



# Global Information Assurance Certification Paper

Copyright SANS Institute  
Author Retains Full Rights

This paper is taken from the GIAC directory of certified professionals. Reposting is not permitted without express written permission.

## Interested in learning more?

Check out the list of upcoming events offering  
"Advanced Incident Response, Threat Hunting, and Digital Forensics (Forensics  
at <http://www.giac.org/registration/gcfa>

# Computer Forensics Investigation

## Analyze an Unknown Image

Raúl Siles Peláez

December 21, 2004

*GIAC Certified Forensic Analyst (GCFA)*  
*(Version 2.0 - Option 1)*



## Abstract

This paper is the practical assignment required to obtain the GIAC Certified Forensic Analyst (GCFA) security certification (version 2.0 - Option 1). It consists on the investigation and forensic analysis of a piece of evidence, an USB flashdrive, collected during the incident response phase of a case involving personal harassment in CC Terminals.

The investigation focuses on obtaining a clear picture of the incident based on the analysis of the evidence gathered, establishing how it might have been used by the suspect.

In order to ensure success of the forensic process, four basic principles were followed: attempt to minimize data loss, record everything, analyze all the data collected and report the findings effectively. The methodology followed, the tools and procedures used and the conclusions obtained have been included in this paper, being as much accurate as possible. The report also covers legal issues related to the laws in my home country, Spain.

The forensic analysis has been mainly performed using open-source tools, because they are free and work well from a forensic perspective <sup>1</sup>.

## Acknowledgments

To our (my wife and I) grandparents, “here and there”, for making our life possible and much more enjoyable since we were young, and nowadays particularly to you, *Cecilio*. To our three grandmothers still alive, for being so lovely. Finally, specially to you, *Yayo*, because you taught me a lot and made my life different. I only would have wanted to let you know much more how important you were for me!!

Thanks to *Dan Farmer* and *Wietse Venema* for The Coroner’s Toolkit, to *Brian Carrier* for TSK and Autopsy, and in general, to all the open-source developers out there for their awesome contribution to the community.

Thanks to *Trinidad Sanchis* for the legal assistance, and *Jorge* and *David*: we are near the end of this very long trip.

*Mónica,*

I just want to share life with you in its whole extent, **now and always!!**

---

## PAPER LAYOUT

*The paper layout slightly varies from the schema suggested in the GCFA practical assignment. The whole disk analysis has been performed in chapter 2, the longest chapter, trying to improve the overall analysis readability. This information is later referenced when required in chapters 3, 4 and 5<sup>2</sup>.*

---

<sup>1</sup>Besides, their code can be analyzed to determine all the actions they accomplish over the evidence.

<sup>2</sup>This fact should be considered when evaluating the different sections and its length.

# Contents

<b>1</b>	<b>Executive Summary</b>	<b>5</b>
<b>2</b>	<b>Examination Details</b>	<b>7</b>
2.1	Evidence image extraction . . . . .	7
2.2	Evidence initial analysis . . . . .	7
2.3	Timeline creation and analysis . . . . .	11
2.4	OS-specific media analysis and Data Recovery . . . . .	12
2.4.1	File System Layer . . . . .	14
2.4.2	File Name Layer . . . . .	15
2.4.3	Metadata Layer . . . . .	16
2.4.4	Data Layer . . . . .	18
2.4.5	Slack space analysis . . . . .	23
2.5	Data Examination . . . . .	24
2.6	Strings Analysis . . . . .	32
2.7	Examination summary . . . . .	35
<b>3</b>	<b>Image Details</b>	<b>36</b>
3.1	Keywords associated with the programs/files . . . . .	37
<b>4</b>	<b>Forensic Details</b>	<b>40</b>
4.1	Program used by the suspect . . . . .	40
4.2	How the program used works? . . . . .	40
<b>5</b>	<b>Program Identification</b>	<b>47</b>
5.1	Compiling the WinPcap and WinDump sources . . . . .	49
<b>6</b>	<b>Legal Implications</b>	<b>54</b>
<b>7</b>	<b>Recommendations</b>	<b>56</b>
7.1	Victim's confirmation of evidences . . . . .	56
7.2	Suspect's computer forensic analysis . . . . .	56
7.3	CC Terminals infrastructure analysis . . . . .	58
7.4	CC Terminals physical and policy analysis . . . . .	59
7.5	Preventive actions . . . . .	59
<b>8</b>	<b>Additional Information</b>	<b>61</b>
8.1	"Digital Investigation" Journal . . . . .	61
8.2	National Institute of Justice (NIJ) . . . . .	61
8.3	The Sleuth Kit Informer . . . . .	61
8.4	Analyzing Windows binaries . . . . .	62
8.5	FAT filesystem layout . . . . .	62

8.6 E-Evidence Information and Resource . . . . .	62
8.7 Sniffing (network wiretap, sniffer) FAQ . . . . .	62
<b>A Chain of custody form and custody facilities</b>	<b>63</b>
<b>B Forensic analysis environment</b>	<b>65</b>
B.1 Forensic workstation . . . . .	65
B.2 Reverse Engineering workstation . . . . .	65
<b>C USB flashdrive timeline</b>	<b>67</b>
<b>D USB flashdrive timeline simulation session</b>	<b>69</b>
<b>E Allocated partition entry and boot sector analysis</b>	<b>71</b>
E.1 Disk allocated partition entry and CHS analysis . . . . .	71
E.2 Disk boot sector analysis . . . . .	72
E.3 Partition type change analysis . . . . .	73
<b>F USB flashdrive image extraction &amp; Methodology</b>	<b>75</b>
F.1 Methodology considerations & Reporting . . . . .	76
<b>G Compilation of WinPcap and WinDump</b>	<b>78</b>
<b>H MD5 value of all analysis files</b>	<b>81</b>
<b>References</b>	<b>83</b>

# 1 EXECUTIVE SUMMARY

The following summary is a chronological description of the investigation findings obtained during the forensic analysis of the USB flashdrive evidence, including what was found, how the conclusions were obtained and the laws that were broken.

All the conclusions obtained are based on the computer forensic analysis of the portable disk found on the suspect's cubicle. A exact copy of the disk was created in order to analyze all its contents without modifying them. Then, the first step performed was the extraction of the events occurred in the disk along time, what allows to know when each event happened. This is possible because the disk saves information about when a file or folder <sup>1</sup> has been created, read or used (for programs) and modified.

Then, an in-depth analysis of the contents was performed, using complex, reproducible and rigorous methods that preserve the copy integrity <sup>2</sup>. This allows to inspect the data contents of both, the files currently available on the disk and the files that were deleted. It is possible even to recover these deleted files if no other file has overwritten the space they occupied before. The analysis also allows to get the whole structure of the disk and its information, such as what files were contained in which folders, their sizes and the time information mentioned before.

Therefore, based on this methodology, the following facts were found confirming the victim's concerns. The morning of Monday October 25th, Robert Lawrence (the suspect) created a file called "her.doc" (see figure 2.24). This is the first of three files found that were sent as e-mails to Leila Conlay (the victim) based on her declaration; all them were created using the MS Word editor. The next morning, Tuesday October 26th, a new file called "hey.doc" (see figure 2.25) was created, containing more aggressive language.

The next day, Wednesday October 27th, the suspect obtained a program, called Win-Dump, whose main purpose is to capture all the communications taking place over a data network <sup>3</sup>. It was used on Thursday October 28th at 11:08am, to capture the communication associated to a victim's personal Hotmail session <sup>4</sup>. The contents of the e-mail were found in the disk, including a date with another person the same day in a coffee shop located on "Hollywood Blvd" and "McCadden" streets. Eight minutes after this information was captured, the suspect saved a map image of the coffee location (see figure 2.27) in the disk <sup>5</sup>.

Again, based on the victim's declaration, the evening of October 28th, at 7pm, she met with her friend at the coffee shop where the suspect appeared. This evening, 25 minutes later, a new aggressive message was written to the disk, including the suspect's complaints about the coffee meeting. The file is called "coffee.doc" (see figure 2.26), and it would be sent as a new e-mail to the victim.

The next afternoon, Friday October 29th, the victim's contacted CC Terminals corporate security, confirming that she was being harassed by the suspect, the numerous attempts he made to meet her, the increasingly aggressive nature of the messages received and the events occurred at the coffee shop.

---

<sup>1</sup>When the term "file" is used it includes both, files and folders.

<sup>2</sup>The contents are preserved and never modified from its original state.

<sup>3</sup>A detailed and accurate analysis of this file was performed to corroborate its purpose and functionality.

<sup>4</sup>Hotmail is a free, Web-based, e-mail Internet service offered by Microsoft.

<sup>5</sup>This map was obtained from the MS MapPoint commercial program.

Based on the facts discovered, the following articles of the Spanish law, “Código Penal”, apply once the victim has proceed to denounce them, by “Article 201.1” (see chapter 6):

- Article 197.1: for eavesdropping other's electronic communication without consent of the other party. Punishment of crime: 1 to 4 years prison and 12 to 24 months fine.
- Article 620: for the usage of new technologies to threat, offend or abuse others in an unjustified way. Punishment of offense: up to 6 months fine.

Additionally, the corresponding sanctions due to the violation of the CC Terminals company policy can be imposed.

© SANS Institute 2005, Author retains full rights.

## 2 EXAMINATION DETAILS

It is supposed that the security administrator involved in this incident, Mark Mawer, provided the original USB flashdrive found on Robert's cubicle, therefore the forensic methodology followed starts describing the steps followed to obtain the image file in which the whole analysis will be based on (see appendix F). From the point where the evidence was collected, the forensic methodology used can be summarized in the following phases <sup>1</sup>:

- Evidence image extraction: 22 Nov - 09:00 to 22 Nov - 15:30 (appendix F).
- Initial analysis: 24 Nov - 08:00 to 24 Nov - 18:00.
- Timeline creation and analysis: 25 Nov - 08:00 to 25 Nov - 19:15.
- OS-specific media analysis and Data Recovery: 27 Nov - 10:00 to 28 Nov - 13:00.
- Data examination: 29 Nov - 10:00 to 30 Nov - 16:00.
- String search: 29 Nov - 14:00 to 30 Nov - 18:00.
- Reporting: 22 Nov - 10:30 to 19 Dec - 23:30 (appendix F).

A chronological timestamp registry of when each stage of the methodology was started and finished was kept as part of the process.

### 2.1 Evidence image extraction

---

**From:** 22 Nov, 2004 (10:30) **To:** 22 Nov, 2004 (15:30)

---

Once the USB flashdrive was provided by Mark Mawer the 22nd of November, 2004 at 10:30 <sup>2</sup> the chain of custody form of appendix A was filled up after performing the extraction of the image. See appendix F for the image extraction process.

### 2.2 Evidence initial analysis

---

**From:** 24 Nov, 2004 (09:00) **To:** 24 Nov, 2004 (18:00)

---

The 24th of November 2004, at 09:00, the initial media analysis was started, processing and inspecting the disk image containing an exact copy of the evidence gathered at incident scene. The first step was the discovery of the disk type and the partitions within it, that is, the media management layer (sometimes called physical) analysis. In order to visualize the type and the drive partitions, the standard Linux `file` (v4.07) and `fdisk` (v2.12) commands were used (see figure 2.1).

The x86 boot sector type message (see figure 2.1) was obtained because the magic number 0xAA55 value is located in the 0x1FE offset within the image; defined in `"/usr/share/file/magic"` (see figure 2.2). `fdisk` shows 60959+ blocks (in Kbytes), and the + sign indicates the existence of an additional sector (512 bytes).

---

<sup>1</sup>See appendix F - section F.1 - for methodology considerations.

