the **HurricaneTest** workstation. Using *OllyDbg*, the system libraries and files accessed by *Hurricane Search* were revealed (Figure 30).

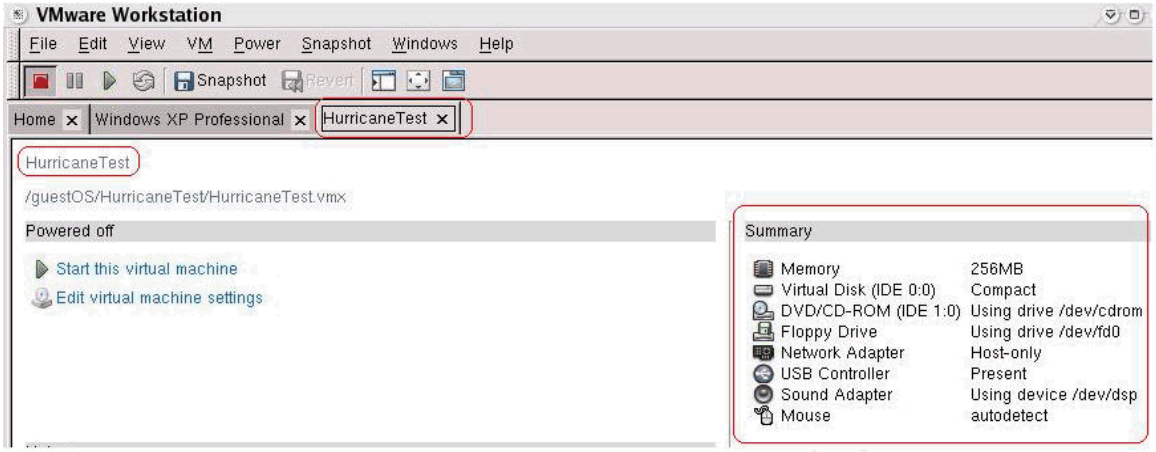


Figure 29 – Screen shot of **`HurricaneTest`** *`Microsoft Windows XP VMware`* Image

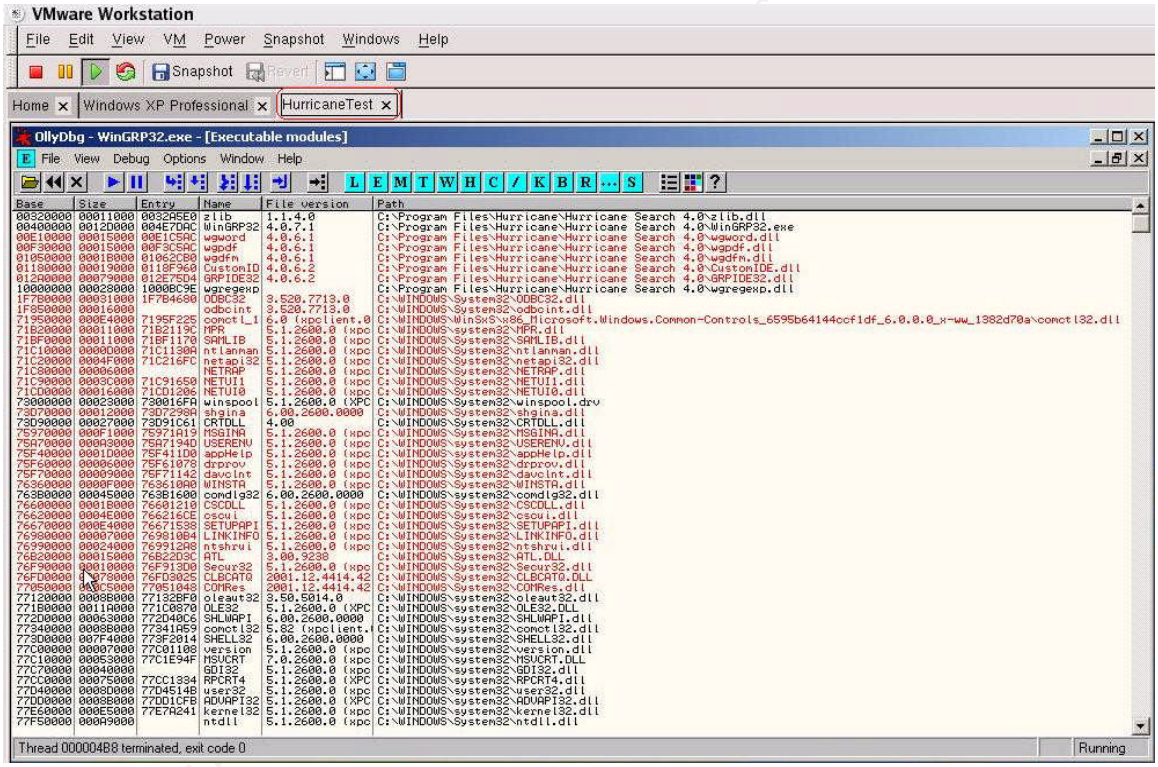


Figure 30 – Screen shot of system libraries and files accessed by *`Hurricane Search`* on the **`HurricaneTest`** VMware Image

The system libraries and files accessed by *`Hurricane Search`* were revealed using *`OllyDbg`*, and compared to those in the *`Microsoft Windows XP VMware`* workstation used for the tool validation. The same system files and libraries are used every time the tool is executed. Further, an **`md5Sum`** hash of the *`dll`* files and libraries included with the executable, the windows files and libraries used by *`Hurricane Search`* were taken using *`Cygwin`*. These hash values were compared to the hash values obtained when the tool was executed in the *`Microsoft Windows XP VMware`* workstation used for the tool

validation. Since hash values act like an electronic fingerprint, it can be concluded the tool uses the same system files and libraries across *Microsoft Windows XP* workstations.

Test Apparatus and Environmental Conditions

To avoid external corruption of the test validation results, the tool validation was performed using a stand-alone *Microsoft Windows XP VMware* workstation consisting of the following:

- Operating System: *Microsoft Windows XP Professional Version 2002*, Service Pack 1
- Hard Drive Capacity (VMWare virtual hard drive): 5 GB
- Processor: Pentium 4 2.40 GHz
- Memory: 256 MB
- Tools: *WinHex 11.8, Google, HashCalc 2.01, Cygwin, Hurricane Search 4.07, OllyDbg 1.09*

VMware allows users such as system administrators, incident responders and forensics analyst the capability of running multiple operating systems in one machine simultaneously. This allows the forensic analyst the ability to readily access more than one operating system concurrently when testing for repeatable, verifiable and reproducible results. As such, a second *Microsoft Windows XP VMware* image was created to verify if the tool can be used in an evidentiary sound way. The second workstation was named **HurricaneTest** and consists of the following:

- Operating System: *Microsoft Windows XP Professional Version 2002*, Service Pack 1
- Hard Drive Capacity (VMWare virtual hard drive): 2 GB
- Processor: Pentium 4 2.40 GHz
- Memory: 256 MB
- Tools: *Cygwin, Hurricane Search 4.07, OllyDbg 1.09*

A free version of *Hurricane Search 4.07 Professional Trial Edition* was downloaded from http://www.hurricanesoft.com/download.jsp and placed in a directory called **Hurricane** in the **C:** drive of both *Microsoft Windows XP VMware* images. The program must be installed in a system in order to execute. As such, *Hurricane Search* was installed by following the directions provided by the installation wizard, which was prompted upon double clicking the stand-alone executable file **hsearch40.exe**. Once installation of the program was completed, the fifteen-day free trial version was registered with the provided registration key.

Description of Procedures

To test the integrity of the files in which the keyword search was performed using *Hurricane Search*, an **md5Sum** hash of the tested files was taken before and after the text search using the tool *HashCalc. HashCalc* is a freeware calculator from SlavaSoft which allows users to compute thirteen of the most popular checksum algorithms and hash values such as **md5Sum, SHA1**, and **RIPEMD160.** A free copy *HashCalc* can be obtained from SlavaSoft website, http://www.slavasoft.com/hashcalc/. To test repeatability and reproducibility, five different directories and five files were created as follows:

1) Under the **C:** drive, five directories were added labeled as: **Test1, Test2, Test3, Test4, Test5**, and **TestResults** (Figure 31).