<sup>2</sup>All time references used along this report are based on the CET (Central European Time) timezone.



```
# file /T8c/usbfd-64531026-rl-001.img
/T8c/usbfd-64531026-rl-001.img: x86 boot sector
#
# fdisk -lu /T8c/usbfd-64531026-rl-001.img
You must set cylinders.
You can do this from the extra functions menu.

Disk /T8c/usbfd-64531026-rl-001.img: 0 MB, 0 bytes
17 heads, 32 sectors/track, 0 cylinders, total 0 sectors
Units = sectors of 1 * 512 = 512 bytes

               Device Boot      Start         End      Blocks   Id  System
/T8c/usbfd-64531026-rl-001.img1    *           32        121950        60959+    4   FAT16 <32M
Partition 1 has different physical/logical endings:
     phys=(249, 16, 32) logical=(224, 2, 31)
#
```

Figure 2.1: Image type and partitions

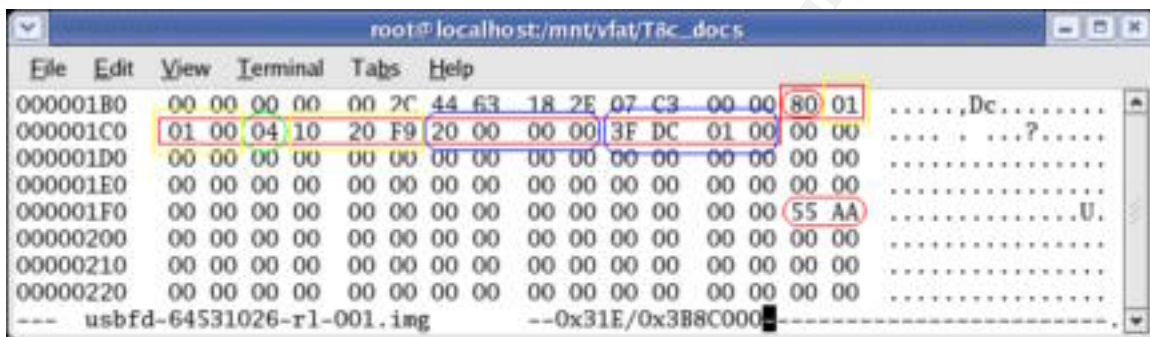


Figure 2.2: Bytes in the end of the first sector of the image (using hexedit)

The unique partition found is a 16-bit FAT partition, identified by type number 4 [PARTY1]. It was initially used in the DOS 3.0+ OS [MSDOS1] and typically required to be located in the first physical 32 MB of the hard disk, that explains the fdisk "FAT16 < 32M" string.

To corroborate the partition information, the Linux standard sfdisk command (v3.07) and the Sleuth Kit [SLEU1] mmls command (v1.72) were used. The drive has a FAT16 partition defined, so it seems to be a DOS-based partition disk (-t dos) (see figure 2.3).

```
# sfdisk -V /T8c/usbfd-64531026-rl-001.img
Warning: partition 1 extends past end of disk
#
# mmls -t dos /T8c/usbfd-64531026-rl-001.img
DOS Partition Table
Units are in 512-byte sectors

   Slot   Start      End      Length    Description
00:  -----  0000000000    0000000000    0000000001    Primary Table (#0)
01:  -----  0000000001    0000000031    0000000031    Unallocated
02:  00:00  0000000032    0000121950    0000121919    DOS FAT16 (0x04)
```

Figure 2.3: Disk partition information

The disk contains (see figure 2.4) the DOS partition table in its first sector (512 bytes) and the remaining 31 sectors of the first track are empty. Then, a bootable DOS FAT16 partition (Type 4) is occupying the rest of the disk space. In order to understand the image layout some mathematics were used (see table 2.1).

From these numbers its deduced that although a limited FAT16 partition type (0x04,

Disk section	Description	# bytes (Kbytes)	# sectors (512 bytes)
Image	Complete disk image	62439424 (60976)	121952
First track	First disk track (32 sectors)	16384 (16)	32
Partition Table	Primary disk table	512 (0.5)	1
Unallocated space	The remaining of the 1st track	15878 (15.5)	31
First partition	DOS FAT16 (<32M)	62422528 (60959)	121919
Missing space	Cluster boundaries	512 = 62439424-62438912	1 = 121952-32-121919

Table 2.1: Evidence image partitions and properties

less than 32MB) was defined, its size is greater than this value (60MB). Besides, it seems that the final disk sector from the total 121952 image sectors is not being used. The reason is the way the FAT16 partitions are created; its data section should follow the cluster boundary specification, in this case 1024 bytes or two sectors (see section 2.4.1 to get the cluster size), so it could not allocate half of a cluster, leaving an empty sector at the end of the disk <sup>3</sup>.

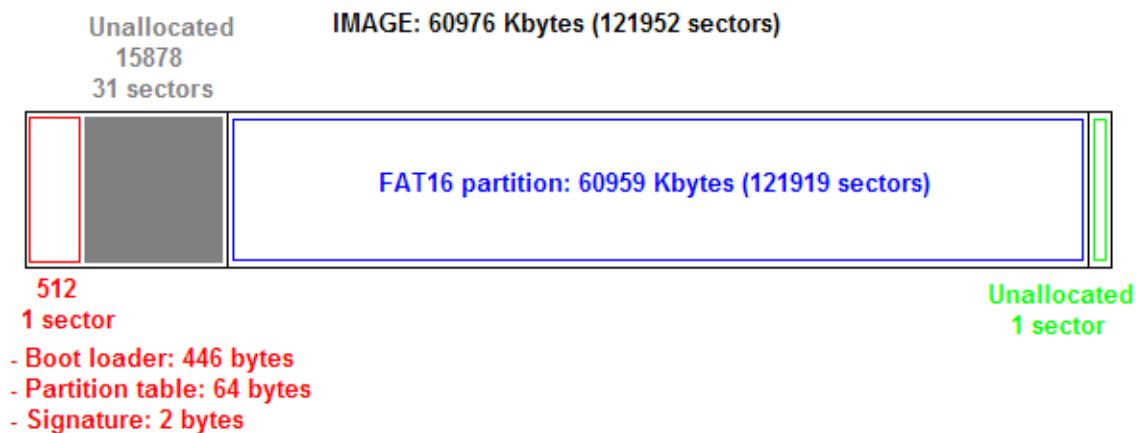


Figure 2.4: Summary diagram of the USB disk partitions

The next step in the process was the extraction of the different partitions found within the image. Using the `dd` command all the different pieces were extracted and their MD5 values verified (see figure 2.5).

Data could be hidden in the unallocated areas, so it was confirmed that the unallocated space and the last unallocated sector are completely empty (filled up with zeros) using the standard Linux `diff` (v2.8.1) and `dd` tools (see figure 2.6). For those cases where the amount of data to check is small, such as a couple of sectors, the `hexedit` (v1.2.7-2) hexadecimal editor can be used to look into the data contents <sup>4</sup>.

The last step of the forensic initial analysis is based on determining the partition type and checking if its contents can be accessed (see figure 2.7).

The analysis using the `file` command corroborates that the partition type (0x04, FAT16 < 32) and its contents (FAT16 greater than 32 MB) don't exactly match; the partition type

<sup>3</sup>A very detailed analysis of the allocated partition entry and the bootloader sector can be found on appendix E.

<sup>4</sup>If the data fragment analyzed is not empty (filled with zeros), `diff` will print out a message like the one showed for the "usb\_sector0" analysis, "Binary files ... differ".

```
# dd if=/T8c/usbfd-64531026-rl-001.img of=/T8c/usb_sector0 bs=512 count=1
1+0 records in
1+0 records out
#
# dd if=/T8c/usbfd-64531026-rl-001.img of=/T8c/usb_unallocated bs=512 count=31 skip=1
31+0 records in
31+0 records out
#
# dd if=/T8c/usbfd-64531026-rl-001.img of=/T8c/usb_FAT16 bs=512 count=121919 skip=32
121919+0 records in
121919+0 records out
#
# dd if=/T8c/usbfd-64531026-rl-001.img of=/T8c/usb_missing bs=512 count=1 skip=121951
1+0 records in
1+0 records out
#
# md5sum usb_*
5f830a763e2144483f78113a8844ad52  usb_FAT16
bf619eac0cdf3f68d496ea9344137e8b  usb_missing
5bf1cea807dec8655ed18b9bbf2ee918  usb_sector0
51596dda30fc38f0df3556d6f115256d  usb_unallocated
```

Figure 2.5: Image partitions extraction with dd

```
# ll
-r----- 1 root root      512 Nov 24 14:33 usb_missing
-r----- 1 root root      512 Nov 24 14:25 usb_sector0
-r----- 1 root root    15872 Nov 24 14:28 usb_unallocated
# dd if=/dev/zero count=512 bs=1 | diff - usb_missing
512+0 records in
512+0 records out
# dd if=/dev/zero count=15872 bs=1 | diff - usb_unallocated
15872+0 records in
15872+0 records out
#
# dd if=/dev/zero count=512 bs=1 | diff - usb_sector0
512+0 records in
512+0 records out
Binary files - and usb_sector0 differ
```

Figure 2.6: Checking contents of the unallocated disk areas

```
# file usb_FAT16
usb_FAT16: x86 boot sector, code offset 0x3c, OEM-ID "MSWIN4.1", sectors/cluster 2, root entries 512,
Media descriptor 0xf8, sectors/FAT 239, heads 17, hidden sectors 32, sectors 121919 (volumes > 32 MB)
, serial number 0x0, unlabeled, FAT (16 bit)
#
# mount -t msdos -o ro -o loop /T8c/usb_FAT16 /mnt/usb
# mount
...
/T8c/usb_FAT16 on /mnt/usb type msdos (ro,loop=/dev/loop0)
# ll /mnt/usb/
total 60
-rwxr-xr-x 1 root root 19968 Oct 28 20:24 coffee.doc
-rwxr-xr-x 1 root root 19968 Oct 25 09:32 her.doc
-rwxr-xr-x 1 root root 19968 Oct 26 09:48 hey.doc
```

Figure 2.7: Determining partition type and checking if it is accessible

should be 0x06 (FAT16)<sup>5</sup>. Additionally, it is possible to mount the partition (in read-only mode to preserve its integrity) and access its contents<sup>6</sup>.

<sup>5</sup>The partition can be mounted and accessed anyway (see appendix E).

<sup>6</sup>The Linux listed permission are based on the umask value of the process that executed the mount

## 2.3 Timeline creation and analysis

**From:** 25 Nov, 2004 (08:00) **To:** 25 Nov, 2004 (19:15)

A Forensic Timeline Analysis [TIME1] helps to identify **when** the main actions took place in the evidence patterns of activity, such as when the files were last modified, accessed or changed (metadata) and tools executed, providing details of the use of the USB flashdrive.

The 25th of November 2004 the timeline analysis was started using The Sleuth Kit v1.72 tools [SLEU1]. Initially we started knowing the window of time in which the incident occurred, around Thursday 28th of October.

Using the `fls` command, all the files and directories in the partition were listed, including “.” and “..”. Apart from this, we gathered details of the unallocated metadata through the `ils` command (see figure 2.8).

```
# fls -f fat16 -a -m -r /T8c/usb_FAT16 > timeline/body
# ils -f fat16 -m /T8c/usb_FAT16 >> timeline/body
# ll timeline/*
-r----- 1 root root 1979 Nov 25 08:31 timeline/body
# md5sum body
ec37d516b8b13748b4210ff6bc668352 body
```

Figure 2.8: Extraction of all image objects and timeline information

Once all the information was gathered, it was formatted nicely using the `mactime` tool<sup>7</sup>. No date range has been specified because we are interested in all the events contained in the USB flashdrive and the amount of information is quite small (see figure 2.9).

```
# mactime -b timeline/body > timeline/mactime.txt
# ll timeline/mactime.txt
-r----- 1 root root 5412 Nov 25 08:38 timeline/mactime.txt
# md5sum mactime.txt
22193dff66c4d62e135c30b97cb2682 mactime.txt
```

Figure 2.9: Mactime timeline generation

The conclusions obtained from the timing of events extracted from the timeline, considering the specific FAT behaviour (see appendix C), are:

1. The USB flashdrive activities started the Mon Oct 25 2004 at 08:32:06 and ended the Thu Oct 28 2004 at 19:24:48.
2. The file “her.doc” was created (and saved) on Mon Oct 25 2004 08:32:06.
3. The file “hey.doc” was created the next day, on Tue Oct 26 2004 08:48:06, also early in the morning.
4. The Wed Oct 27 2004 at 16:23:50 the Windows Packet Capture library, “WinPcap\_3\_1\_beta\_3.exe”, and at 16:24:02 (12 seconds later), the Windows Packet Capture Tool, “WinDump.exe”, were copied to the USB disk<sup>8</sup>.

command, that is, 0022. Windows FAT partitions doesn’t have the concepts of file permissions. The timestamp values will be analyzed later (section 2.3).

<sup>7</sup>Due to the fact that the “CC Terminals” timezone is unknown it has not been specified in the `mactime` (`-z`) and `ils` (`TZ`) commands; instead the forensic workstation TZ has been used: CET (Europe/Madrid).

<sup>8</sup>Both files correspond to Directory Entries 7 and 12.

5. Then, both the library and the tool, were finally copied again to the disk with Directory Entries 10 and 14. The reason is that the first instance of these files were a temporary copy used when they were obtained/downloaded (see appendix D).
6. The “originals” (7 and 12) were **last** accessed the Wed Oct 27 2004, however, the copies (10 and 14) were accessed (see next item) one day later, the Wed Oct 28 2004. So, the file at the Directory Entry 10 is a copy of the one at 7; and 14 is a copy of 12. All four files were deleted latterly.
7. The Thu Oct 28 2004 at 11:08:24 a capture file was created, therefore the WinDump tool (Directory Entry 14), using the WinPcap library (Directory Entry 10), was executed based on the access timestamps. At Thu Oct 28 2004 11:11:00 the capture file was saved, so the network traffic was captured during 2:36 minutes, the time range between 11:08:24 and 11:11:00. This capture file was also deleted afterward.
8. At Thu Oct 28 2004 11:17:44, about 8 minutes later, two GIF image files were created, corresponding to the Directory Entries 16 and 17<sup>9</sup>. Following the same reasoning as before (see appendix D) and given the fact that they have the same size, it can be stated that file with DE 16 is a temporary copy of 17, created during the acquisition process. Again, both image files were deleted thereafter.
9. Finally, at Thu Oct 28 2004 19:24:46 a new file was created, called “coffee.doc”.

In a FAT filesystem, each time a file is created, . . c, it is modified, m . . , almost immediately when its contents are saved to disk, although this event can be registered 2 seconds later<sup>10</sup>. Besides, an access event, . a . is also registered. For example:

```
Thu Oct 28 2004 00:00:00    19968 .a. -/-rwxrwxrwx    0 0 18 -r/coffee.doc
Thu Oct 28 2004 19:24:46    19968 .c -/-rwxrwxrwx    0 0 18 -r/coffee.doc
Thu Oct 28 2004 19:24:48    19968 m.. -/-rwxrwxrwx    0 0 18 -r/coffee.doc
```

The analysis performed over the evidence endorses the theory that in a FAT filesystem, when a file is copied, its modification time (the original file time, not the copy) appears before the creation time, but when it is created from an application, such as Internet Explorer, then the creation time appears before the modification (see appendix D).

As has been already mentioned, when a file is deleted in a FAT filesystem, the first character in the filename is lost, however, if the file is using long filenames, the whole name can be recovered. For example, this is the case of the “WinDump.exe” file. The FAT filesystem doesn’t leave any track about when files were deleted, so it is not possible to exactly determine when all the files discovered and not currently in the disk were deleted. At least we know they were all deleted the 28th of October or later, because, based on the timestamps, all them were last accessed this day.

## 2.4 OS-specific media analysis and Data Recovery

---

**From:** 27 Nov, 2004 (10:00) **To:** 28 Nov, 2004 (13:00)

---

<sup>9</sup>From a very strict perspective, and due to the fact that all the activities related with these files occurred in the same day, that the access timestamps don’t save the time value, and that the first letter of the filename is lost when it is deleted, in this case it wouldn’t be possible to know in detail the difference between both files and its usage.

<sup>10</sup>File timestamps on FAT drives are rounded to the nearest two seconds (even number) when the file is written to the drive: <http://support.microsoft.com/default.aspx?scid=kb;EN-US;127830>.

A detailed Media Analysis of the FAT16 partition was started on November the 27th using The Sleuth Kit [SLEU1] (v1.72) and the Autopsy [AUTO1] (v2.03) tools. To proceed with the media analysis autopsy was started (see figure 2.10) and a new forensic case was created through the Web interface: <http://localhost:9999/autopsy>. Using the “New Case” button, all the case descriptive information was registered, such as the case name “GCFA-v2.0-Option1-11-2004”, the description and the investigator name, “Raul\_Siles”. Then the new case was created in the Autopsy repository, called “Evidence Locker”, <sup>11</sup>.

It is very important to point out that during the Media Analysis phase, the different tasks associated to the recovery of the disk contents (“Data Recovery” phase) were performed, so, for clarity, both phases has been merged in a single section.

```
# autopsy

=====

Autopsy Forensic Browser
http://www.sleuthkit.org/autopsy/
ver 2.03

=====

Evidence Locker: /opt/EvidenceLocker
Start Time: Sat Nov 27 10:08:12 2004
Remote Host: localhost
Local Port: 9999

Open an HTML browser on the remote host and paste this URL in it:

    http://localhost:9999/autopsy

Keep this process running and use <ctrl-c> to exit
```

Figure 2.10: Autopsy Forensic Browser execution

Once inside the case (selecting it in the “Case Gallery”) a new host representing the USB flashdrive was added (through the “Add host” button of the “Host Gallery”). The host-name, “LexarUSB” and the description were set (all the other fields were left empty <sup>12</sup>). Finally, the host was selected in the “Host Gallery” and a new image, corresponding to the FAT16 partition was added through the “Add image” button <sup>13</sup>. Its location in the forensic workstation was set, “/T8c/usb\_FAT16”, the “Copy” import method was selected (verifying the copy MD5 value), its filesystem type selected, “fat16”, and the mount point chosen for cosmetic purposes was “A:\”.

Through the “Image Integrity” button of the “Host Manager” menu, the image MD5 value was validated. Autopsy also allows the creation of a file activity timeline using the “File Activity Time Lines” button (similarly to the one performed manually before, see section 2.3, so it was not used for this task). Along the analysis process the Autopsy “Notes” and “Events” functionality was used to document all the findings <sup>14</sup>.

Once the initial definition of the case, host and image were finished, everything was ready to start the media analysis. As can be seen in the menu bar on top of figure 2.11, Autopsy allows the analysis of the different filesystem conceptual layers, also defined in

<sup>11</sup>“/opt/EvidenceLocker/GCFA-v2.0-Option1-11-2004/”

<sup>12</sup>No hash databases has been used during this investigation; all hash checks has been performed manually.

<sup>13</sup>The other drive partitions, like the boot sector or the unallocated space were not added because they were manually inspected in the initial analysis phase due to their small size.

<sup>14</sup>These Autopsy reports have not been included because its contents are redundant with the paper itself.



Layer	Description	Autopsy menu	TSK commands
Physical layer	The drive or media itself; the partitioning	N/A	m-commands: mmls
Data layer	Real data blocks or clusters; the content	Data Unit	d-commands: dcat
Metadata layer	Structure information of files	Meta Data	i-commands: icat
File System layer	Structure information of filesystem	Image details	fs-commands: fsstat
File Name layer	Naming of metadata structures; human interface	File Analysis	f-commands: fls

Table 2.2: Forensic media analysis layers and tools relationship

The Sleuth Kit commands <sup>15</sup> (see table 2.2).

### 2.4.1 File System Layer

The most generic information related with a partition is in the “File System Layer” where the structure of the filesystem itself is described: unit sizes, offsets to critical data... See figure 2.11 and 2.12 for the USB FAT16 image details.

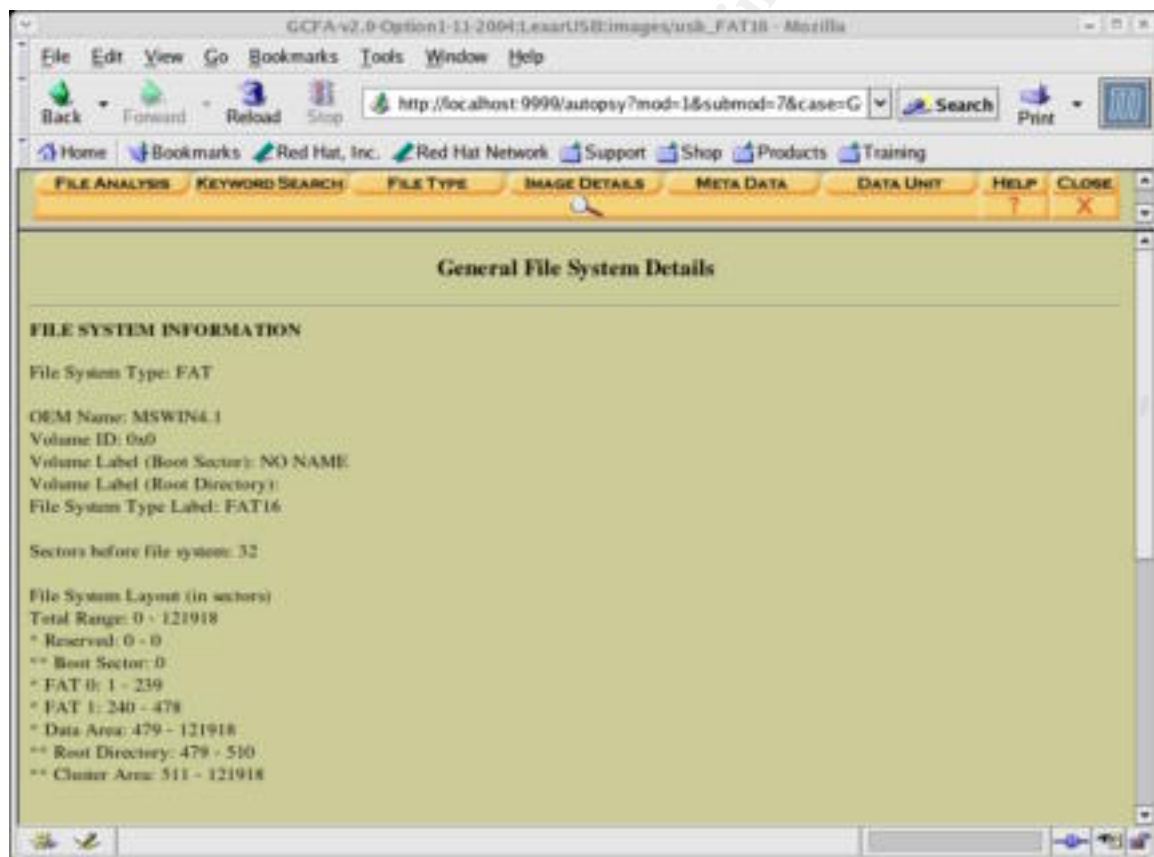


Figure 2.11: Autopsy FAT16 image details (Part 1)

A generic FAT16 partition can address 2 to the 16th power of addressable clusters, 65536 [MSFAT1]. This partition has 2 sectors per cluster, which is the default value for FAT partitions up to 64 MB [DISK1]. It uses one FAT entry (Directory Entry) per file [KAMP1]

<sup>15</sup><http://www.sleuthkit.org/sleuthkit/tools.php>

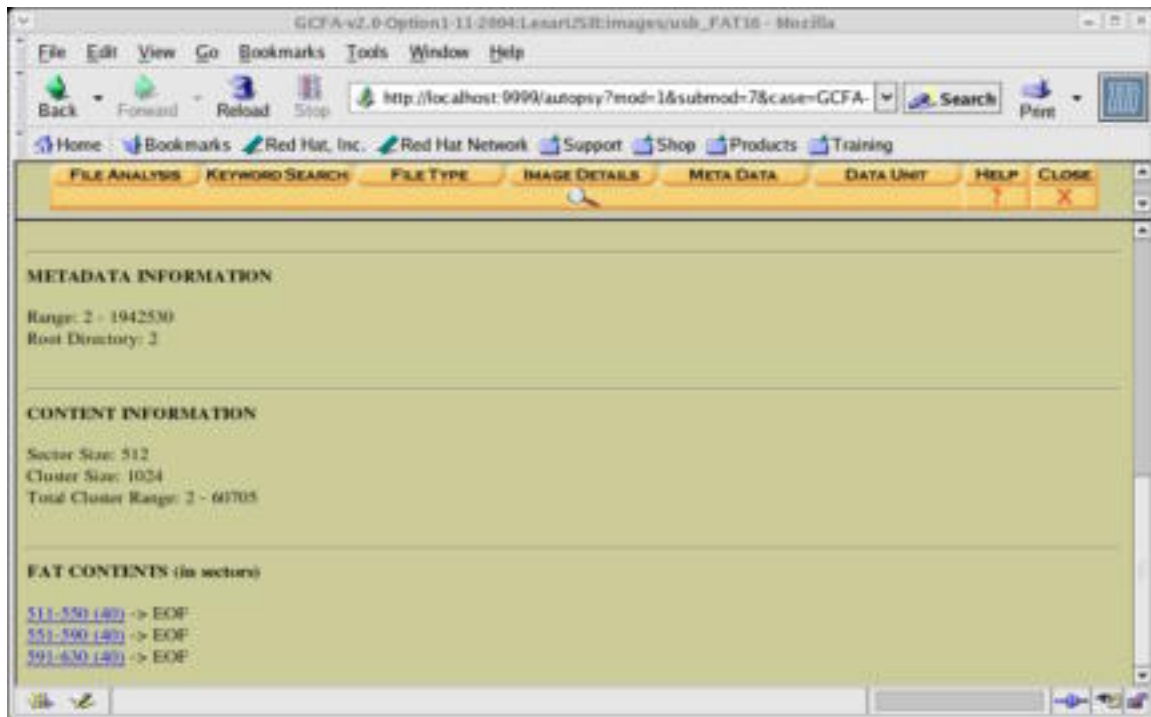


Figure 2.12: Autopsy FAT16 image details (Part 2)

<sup>16</sup> Finally, the standard Windows 8.3 naming convention is used plus the long filename support, available nowadays in any FAT partition (post Win95).