Figure 31 – Screen shot of directories added on C:\

2) A *Microsoft Word* document containing a description of *Hurricane Search* was created as saved as ***test1.doc*** under the **Test1** directory, (Figure 32). Using *HashCalc*, an **md5Sum** hash value of the file was obtained (Figure 33).

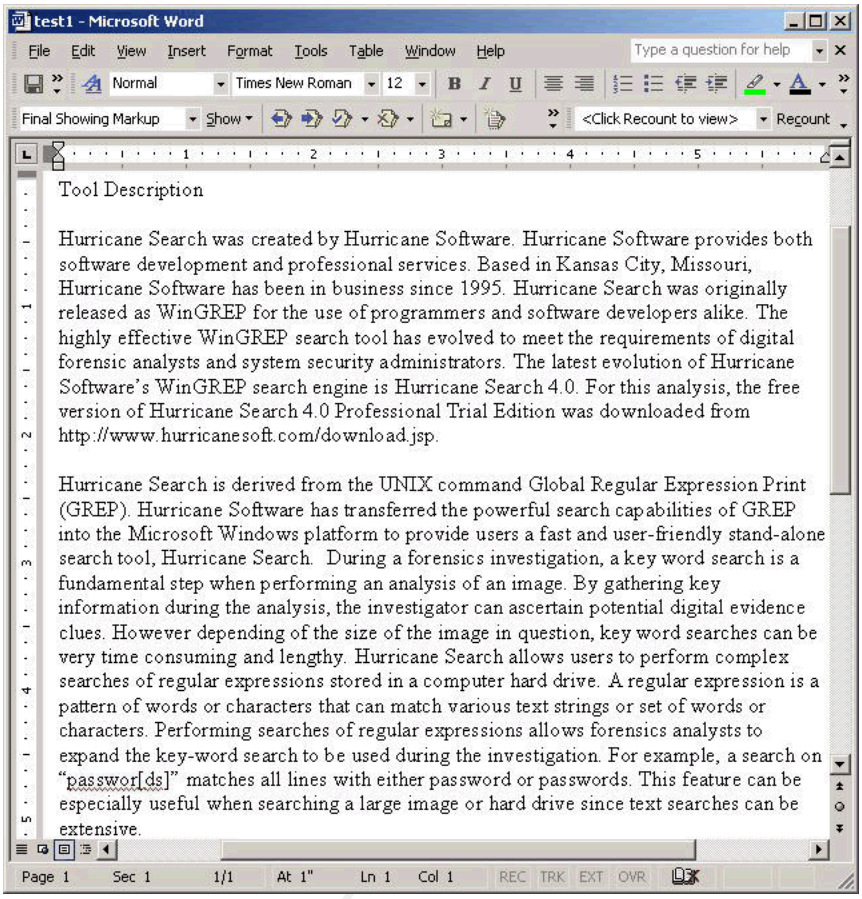


Figure 32 – Screen shot of ***test1.doc*** document containing description of *Hurricane Search*

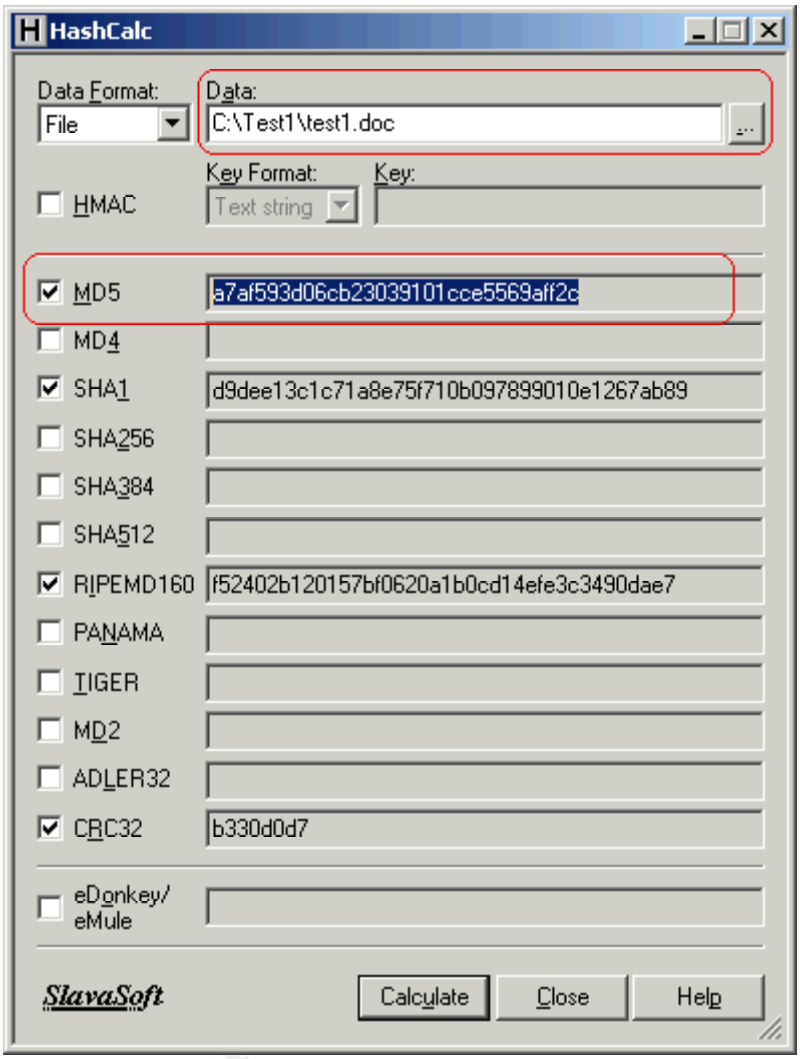


Figure 33 – Screen shot of hash value of ***test1.doc***

3) An *Adobe Acrobat PDF* file containing the same description of *Hurricane Search* was created as saved as ***test2.pdf*** under the **Test2** directory (Figure 34). Using *HashCalc*, an **md5Sum** hash of ***test2.pdf*** was obtained (Figure 35).

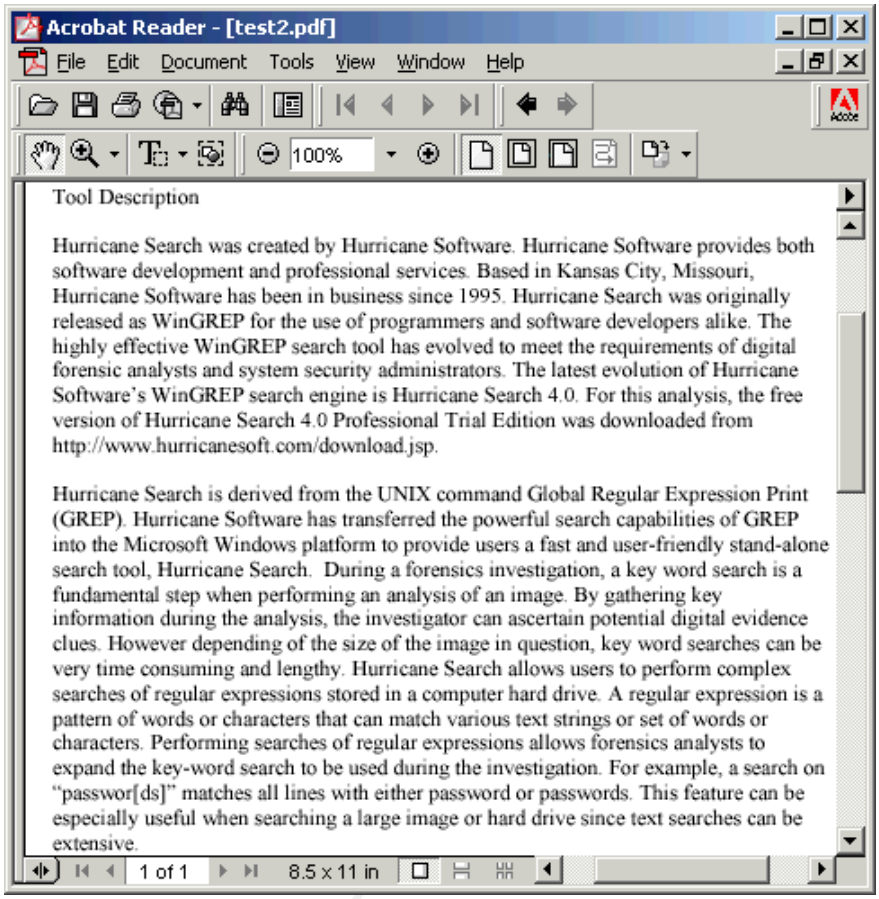


Figure 34 – Screen shot of ***`test2.pdf`*** document containing description of *`Hurricane Search`*