At the end of figure 2.12 the 3 entries corresponding to the 3 files available in this filesystem can be visualized. This partition occupies 121919 sectors based on the figure 2.13 layout:

- The metadata information fits in the first 479 sectors (from 0 to 478); each of the two FAT tables (FAT0 and FAT1) needs 239 sectors, and the additional sector belongs to the partition boot sector.
- The root directory gets 32 sectors (from 479 to 510) and it can hold up to 512 entries (32 bytes/entry).
- The data/cluster area corresponds to the next 121408 sectors (from 511 to 121918) until the end of the partition. Focusing on clusters (2 sectors) as the measurement unit (the data area unit), the data area has 60704 clusters (2 - 60705).

This explains why a missing sector was found in the initial analysis (see section 2.2), because the total partition size (121919) requires an odd number of sectors.

### 2.4.2 File Name Layer

The “real” in-depth media analysis started with the “File Name Layer”, the most human-readable layer. Through the Autopsy “File Analysis” menu it is possible to visualize the files and directories in a “File Browsing” mode, both existent (allocated) and deleted. In a FAT filesystem, the “File Name Layer” and the “Metadata Layer” are the same structure,

<sup>16</sup>The FAT is like the MFT in NTFS filesystems or the superblock in Linux ext2/3 filesystems.



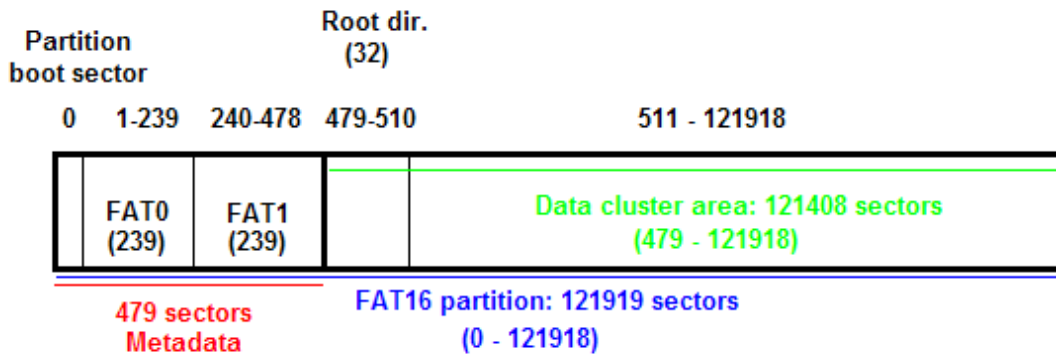


Figure 2.13: Summary diagram of the FAT16 partition

that is, the table of Directory Entries and the parent directory contain everything, the block pointers and the filenames.

There are no directories in the evidence media structure <sup>17</sup>, and only 3 files are available, all of the same size (19968 bytes), confirming the conclusions previously obtained in the timeline analysis (see section 2.3). Clicking on any of the non-deleted files (see figure 2.14) it is possible to check its type, "Microsoft Office Document", and contents. All three files were extracted using the Autopsy "Export" capabilities <sup>18</sup>.

Using the same procedure over the deleted files (showed in red in figure 2.15) it is possible to recover them, specifically the 3 ones associated to Directory Entries 17 ("\_ap.gif", "GIF image data, version 89a, 300 x 200"), 15 ("\_apture", "tcpdump capture file (little-endian) - version 2.4 (Ethernet, capture length 4096)") and 14 ("WinDump.exe (\_INDUMP.EXE)", "MS-DOS executable (EXE), OS/2 or MS Windows"). See chapter 3 for the complete file listing and section 2.5 for their details <sup>19</sup>. This semi-manual process can be automated using the Autopsy "File Type" functionality (based on `sorter`) that allows to categorize all the files in the image by type and even extract them <sup>20</sup>.

However the file of Directory Entry 10 ("WinPcap\_3\_1\_beta\_3.exe (\_INPCA 1.EXE)") seems to be empty (see figure 2.15) from the "File Analysis" point of view. If its Directory Entry, number 10, is accessed directly, from the "Meta Data" perspective (next section) it is possible to get more information about the file (see figure 2.16).

### 2.4.3 Metadata Layer

In order to recover this last file the analysis must be performed from the "Metadata Layer" perspective. The metadata is the information that describes how/where the real data is stored on the filesystem. The FAT Directory Entries contain pointers to the data layer sectors and other descriptive information, such as MAC times, size, filename. . .

To reconstruct the file associated to the Directory Entry number 10 we need to get

<sup>17</sup>The Autopsy "Expand Directories" button can be used to get this information.

<sup>18</sup>It would be possible at this time to perform a string analysis of the files, but this task will be executed in a later methodology stage.

<sup>19</sup>Files have been saved replacing the first letter, if missing, by its most probable guessed original value.

<sup>20</sup>For educational purposes the semi-manual option was considered more interesting for the reader.

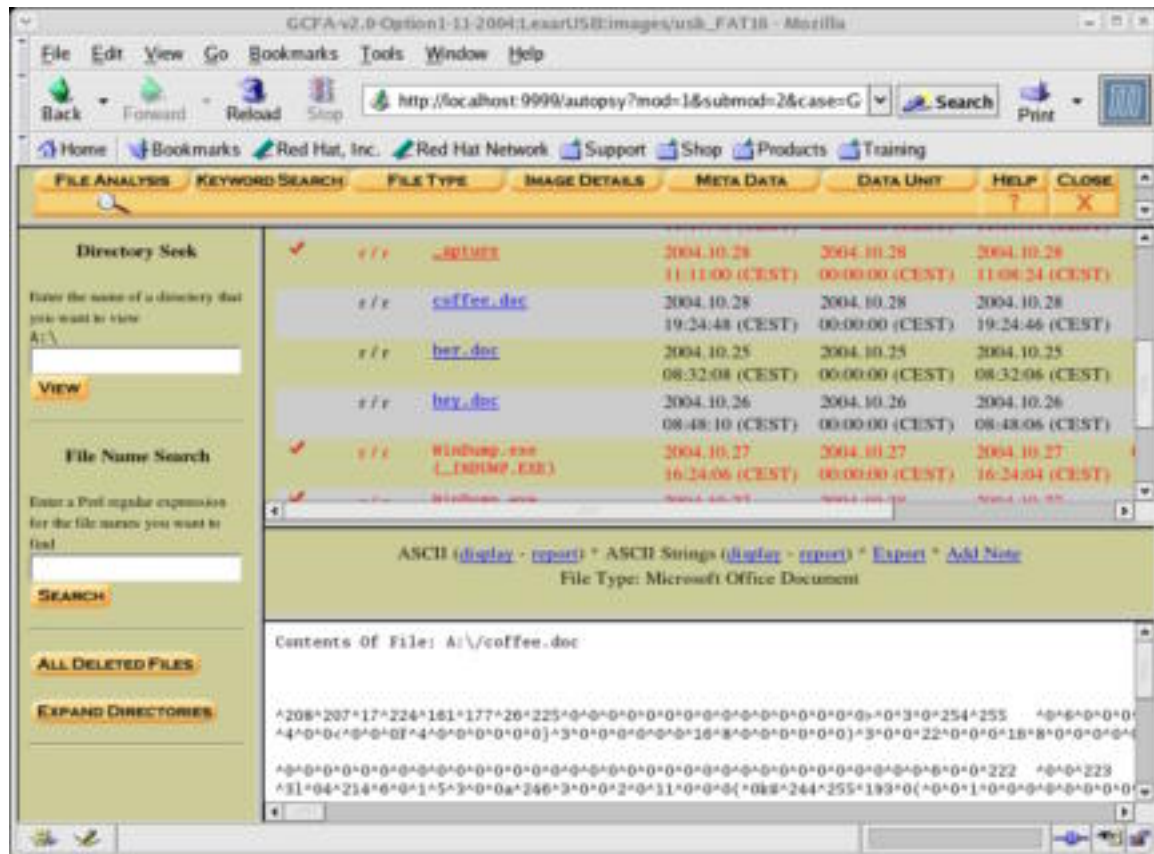


Figure 2.14: Autopsy File Analysis details

its block list, that currently goes from sector 591 to 630 (40 sectors)<sup>21</sup> (see figure 2.16). The theoretical file size is 485810 bytes, that is, 949 sectors, therefore the current pointer values are not valid.

Looking into the whole Metadata structure, specifically in the Directory Entries “Allocation List” functionality (see figure 2.17, left menu) it can be seen that there are only four entries occupied (“allocated”), the root directory (entry number 2) and the three “.doc” files (dir entries number 3, 4 and 18), and seven entries that have been occupied in the past and are currently free (7, 10, 12 and 14 to 17). This entries match the unallocated Directory Entries identified in the timeline analysis (see section 2.3).

The contents of all these 11 entries were strictly analyzed (see table 2.3)<sup>22</sup>. All the entries marked with a “(\*)” in the table have been round up to a cluster boundary, that is, an even number of sectors because this FAT16 partition has 2 sectors/cluster (see figure 2.12).

From this analysis we know that the sectors from 479 to 2542 have been occupied along time (currently or in the past). In a given moment, the file corresponding with Directory Entry 10, “WinPcap\_3\_1\_beta3.exe” was deleted, and its data sectors, from 591 to 1540 (949+1=950), were released (placed in the “Not Allocated” status)<sup>23</sup>. Then a new

<sup>21</sup>All sector (and unit) ranges used in this paper, unless otherwise specified, take both values inclusively.

<sup>22</sup>In fact, 17 Directory Entries (DE) were analyzed, 11 plus 6 entries used for long filename support. A long filename entries stores 13 characters of the name in Unicode [DISK1].