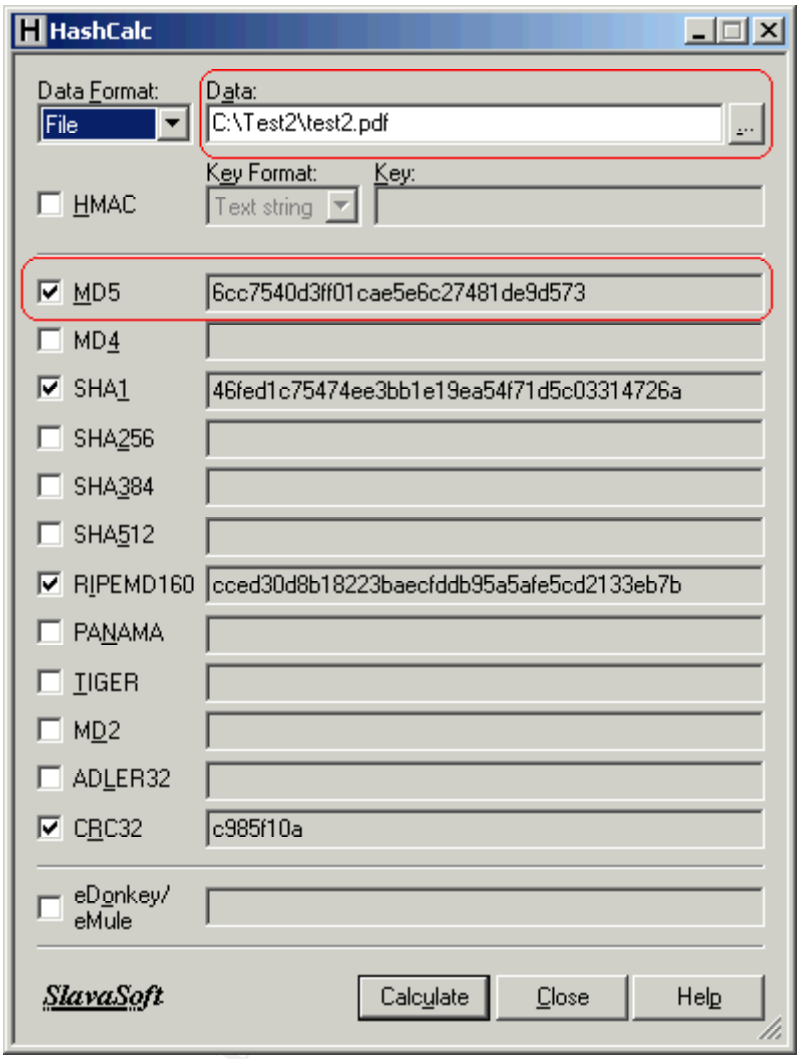


Figure 35 - Screen shot of hash value of ***test2.pdf***

4) Using *Microsoft WordPad*, a file with the same *Hurricane Search* description was created and saved as ***test3.dll*** under the **Test3** directory (Figure 36). *HashCalc* was then used to obtain an **md5Sum** hash of ***test3.exe***, as illustrated in Figure 37.

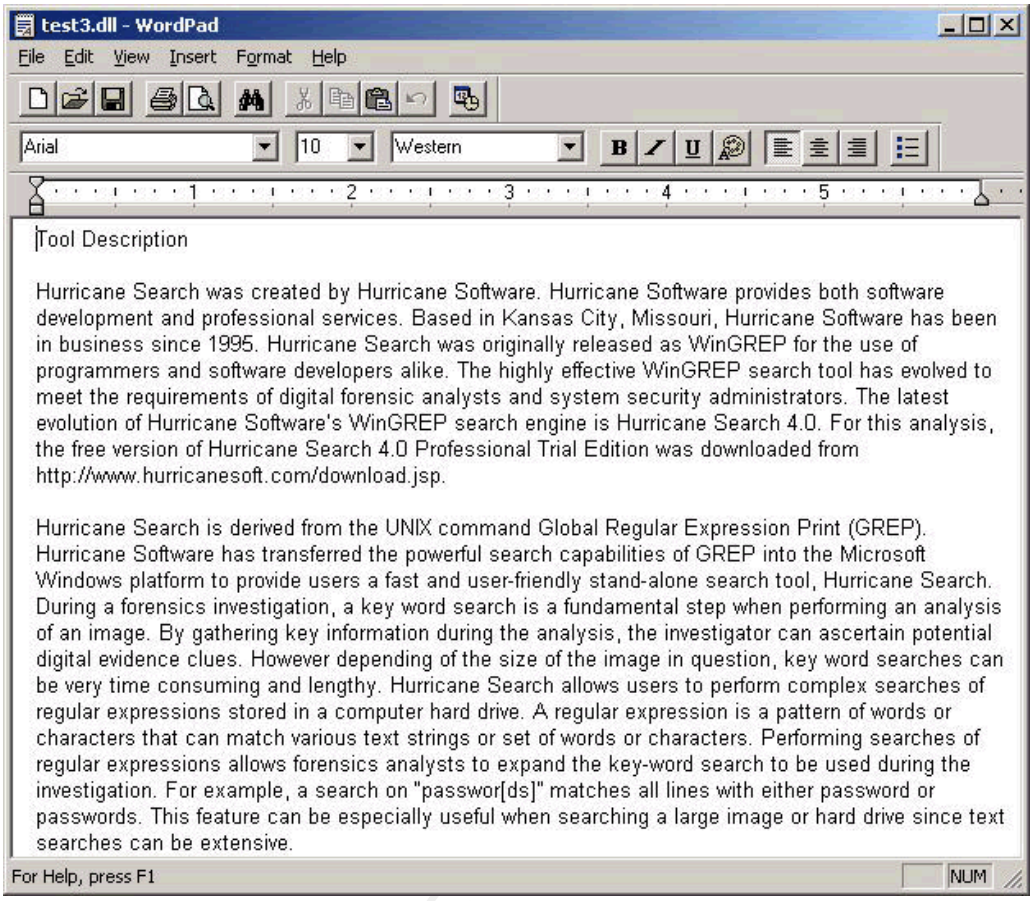


Figure 36 - Screen shot of ***`test3.dll`*** document containing description of *`Hurricane Search`*

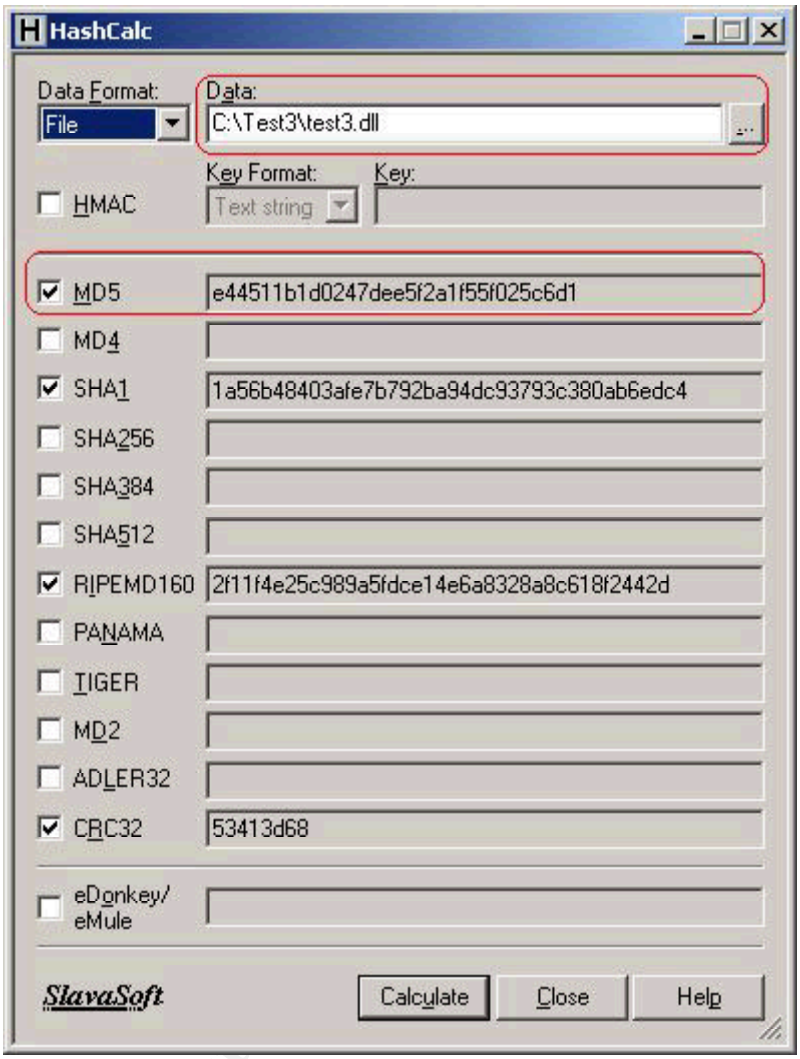


Figure 37 - Screen shot of hash value of ***test3.dll***

5) Using *Microsoft Note Pad*, an *HTML* file with the same *Hurricane Search* description was created and saved as ***test4.htm*** under the **Test4** directory (Figure 38). Once again, using *HashCalc* an **md5Sum** hash of ***test4.htm*** was obtained (Figure 39).

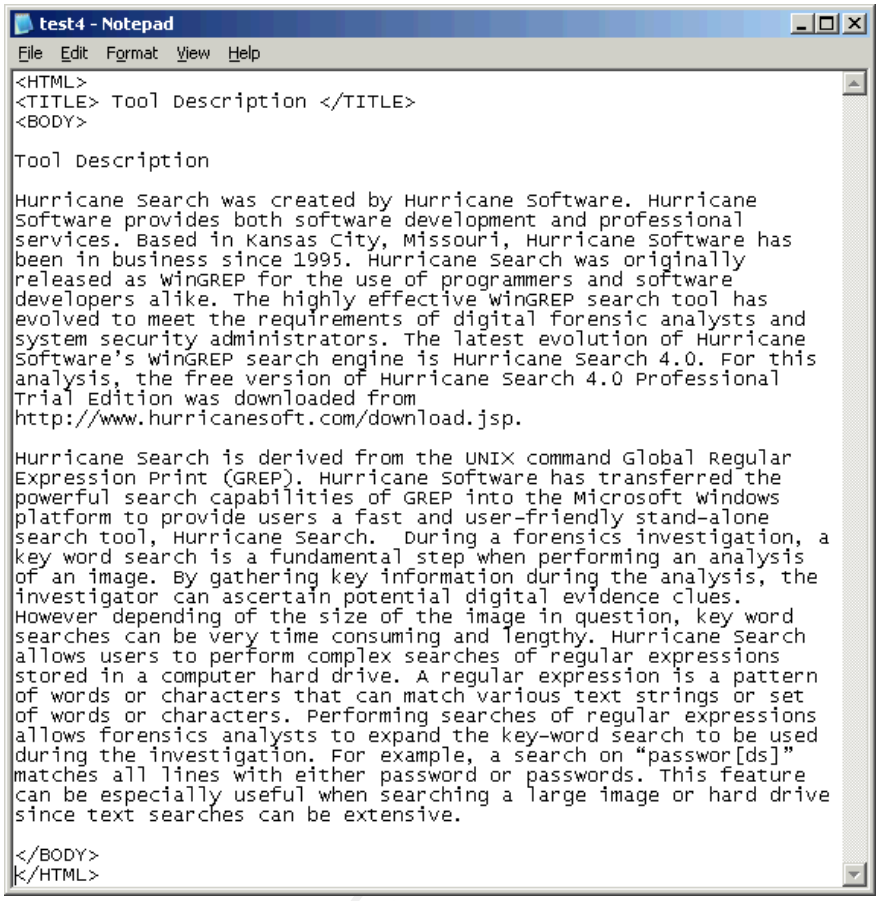


Figure 38 - Screen shot of ***test4.htm*** document containing description of *Hurricane Search*

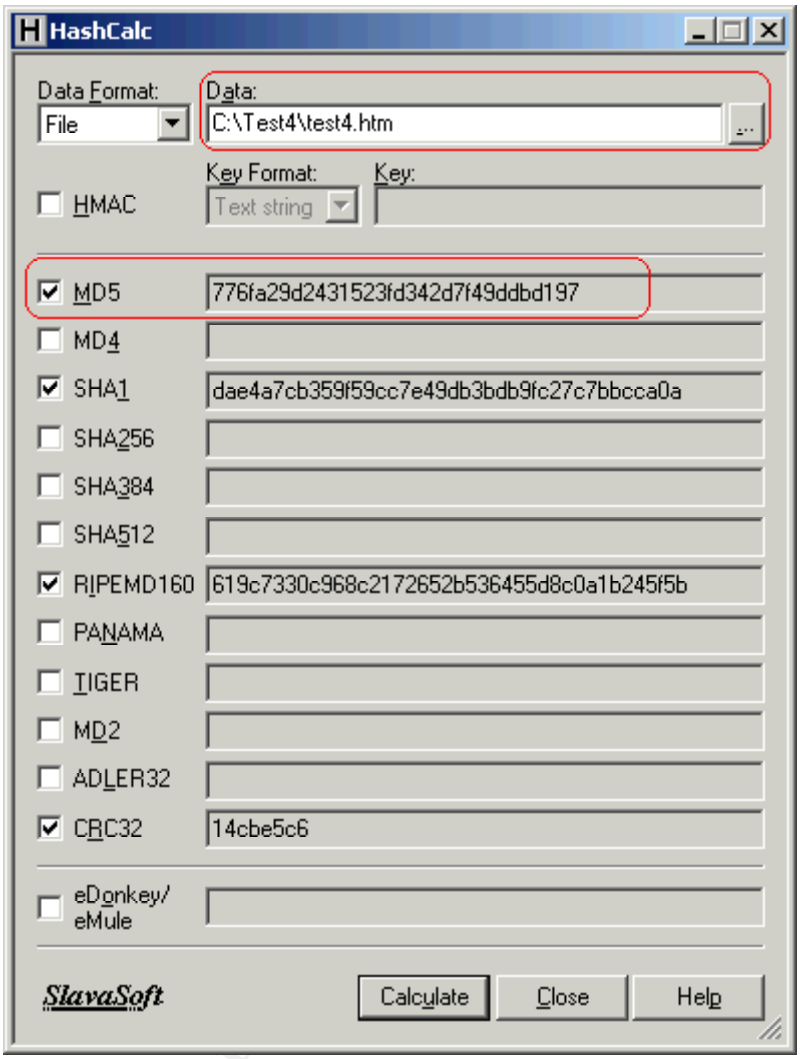


Figure 39 - Screen shot of hash value of ***test4.htm***

6) Finally, using Microsoft *Word* a file containing the same *Hurricane Search* description as all the other files was created and saved as ***test5.doc*** under the **Test5** directory (Figure 40). Further, using the Steganography program *camouflage*, a copy of the *Hurricane Search* executable, **hsearch40.exe** was camouflaged within ***test5.doc***, and an **md5Sum** hash of this file was obtained using *HashCalc* (Figure 41)