<sup>23</sup>Marked in the table with (\*). The “WinPcap\_3\_1\_beta3.exe” was also affected by cluster round up

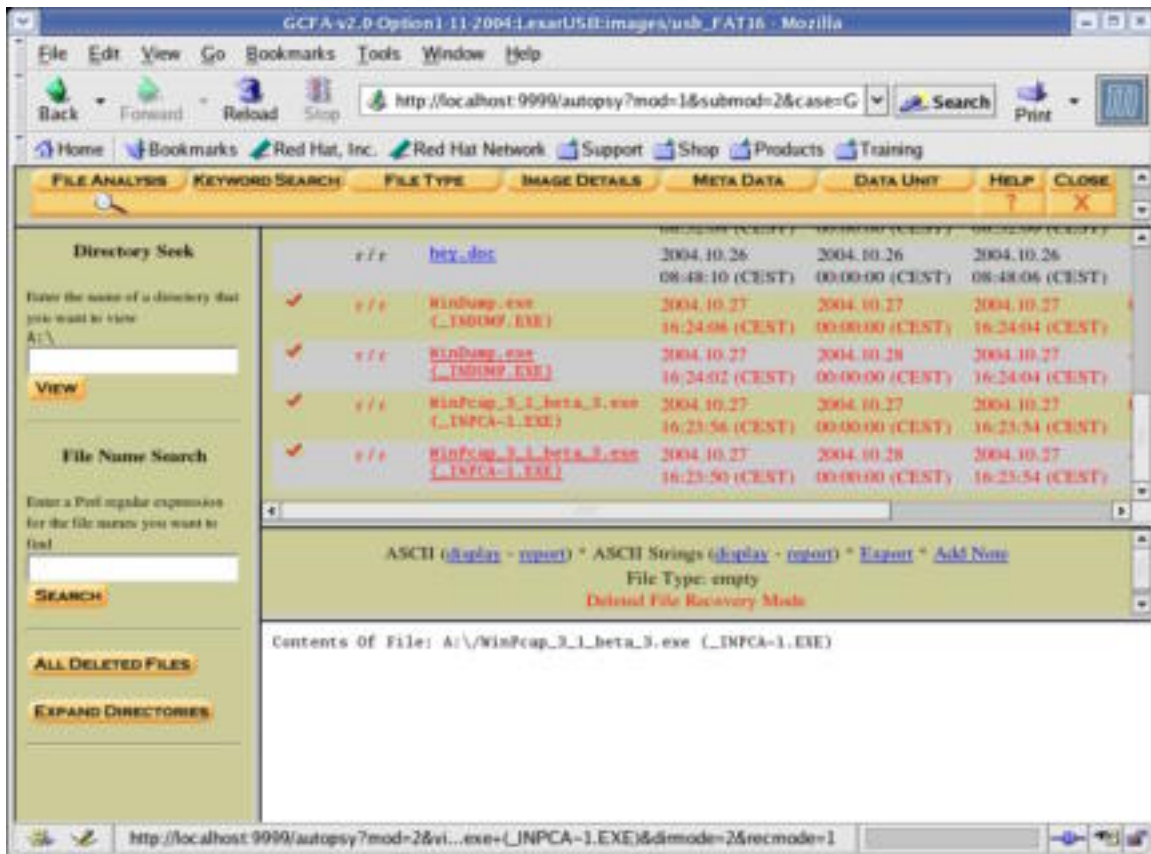


Figure 2.15: Autopsy File Analysis details (deleted files)

file was created, “coffee.doc”, using Directory Entry 18, and its data sectors overwrote part of the previous range, from 591 to 630 (39+1=40). That’s the reason why this file cannot be completely recovered. More specifically, only 39 sectors were overwritten; the last sector is slack space and was assigned to the file to match cluster boundaries (see section 2.4.5).

#### 2.4.4 Data Layer

The Data Layer represent where the real data is stored, typically in 512-byte sectors, as in this case. In FAT filesystems, sectors are organized and allocated together in clusters, the minimum filesystem addressable unit (defined in the ‘File System Layer’). In the evidence partition one cluster is made up of 2 sectors, 1024 bytes.

The data sectors could be in two states, allocated or in use, or unallocated or not being used (although they could have information inside). When a file is not fully recoverable, some pieces, called file fragments, can be still obtained. From the Autopsy “Data Unit” menu was possible to confirm the section 2.4.3 conclusions using the “Allocation List” functionality, all sectors from 0 to 630 are in the allocated state and all the rest are free.

Selecting the “details” link in the Autopsy “Host Manager” menu it is possible to access the Autopsy capabilities for extracting the image unallocated sectors (see figure 2.18); the

operation and it had 950 sectors reserved.

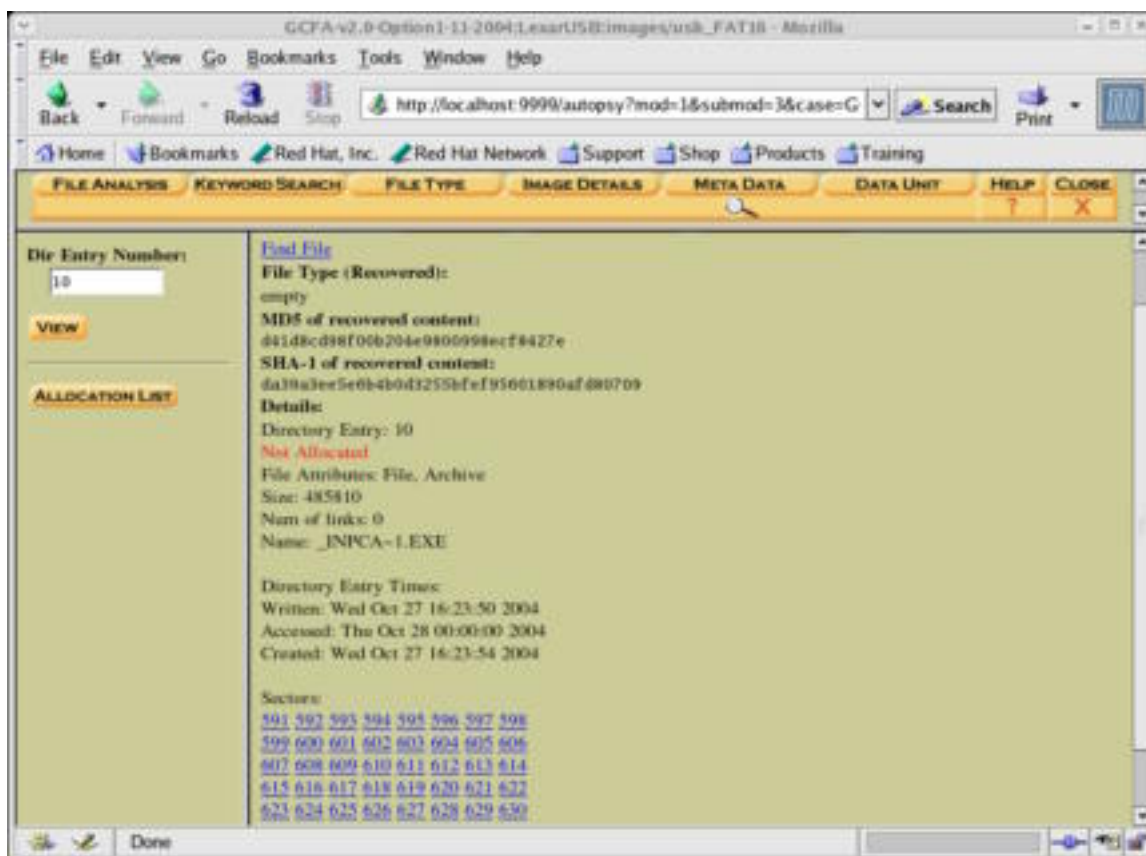


Figure 2.16: Autopsy Metadata details of Directory Entry 10 (WinPcap)

unallocated space could contain data from deleted files and may have evidence from the incident. Once the unallocated section was obtained (see figure 2.19), it is possible to work only with this image (selecting it, “unalloc”, in the “Host Manager” menu), running keyword searches or analyzing its data units <sup>24</sup>.

Therefore, based in the analysis performed in section 2.4.3, there is some data associated to the “WinPcap\_3\_1\_beta3.exe” file that can still be recovered. We would like to get the WinPcap fragment still remaining in the disk, not overwritten by the “coffee.doc” file. The original file resided in sector 591 up to sector 1540, and the overwritten area was between sector 591 and 630 <sup>25</sup>, so the remaining fragment resides between sector 631 and 1540. As explained previously, this should be the first 909 sectors of the unallocated space (everything is being used up to sector 630), because remember that sector 1540 was not part of the file but a cluster boundary addition, so it should be empty.

Figure 2.20 shows the extraction of the first 909 sectors of the unallocated space. This fragment was exported as file “WinPcap\_3\_1\_beta3\_fragment.exe”, with a size of 465408 bytes. This fragment corresponds in the entire original image (not the unallocated one) to sectors 631 to 1539.

This fragment could be extended or complemented adding at the beginning the unique

<sup>24</sup>The same steps (as with almost all the actions described along the forensic analysis) could be performed using only The Sleuth Kit command line tools, in this case, using the `dlis` command without arguments instead of Autopsy (see [http://www.sleuthkit.org/sleuthkit/docs/ref\\_fs.html](http://www.sleuthkit.org/sleuthkit/docs/ref_fs.html)).

<sup>25</sup>Remember that sector 630 was not really overwritten. A brief analysis can be found later.



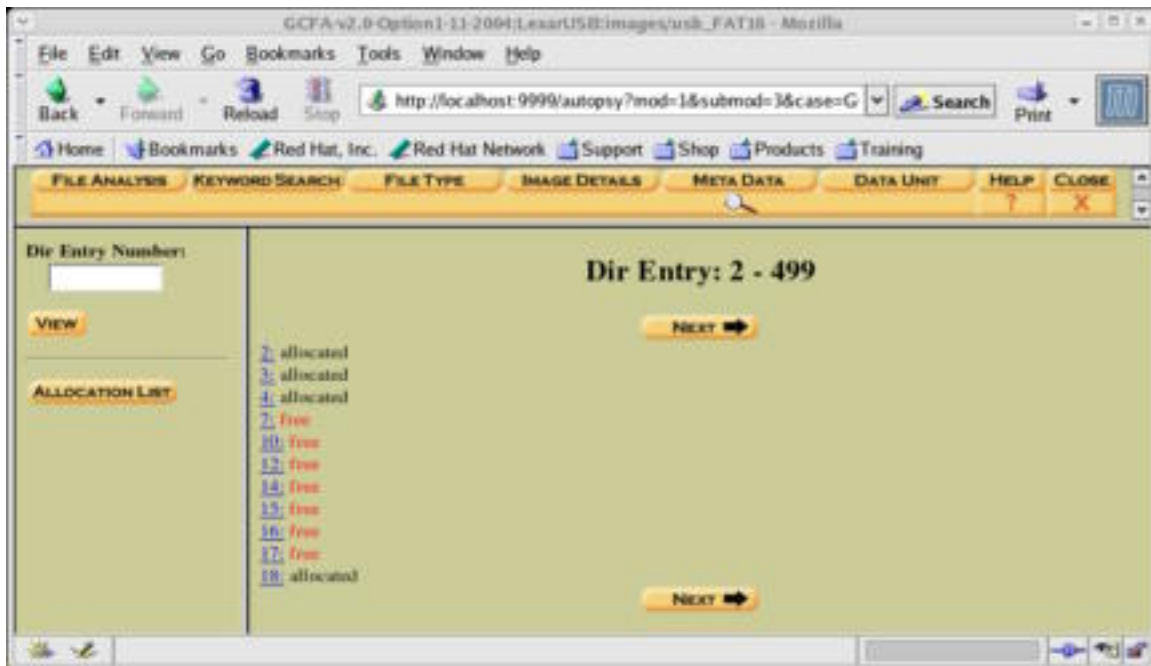


Figure 2.17: Autopsy Metadata details of all the used Directory Entries

DE	Type	Name	Size (sectors)	# sectors	Range	Free?
2	Directory	Root dir	16384 (32)	32	479-510	no
3	File, MS Office	her.doc	19968 (39)	40 (*)	511-550	no
4	File, MS Office	hey.doc	19968 (39)	40 (*)	551-590	no
5	Empty (Long File Name)	eta.3.exe				yes
6	Empty (Long File Name)	WinPcap.3_1_b				yes
7	Empty (File)	_INPCA 1.EXE				yes
8	Empty (Long File Name)	eta.3.exe				yes
9	Empty (Long File Name)	WinPcap.3_1_b				yes
10	Empty (File)	_INPCA 1.EXE	485810 (949)	950 (**)	591-1540	yes
11	Empty (Long File Name)	WinDump.exe				yes
12	Empty (File)	_INDUMP.EXE				yes
13	Empty (Long File Name)	WinDump.exe				yes
14	File, MS-DOS executable	_INDUMP.EXE	450560 (880)	880	1541-2420	yes
15	File, tcpdump capture	.apture	53056 (104)	104	2421-2524	yes
16	Empty (File)	.ap.gif				yes
17	File, GIF image data	.ap.gif	8814 (18)	18	2525-2542	yes
18	File, MS Office	coffee.doc	19968 (39)	40 (*)	591-630	no

Table 2.3: Listing of all the Directory Entries somewhere in time used (manual verification)

sector that is part of the slack space associated to the “coffee.doc” file (see section 2.4.5). This sector previously belonged to the WinPcap file and was not overwritten, it was assigned to the new file to round up to a whole number of clusters. For the purpose of recovering part of the file in order to analyze it (see chapter 5), adding this small portion is not very valuable so this step was omitted. The sector itself is analyzed in section 2.4.5.