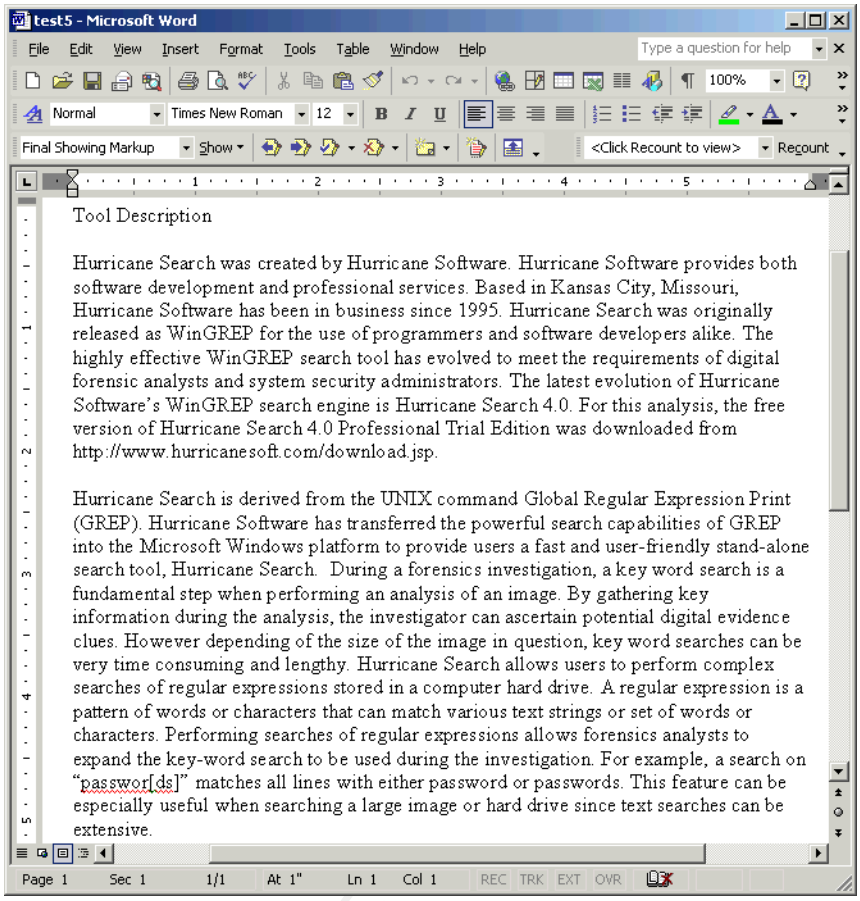


Figure 40 - Screen shot of ***test5.doc*** document containing description of *Hurricane Search* and a "camouflaged" file called **hsearch40.exe**

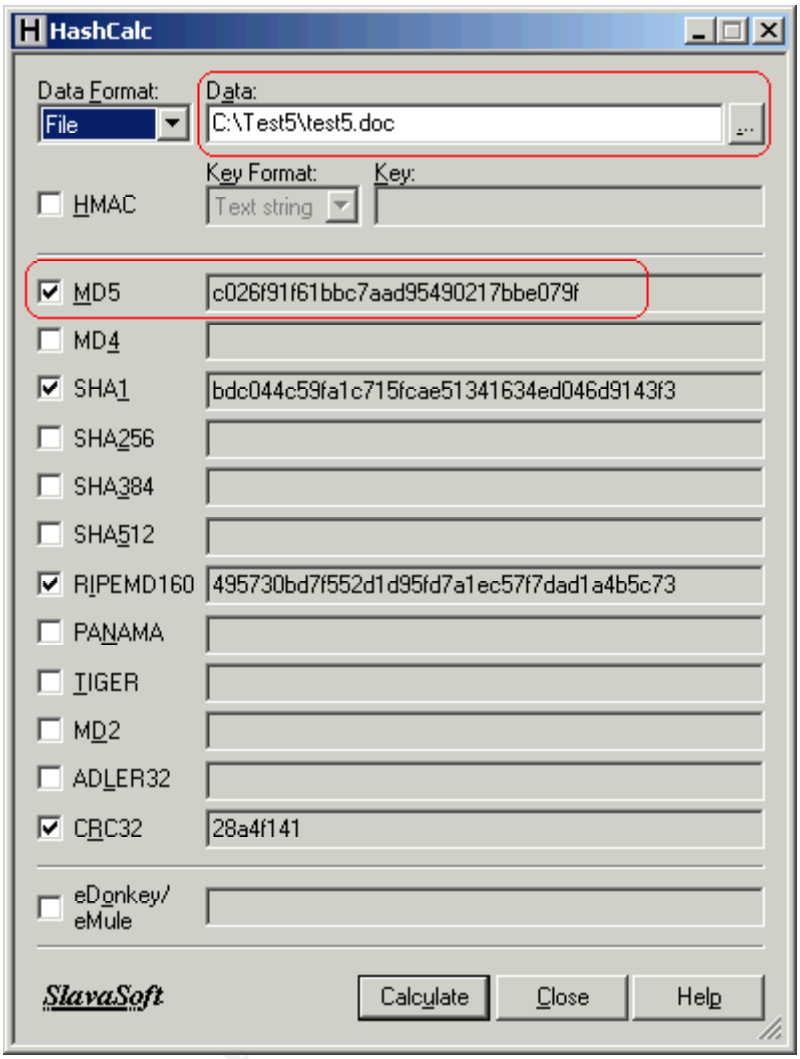


Figure 41 - Screen shot of hash value of ***test5.doc***

Criteria for Approval

Once all the files were created, an **md5Sum** hash of each file was obtained to test for evidence integrity after completing the text search using *Hurricane Search*. At this point, *Hurricane Search* was launched by double clicking on the *Hurricane Search* icon. Because all files contain the same exact information, i.e., the description of *Hurricane Search,* a search on each of the directories should yield the same exact results. For example, if a search on the word "***grep"*** is performed on the **Test1** directory which contains ***test1.doc*** the same search results should be yielded when repeating the same search on all other test files.

The first keyword search was performed on ***test1.doc*** under the **Test1**

directory for the word **"grep"** which yielded the following results as illustrated in Figure 42). These results were then exported and saved as ***test1Results.csv*** under the directory called **TestResults.** Using the option **Clear All Results** under the **Results** menu, the test results were cleared. A search for the word **"grep"** was then performed on ***test2.pdf*** under the **Test2** directory (Figure 43). Again the results were exported and saved as ***test2Results.csv*** under the directory called **TestResults.** This procedure was repeated on files ***test3, test4,*** and ***test5*** accordingly.