Additionally, based on the previous “Metadata Layer” conclusions (see table 2.3), the first 1912 sectors of the unallocated area were used in the past ( $2542-631+1=1912$ ). So,

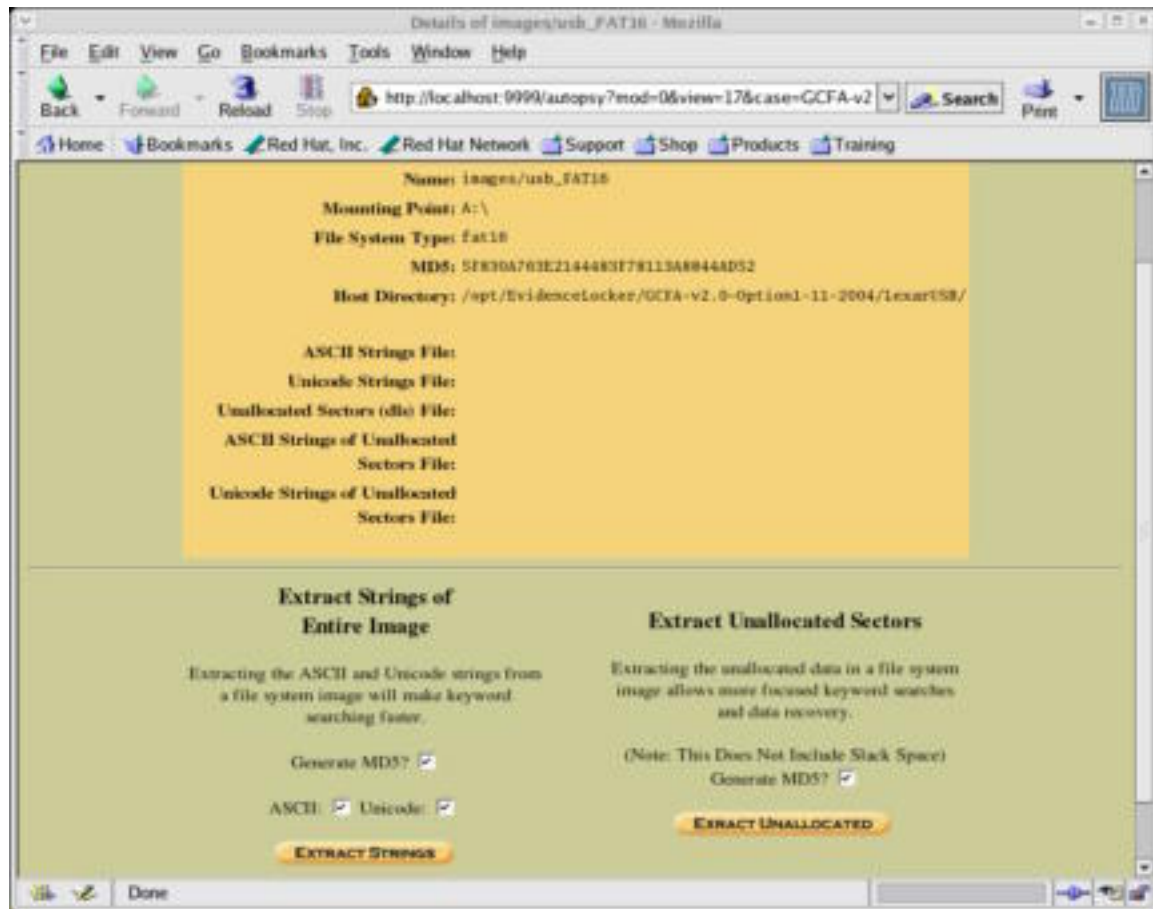


Figure 2.18: Autopsy strings and unallocated sector extraction capabilities

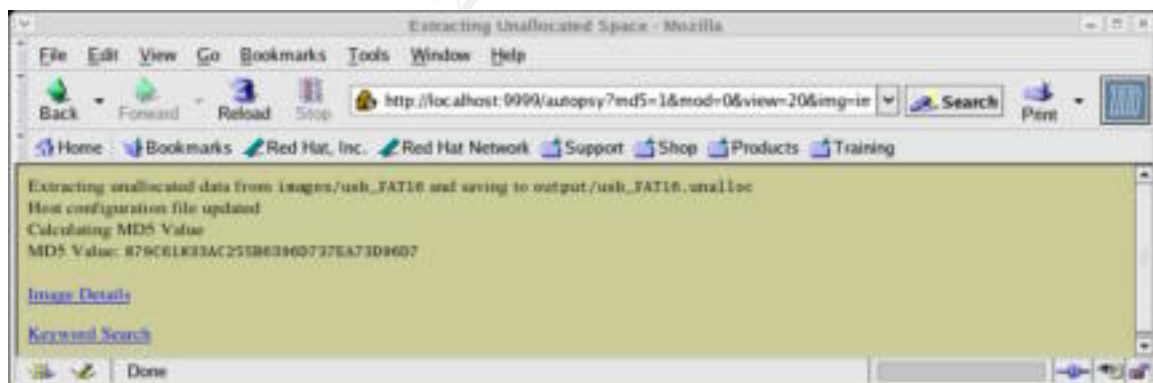


Figure 2.19: Autopsy unallocated image

from sector number 1912<sup>26</sup> until the end of the unallocated image everything should be empty. The area has 62099456 bytes or 121288 sectors (0 to 121287); subtracting the 1912 sectors containing data, we have 119376 sectors remaining. This area was extracted and analyzed (see figure 2.21).

The same method used in the initial analysis (see section 2.2) to check if a disk area

<sup>26</sup>Remember that Autopsy sector numbering starts at 0.

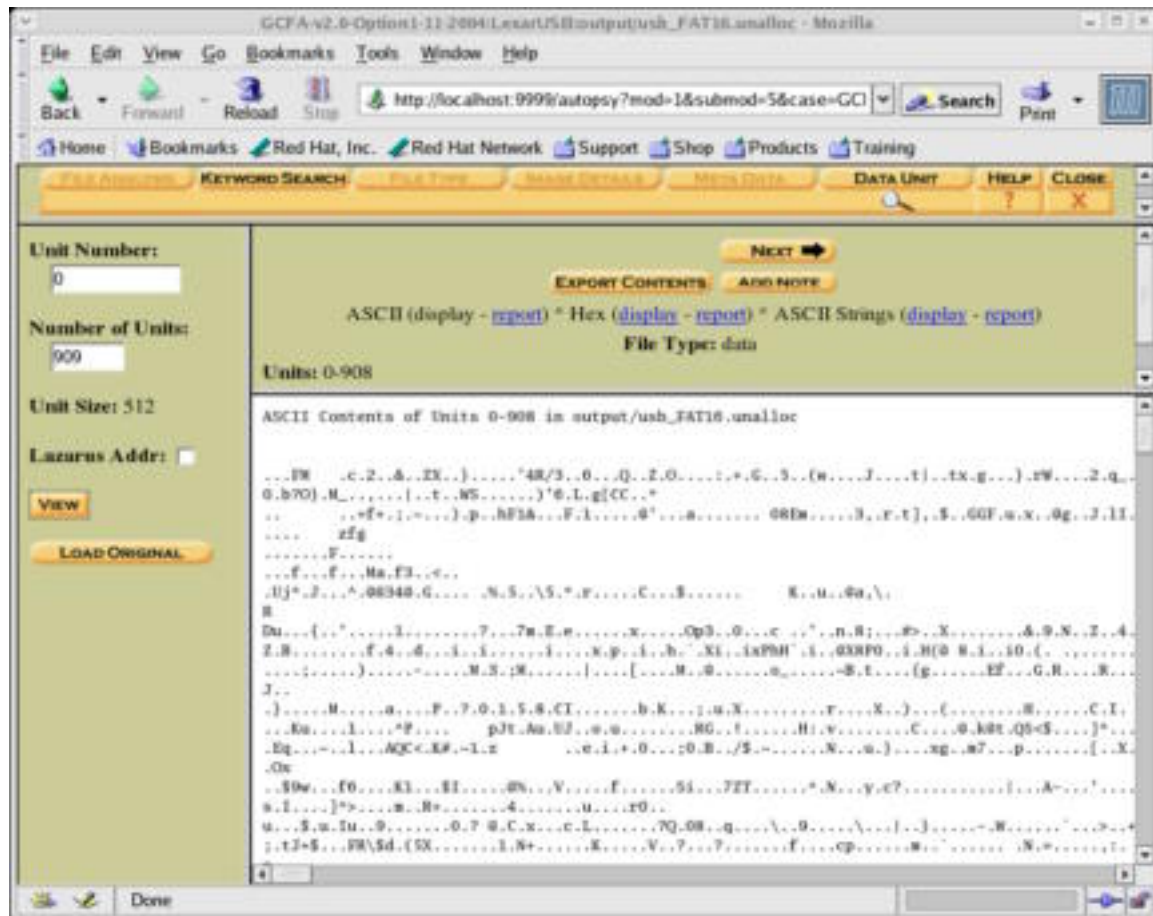


Figure 2.20: WinPcap fragment restored

```
# ll /opt/EvidenceLocker/GCFA-v2.0-Option1-11-2004/LexarUSB/output/usb_FAT16.unalloc
-rw-r--r-- 1 root root 62099456 Nov 27 20:24 \
/opt/EvidenceLocker/GCFA-v2.0-Option1-11-2004/LexarUSB/output/usb_FAT16.unalloc
# pwd
/opt/EvidenceLocker/GCFA-v2.0-Option1-11-2004/LexarUSB/output
# dd if=usb_FAT16.unalloc of=usb_FAT16.unalloc.rest bs=512 skip=1912
119376+0 records in
119376+0 records out
#
# dd if=/dev/zero count=61120512 bs=1 | diff - usb_FAT16.unalloc.rest
61120512+0 records in
61120512+0 records out
```

Figure 2.21: Analysis of the empty disk space

is composed only of 0x00 values, has been used (see figure 2.21); however, it has its advantages, once the first mismatch byte is found the processing stops, and disadvantages, it requires the same amount of forensic workstation RAM memory as the area/file size analyzed, in this case 60 MB. Based on the results, the rest of the unallocated space is empty so it is worthless to analyze it<sup>27</sup>.

<sup>27</sup>It was not possible to extract this unallocated area through Autopsy using the “Data Unit” option because the process hanged when joining together all these data units (60 MB), using the forensic workstation with 1 GB of RAM. The method used based on dd took around 8 minutes to complete.

## 2.4.5 Slack space analysis

Data can be hidden in the slack area, the region from the end of a file to the end of the cluster boundary it resides on. This area appears in files whose size doesn't exactly match the size of the clusters in which they are stored, and these 'leftover' bits in the cluster go unused. In FAT16 filesystems, where the cluster size increases based on the partition size, this can result in a very large amount of 'empty' space that could be used to covertly store data. In this case this area only represent 1 sector per file.

The `dls` tool [SLEU1] was used to extract the slack space. The extracted slack space file contains the slack spaces of all existent files marked with (\*) in table 2.3, that is, the additional sector of each of the three ".doc" files, reason why the slack space file has 1536 bytes (512 bytes x 3 sectors or files) (see figure 2.22).

```
# dls -s -f fat usb_FAT16 > usb_FAT16.slack
# ll usb_FAT16.slack
-r----- 1 root root 1536 Nov 27 21:22 usb_FAT16.slack
# hexedit usb_FAT16.slack
```

Figure 2.22: Slack space extraction from FAT16 partition

Its contents were inspected using the `hexedit` tool and the first and third sectors contained binary data from the files that resided previously (in the past) in these sectors. It is not possible to directly map data from the `dls -s` output to the original image file, but the table 2.3 would help in this task: the three extracted sectors should be numbers 550, 590 (empty) and 630 (the last sectors of each of the ".doc" files).