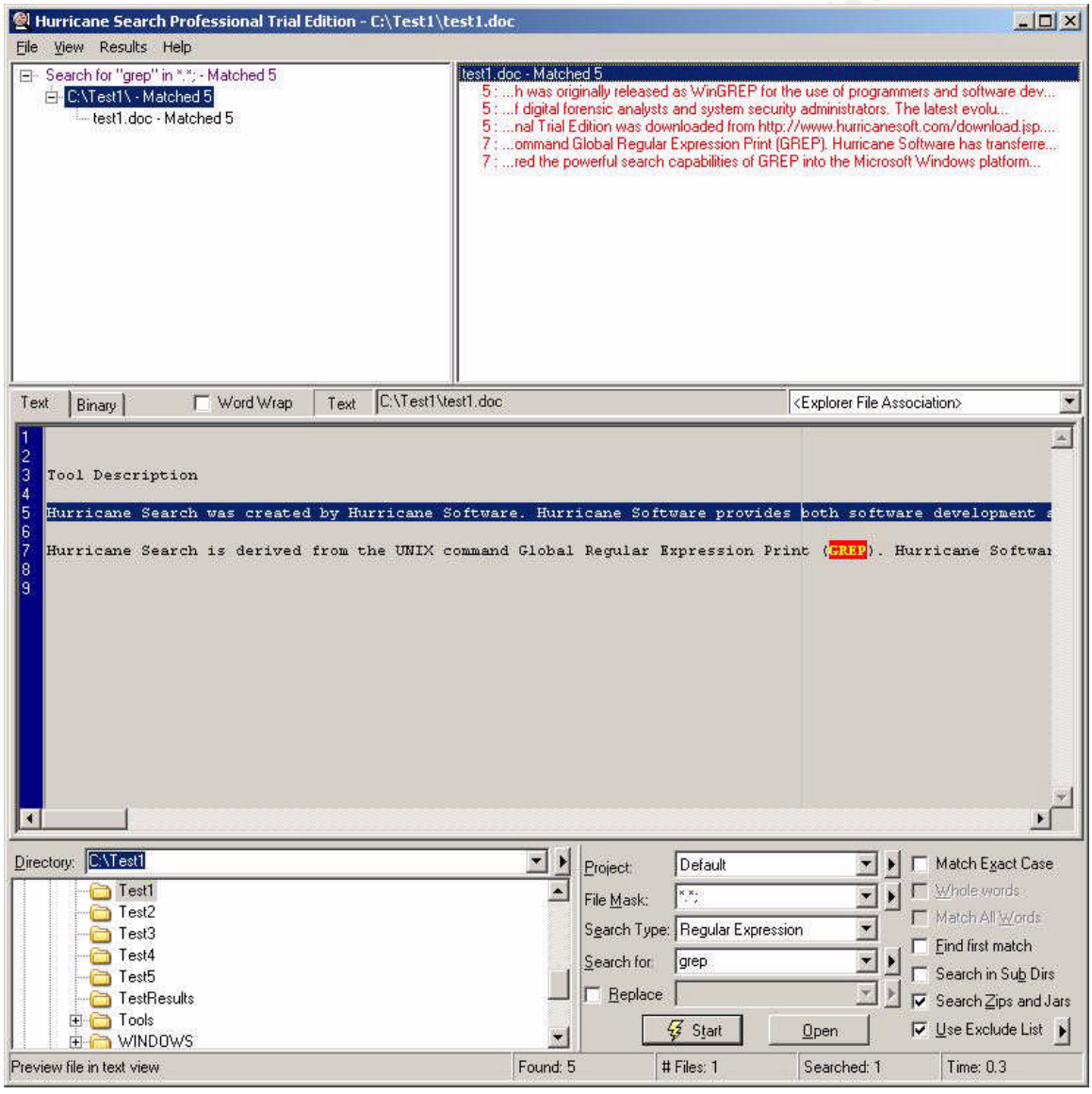


Figure 42 – Screen shot of keyword search performed on ***test1.doc***

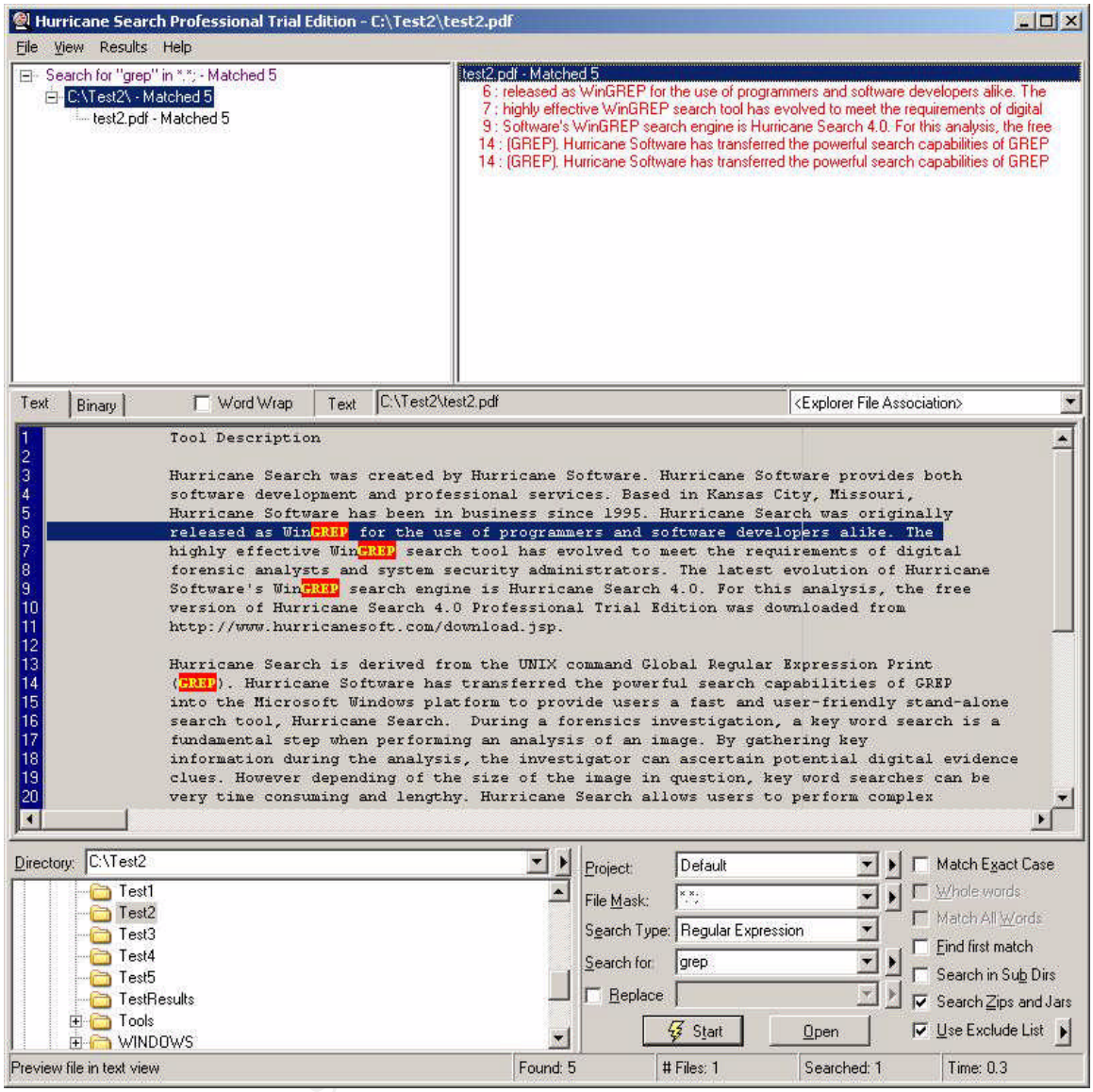


Figure 43 - Screen shot of keyword search performed on ***test2.pdf***

Data and Results

Upon completing the text search on all five test files, an **md5Sum** hash of each file was obtained to test for preservation of evidence. As illustrated in Figure 44, the **md5Sum** hash of each file is exactly the same before and after the text search using *Hurricane Search.* Since the **md5Sum** hash value acts like an electronic fingerprint, it can be concluded that the test files were not corrupted during the text search.

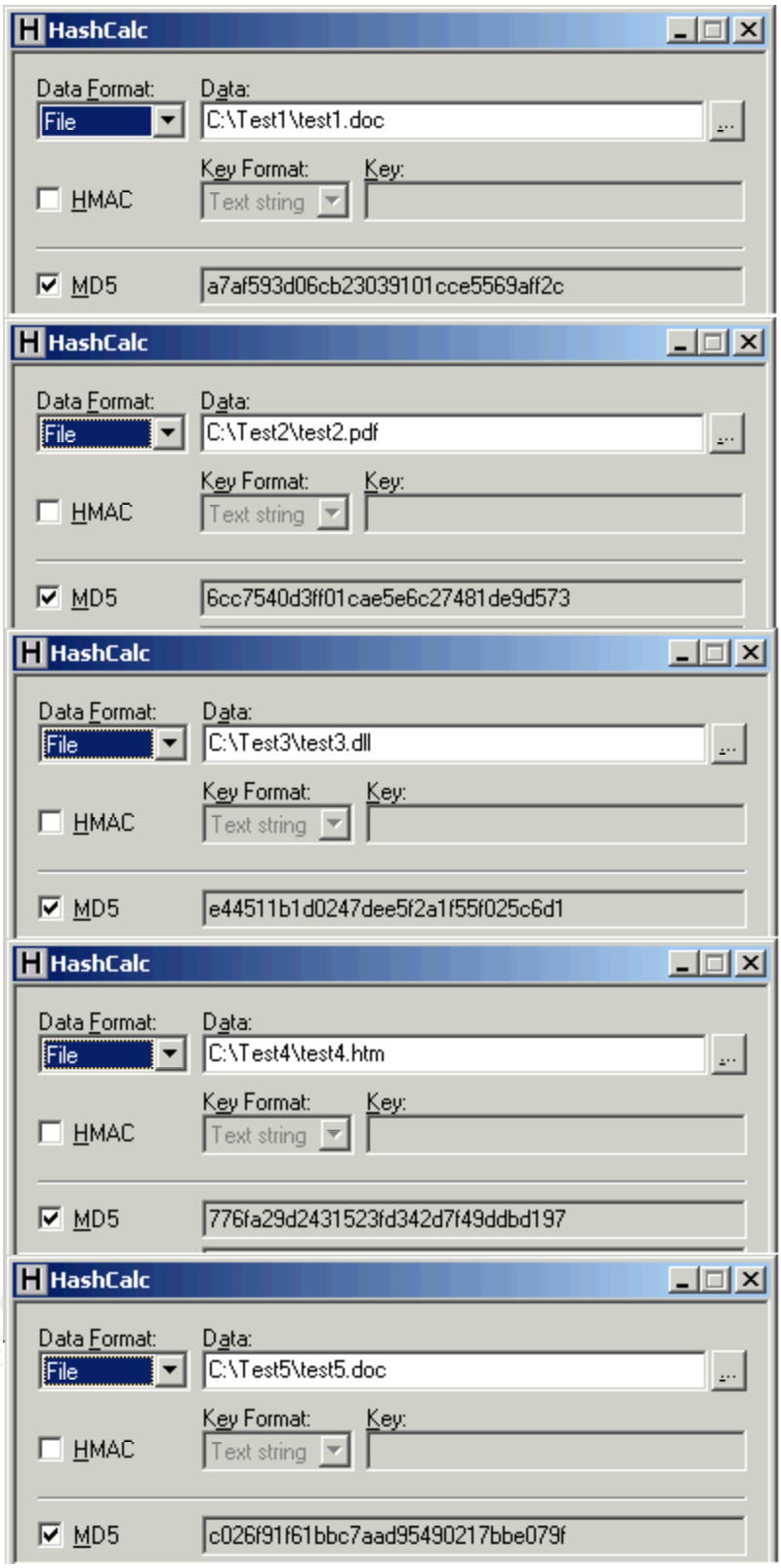


Figure 44 – Screen shot of **`md5Sum`** hash value for each test file after keyword search using *`Hurricane Search`*

Given that Hurricane Software advertises *Hurricane Search* as a search tool that can be used for forensics or active discovery of specific strings within files in a hard drive, this result meets expectations. However, a forensics analyst should keep in mind that this tool must be installed in the system in order to execute. Therefore, *Hurricane Search* is not recommended when performing a live-system analysis because installation of the program would disrupt the original state of the system in question.

Once it was determined that *Hurricane Search* does not jeopardize digital evidence when performing a search, the search results were examined to test verifiability and repeatability. *Hurricane Search* enables users to export the search results and further save the results in a *Microsoft Excel* document. The output of each of the five test files were exported and examined to determine if the results could be repeated and verified (Figure 45). The keyword search on each test file yielded the same results. In each test file, five outputs of the word "**grep**" resulted from the search. As a result, it is concluded that the results obtained when performing a keyword search using *Hurricane Search* are verifiable and repeatable. To test if the results are reproducible, the same test was repeated on a separate workstation. Once again, five instances of the keyword "grep" resulted from the search on each test file.

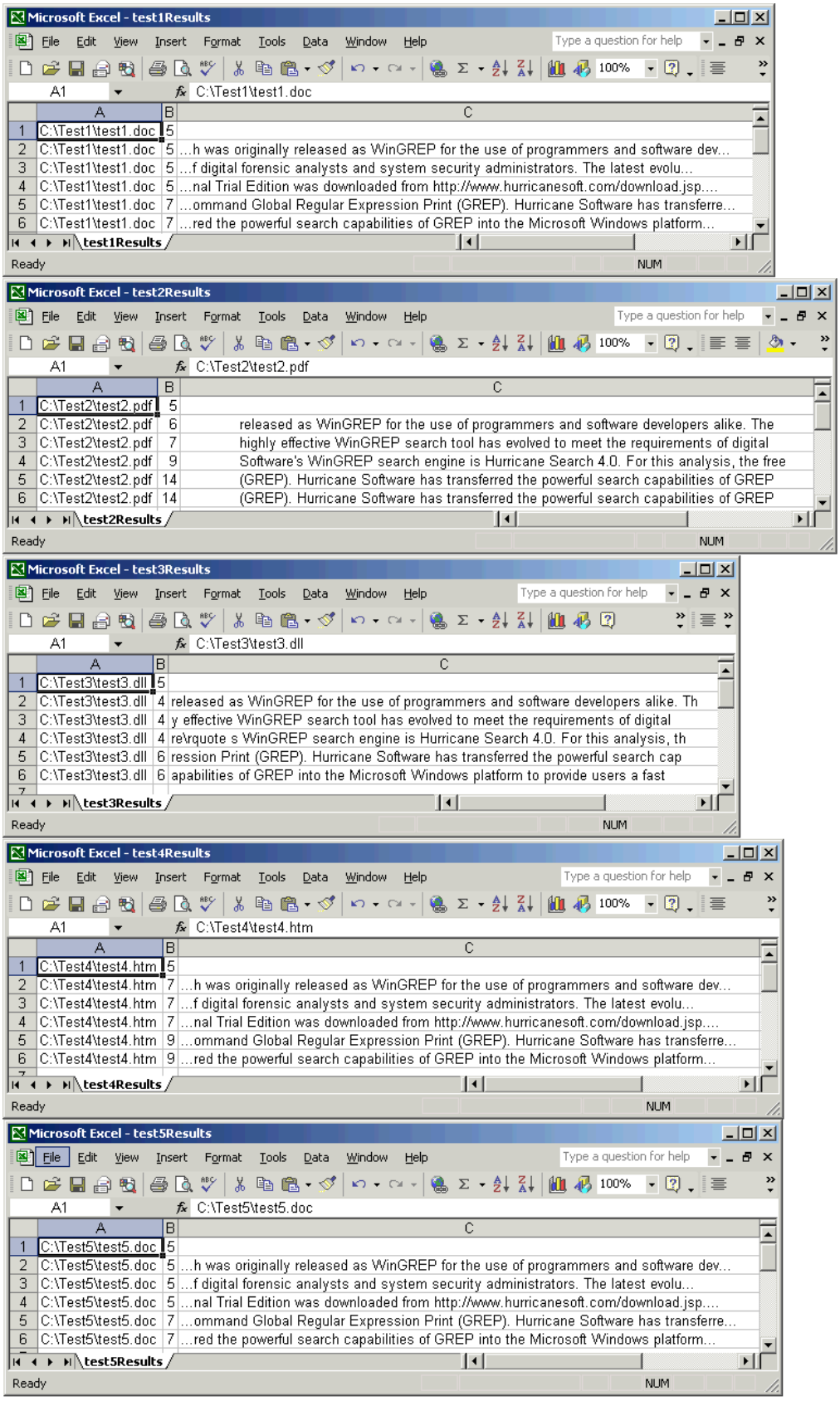


Figure 45 – Screen shot of keyword search results exported to Microsoft Excel

Analysis

Because a keyword search is an integral part of a computer forensics investigation, data (potentially in the form of clues) obtained using *Hurricane Search* is valuable to the investigator. The search results obtained by using *Hurricane Search* during a computer forensics analysis can aid the investigator in establishing potential clues of possible digital evidence such as passwords, names of hidden files, and other latent data relevant to the investigation. Because *Hurricane Search* enables the user to perform several search tasks simultaneously, collection of potential digital evidence is maximized. Further, the capability of exporting and saving search results allows the computer forensics investigator to revisit the potential digital evidence throughout the forensic investigation.