Using Autopsy over the entire USB image, through the "Data Unit" module, the contents of these 3 sectors were exported to 3 files. The non-empty files contents were inspected and its MD5 value obtained. Their strings will be inspected later (see section 2.6).

```
# md5sum images-usb_FAT16-Sector*
aae15c5605d8708dbd7e3fef53b50554 images-usb_FAT16-Sector550.raw
bf619eac0cdf3f68d496ea9344137e8b images-usb_FAT16-Sector590.raw
fc930a486a89ad02091e64870435e825 images-usb_FAT16-Sector630.raw
```

From other previous analysis we know that sector 630 was part of the "WinPcap\_3\_1\_beta3.exe" file (sector number 40), so the same file portion has been extracted from the official library file and both have been compared to confirm it contains this data (following a similar process as in chapter 5) (see figure 2.23).

```
# dd if=WinPcap_3_1_beta_3.exe of=WinPcap_3_1_beta_3_sector40.exe bs=512 skip=39 count=1
1+0 records in
1+0 records out
# ll WinPcap_3_1_beta_3_sector40.exe
-r----- 1 root root 512 Nov 28 11:07 WinPcap_3_1_beta_3_sector40.exe
# md5sum WinPcap_3_1_beta_3_sector40.exe
fc930a486a89ad02091e64870435e825 WinPcap_3_1_beta_3_sector40.exe
# md5sum /T8c/files/images-usb_FAT16-Sector630.raw
fc930a486a89ad02091e64870435e825 /T8c/files/images-usb_FAT16-Sector630.raw
```

Figure 2.23: Comparison of the WinPcap sector in slack space

Additionally, the slack space of the deleted files <sup>28</sup> (part of the unallocated space) can be inspected individually. Based on table 2.3 there is only one deleted entry that didn't

<sup>28</sup>As a suggestion, this very specific unallocated area could be called pseudo-slack space, because it is not "real" slack space (associated to the current files) but to the free disk area (deleted files).



fit in a whole number of sectors, the Directory Entry 10, whose last sector was 1540. Its contents has been analyzed through the Autopsy “Data Unit” menu confirming it is completely empty.

## 2.5 Data Examination

---

**From:** 29 Nov, 2004 (10:00) **To:** 30 Nov, 2004 (16:00)

---

Once all the different files and data pieces, seven in total, were reconstructed and its types identified (see section 2.4)<sup>29</sup> their sizes were compared with the values in the timeline, verifying that all them match, and their MD5 values were obtained. See chapter 3 for the results.

Up to this point, all the disk contents (data) have been recovered to its greater extent; the final step is the inspection of the contents of all the files found during the investigation, phase started on November 29, 2004.

The 3 allocated MS Word files have the contents of three mails potentially sent by the suspect to the victim. These documents have been opened using an appropriate interpreter, in this case MS Office Word 2003 (v11.6359.6360) SP1. They were sent in the following order based on the contents, “her.doc” (see figure 2.24), “hey.doc” (see figure 2.25) and “coffee.doc” (see figure 2.26), order that matches with the files creation timeline (see section 2.3).

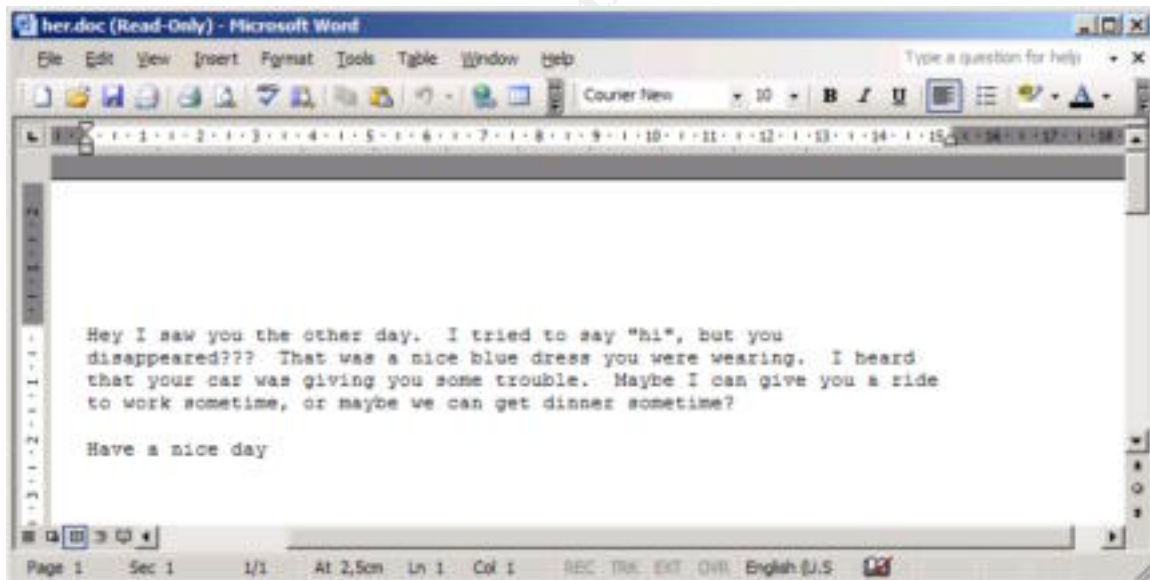


Figure 2.24: The “her.doc” file contents

The GIF image file, as its partially guessed name indicates, “\_ap.gif (map.gif)”, its a map file showing the address where the coffee where Leila was going to meet with a friend on October 28th is located, on the corner of “Hollywood Blvd” and “McCadden” streets (see figure 2.27 and the mail analysis below). As can be read on it, it is a map generated by Microsoft MapPoint<sup>30</sup>.

<sup>29</sup>Autopsy uses its own `file` command to get the file types, similarly to the standard Linux command.

<sup>30</sup><http://www.microsoft.com/mappoint/default.msp>

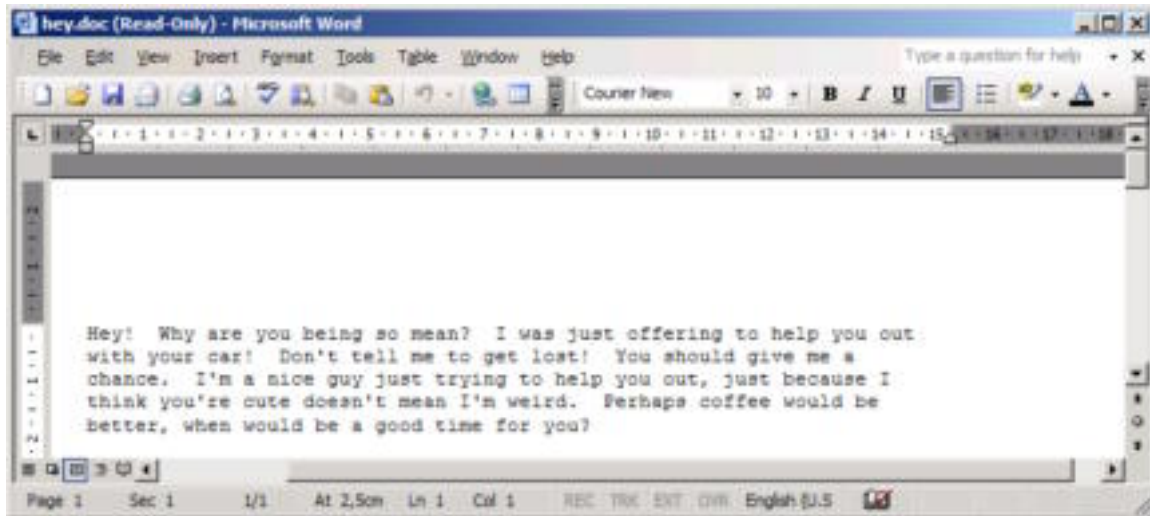


Figure 2.25: The “hey.doc” file contents

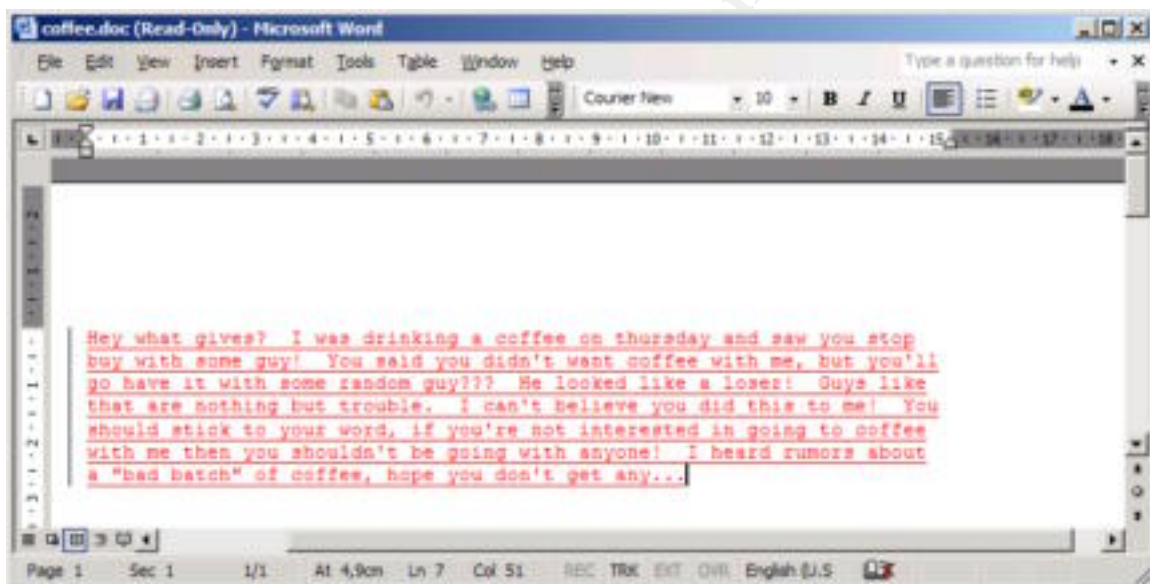


Figure 2.26: The “coffee.doc” file contents

MS MapPoint 2001 was used to locate the streets and identify the city. A search was performed using “Hollywood Blvd” first and “McCadden” next, and the first city found containing both in the expected timezone (see section 2.5) was Los Angeles. Extracting a map of the area, it exactly matches the one found in the USB drive (see figure 2.28)<sup>31</sup>.

The binary files, “WinDump.exe” and “WinPcap\_3\_1\_beta3\_fragment.exe”, are a Windows network traffic capture tool and its associated library (see chapter 5). The partially guessed name called “\_apture (capture)” is the output obtained when running the packet capture tool, that is, the network traffic captured. Its contents can be inspected with any libpcap-aware tool, such as ethereal (v0.10.5) [ETHE1]. This tool was used

<sup>31</sup>MapPoint references: “Hollywood Blvd, Los Angeles, CA 90028” & “N McCadden Pl, Los Angeles, 90028”.