Presentation

Hurricane Search allows users to export the results of the text search into *Microsoft Excel*. The output from *Hurricane Search* is in comma delimited format, thus reports can be easily prepared and data can be presented in a court of law in a logical manner. As illustrated in Figure 46, the output is easy to read, even for a non-technical audience.

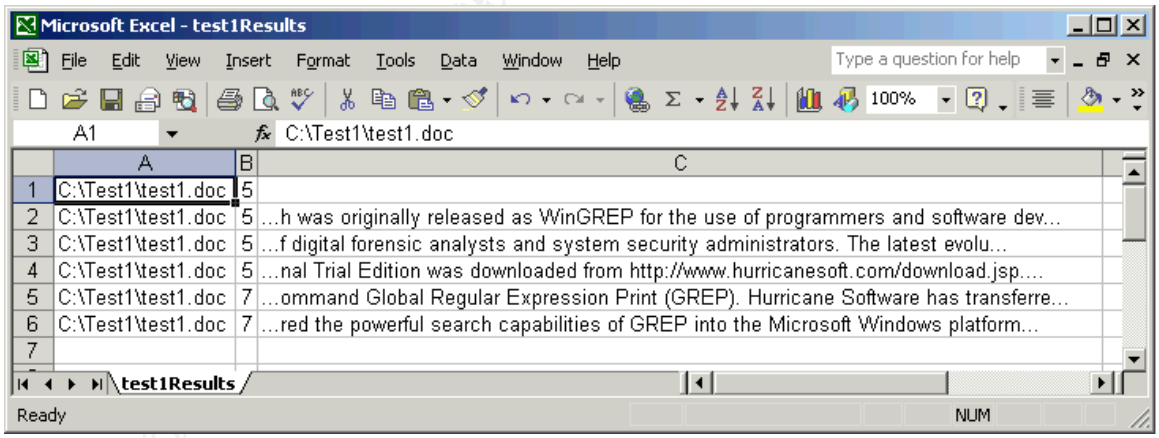


Figure 46 – Screen shot showing exported results

The first column contains the path of the file containing the keyword that was searched using *Hurricane Search*. In this case, it can be explained that the keyword was found in the file named ***test1.doc***, which is located within the directory **Test1**, which is located in the **C:** drive of the *Windows XP VMware* machine. The second column indicates the line number and the third column contains the syntax of the sentence that contains the keyword that was

searched.

However, explaining the results of any forensic investigation in a court is just one of many challenges faced by forensic analysts. The investigator must be able to prove that the results are accurate and the integrity of the evidence has been preserved.

Conclusions

The purpose of the forensic tool validation performed on *Hurricane Search 4.07*, formerly known as *WinGREP*, was to determine if: (i) digital evidence could be corrupted while using this tool during a forensic investigation, (ii) the results are verifiable and repeatable, and (iii) output is reproducible.

A keyword search was conducted to verify potential corruption of digital evidence during the use of *Hurricane Search*. To test the integrity of the test files in which the keyword search was performed, an **md5Sum** hash of the tested files was taken before and after the search using the tool *HashCalc.* The hash values of all five test files were exactly the same before and after the text search. Because hash values function as an electronic fingerprint, it can be concluded the test files were not corrupted during the text search.

To test verifiable and repeatable results, a keyword search was performed on five test files where each test file contained the same information: a description of *Hurricane Search.* In addition, each file was created either using a different application or saved with a different file extension. The same product of the keyword search was obtained for the individual searches. Thus the output from *Hurricane Search* is verifiable and repeatable. Lastly, to test if the results can be duplicated, the same five test files were executed on a separate workstation. Once again, the same output from the keyword search was obtained. Thus the conclusion can be drawn that the resulting output when using *Hurricane Search* is reproducible.

Hurricane Search is a valuable tool that can be utilized by computer forensic analysts during an investigation with the use of a dedicated investigation machine. Given that the tool must be installed in the system in order for it to execute, it should be used in a controlled setting such as on a dedicated investigation machine and not on a compromised system of interest. Nevertheless, as with any tool or methodology used during a computer forensic investigation, care must be taken to ensure that the integrity of the digital evidence is not jeopardized prior to, during and/or after analysis. A forensics analyst must keep in mind that when installing any tool or program in a computer, especially when dealing with *Microsoft Windows* platforms, certain program files, system files and libraries are modified, as was the case

with *Hurricane Search*. As a result, *Hurricane Search* is not recommended to be used on a live-system analysis during initial an incident response because the program must be installed on the system to run which would potentially corrupt crucial digital evidence.

For *Hurricane Search* to be more forensically sound, it would have to be designed in a matter that assures no data on the system under investigation is changed. One possible way of achieving such system integrity is for the tool to be compiled statically. A statically compiled program incorporates copies of system files and library routines necessary to run directly into the binary code of the executable program. As such, a statically compiled program should not alter any system files, thus preserving potential evidence intact. Further, the statically compiled program should be designed so that it runs from a bootable CD to further ensure that no external alterations are introduced to the system during the analysis. In summary, a forensically sound tool would not jeopardize the integrity of the system under investigation. Further, a forensically sound tool would allow a forensics investigator to obtain repeatable, verifiable, and reproducible results under comparable conditions.

References

Tools/Software

- Cygwin http://www.cygwin.com/mirrors.html
- Google http://www.google.com/
- Helix 1.4 “System Forensics, Investigation, and Response” CDROM
- Hurricane Software, Inc. “Hurricane Search 4.07 Professional Trial Edition“ http://www.hurricanesoft.com/download.jsp
- OllyDbg http://home.t-online.de/home/Ollydbg/download.htm
- SlavaSoft, “HashCalc“ http://www.slavasoft.com/hashcalc/
- Twisted Pear Productions, “Camouflage“ http://camouflage.unfiction.com/
- VMware, http://www.vmware.com/
- X-Ways Software Technology AG, “WinHex 11.8“ http://www.x-ways.net/winhex/forensics.html

Referenced Material

- ASCII table, http://www.asciitable.com/
- Bartlett, John. “The Ease of Steganography and Camouflage” John http://www.sans.org/rr/whitepapers/vpns/762.php
- Kruse II, Warren G. and Heiser Jay G., Computer Forensics Incident Response Essentials. Boston: Addison-Wesley, 2002. page 2
- Michaud, Erin. “Current Steganography Tools and Methods” http://www.giac.org/practical/GSEC/Erin_Michaud_GSEC.pdf
- Microsoft Corporation. http://msdn.microsoft.com/library/default.asp?url=/library/en-us/cpguide/html/cpconidentifyingfunctionsindlls.asp?frame=true&hidetoc=true

- Raggo, by Michael T. “Steganography, Steganalysis, & Cryptanalysis” http://www.blackhat.com/presentations/bh-usa-04/bh-us-04-raggo/bh-us-04-raggo-up.pdf

- Richer, Pierre. “Steganalysis: Detecting hidden information with computer forensic analysis” http://www.sans.org/rr/whitepapers/stenganography/1014.php

Legal

- Title 18, Crimes and Criminal Procedure http://www.cybercrime.gov/1832NEW.htm

- U.S. Department of Justice http://www.cybercrime.gov/1832NEW.htm

- United States Attorney’s Office http://www.usdoj.gov/usao/can/home.html

- United States Of America v. Trieu Lam and Thanh Tran http://www.usdoj.gov/usao/can/press/assets/applets/2004_11_04_Lam_ind.pdf November 03, 2004, CR 04 20198, CASBN 118321