Figure 2.27: The “map.gif” file map image

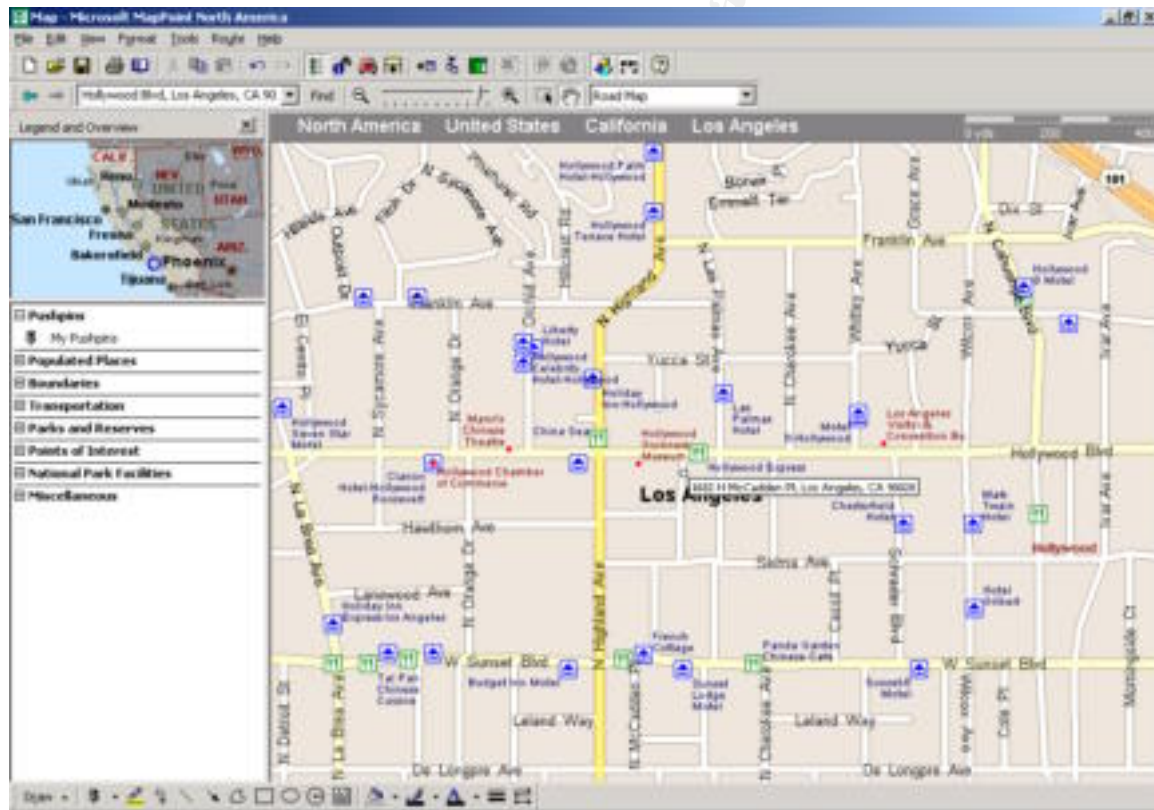


Figure 2.28: Matching map found through MS MapPoint 2001: Los Angeles, CA

(see figure 2.29) to analyze the traffic that was obtained from the network <sup>32</sup>.

The default time display format was changed <sup>33</sup> to “Date and Time of Date”, confirm-

<sup>32</sup># ethereal -n -r capture &

<sup>33</sup>“View – Time Display Format”.

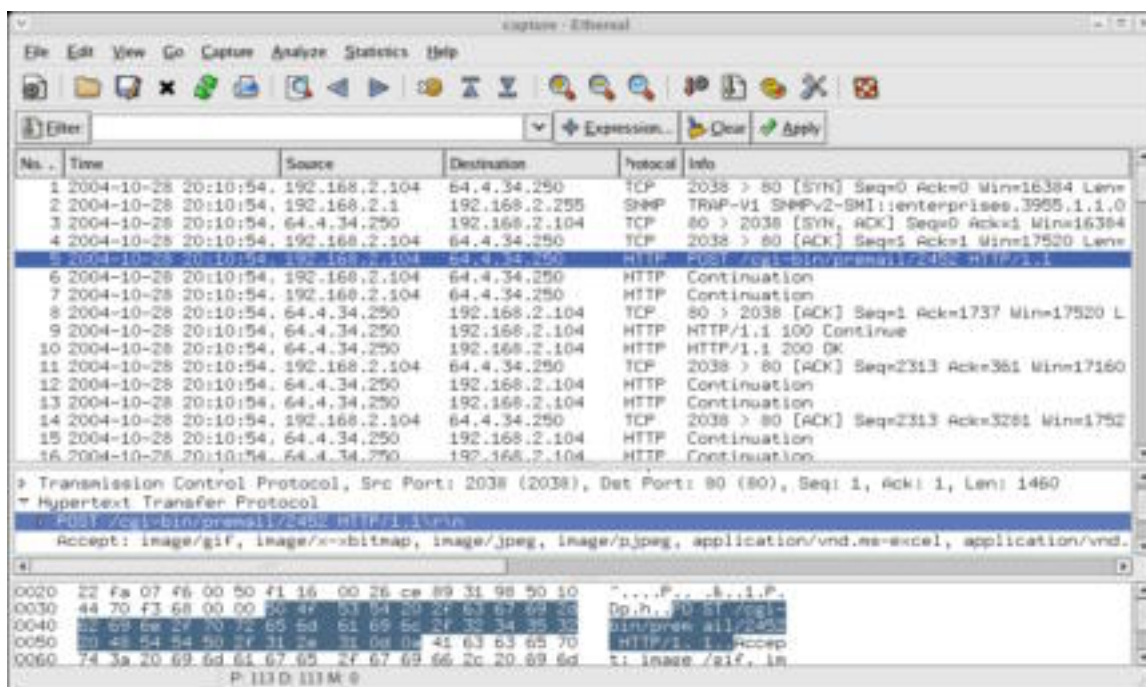


Figure 2.29: Network traffic captured visualized with ethereal

ing that the data was captured the 28th of October, from 20:10:54 to 20:10:55. This timestamps can be verified with the filesystem information (see section 2.3), however the timeline shows that the capture file was created in October 28th between 11:08:24 and 11:11:00. Ethereal displays the data applying the local timezone, so the difference of 9 hours is based on the timezones between CC Terminals and the forensic workstation (CET) <sup>34</sup> <sup>35</sup>.

As will be analyzed below, the capture file contents are very specific and “clean”, so they were probably filtered out <sup>36</sup>. For this reason, of the network traffic captured in the 2:36 minutes range (from –:08:24 to –:11:00) only packets belonging to a single second of this period, between –:10:54 and –:10:55, were saved to disk.

The network capture contains 113 network packets categorized in the types showed in figure 2.31 <sup>37</sup>. There are also 7 IP conversations, showed in figure 2.32 <sup>38</sup>. All them are related with the TCP protocol and host 192.168.2.104 (potentially the victim's IP address) except the last one, that corresponds to 6 UDP SNMP packets sent from 192.168.2.1 (probably the default company router/proxy) to the IP address 192.168.2.255 (probably the network broadcast address).

To confirm the computer and operating system behind the IP address found, 192.168.2.104,

<sup>34</sup>This means that CC Terminals is located in a timezone 9 hours later than Europe-Madrid, that is, in the US West coast, like San Francisco or Los Angeles. Definitely, CC Terminals is based in Los Angeles, based on the timezone analysis and the map addresses “Hollywood Blvd and McCadden”.

<sup>35</sup>Use <http://www.timezoneconverter.com> and specify date, time and timezones.

<sup>36</sup>The suspect could have used a similar capture filter as “(host 192.168.2.104 and tcp port 80) or udp port 162”, meaning all Web traffic associated with the victim IP address plus all SNMP Traps. WinDump cannot be used to selectively save specific marked packets, as Ethereal.

<sup>37</sup>“Statistics – Protocol Hierarchy”.

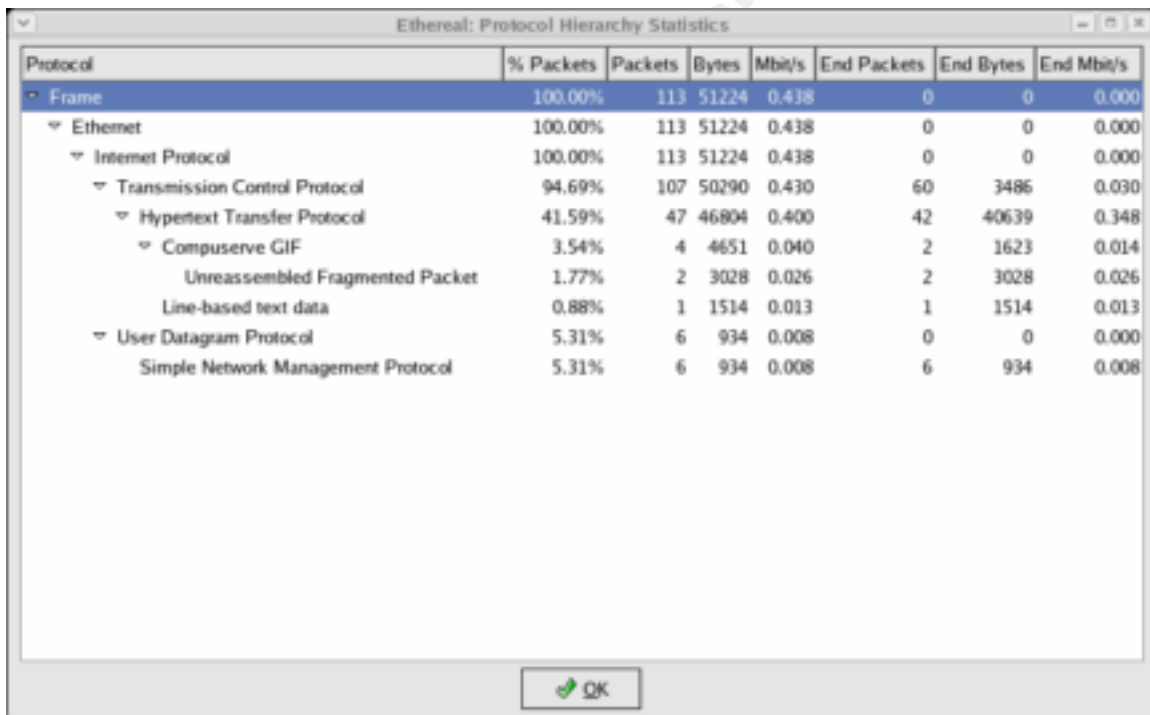
<sup>38</sup>“Statistics – Conversations”.



some basic network forensics and OS passive fingerprinting methods were used. As can be seen in table 2.4, this computer uses correlative source port numbers. All the packets generated by it have a TTL value of 128, typically associated to Windows computers. Its OS can be guessed more accurately based on the SYN packets features using a tool like p0f (v2.0.5)[P0F1] (see figure 2.30). The analysis confirms that the victim PC is a Windows 2000 SP2 (or greater) or Windows XP SP1.

```
# ./p0f -s /T8c/files/capture
p0f - passive os fingerprinting utility, version 2.0.5
(C) M. Zalewski <lcantuf@diode.cc>, W. Stearns <wstearns@pobox.com>
p0f: listening (SYN) on '/T8c/files/capture', 231 sigs (13 generic), rule: 'all'.
192.168.2.104:2038 - Windows 2000 SP2+, XP SP1 (seldom 98 4.10.2222)
-> 64.4.34.250:80 (distance 0, link: ethernet/modem)
...
192.168.2.104:2046 - Windows 2000 SP2+, XP SP1 (seldom 98 4.10.2222)
-> 63.166.13.75:80 (distance 0, link: ethernet/modem)
[+] End of input file.
```

Figure 2.30: Network forensics: OS identification using p0f



Protocol	% Packets	Packets	Bytes	Mbit/s	End Packets	End Bytes	End Mbit/s
Frame	100.00%	113	51224	0.438	0	0	0.000
Ethernet	100.00%	113	51224	0.438	0	0	0.000
Internet Protocol	100.00%	113	51224	0.438	0	0	0.000
Transmission Control Protocol	94.69%	107	50290	0.430	60	3486	0.030
Hypertext Transfer Protocol	41.59%	47	46804	0.400	42	40639	0.348
Compuserve GIF	3.54%	4	4651	0.040	2	1623	0.014
Unreassembled Fragmented Packet	1.77%	2	3028	0.026	2	3028	0.026
Line-based text data	0.88%	1	1514	0.013	1	1514	0.013
User Datagram Protocol	5.31%	6	934	0.008	0	0	0.000
Simple Network Management Protocol	5.31%	6	934	0.008	6	934	0.008

Figure 2.31: Categorization of the network traffic captured

Each of these SNMP Trap packets<sup>39</sup> is a message sent by the network gateway device notifying that a new outbound connection has been established. Six TCP connections establishment packets to new destinations (IP addresses) were initiated, so, six SNMP Traps were generated (corresponding to the first 6 entries of figure 2.32).

An Ethereal display filter was created to get the relationship between the initiation of the TCP connections and the SNMP Traps<sup>40</sup>. The SNMP notifications specify traffic direction,

<sup>39</sup>The SNMP read community used is the default one, "public".

<sup>40</sup>'(tcp.flags.syn == 1 and tcp.flags.ack == 0) or snmp'.

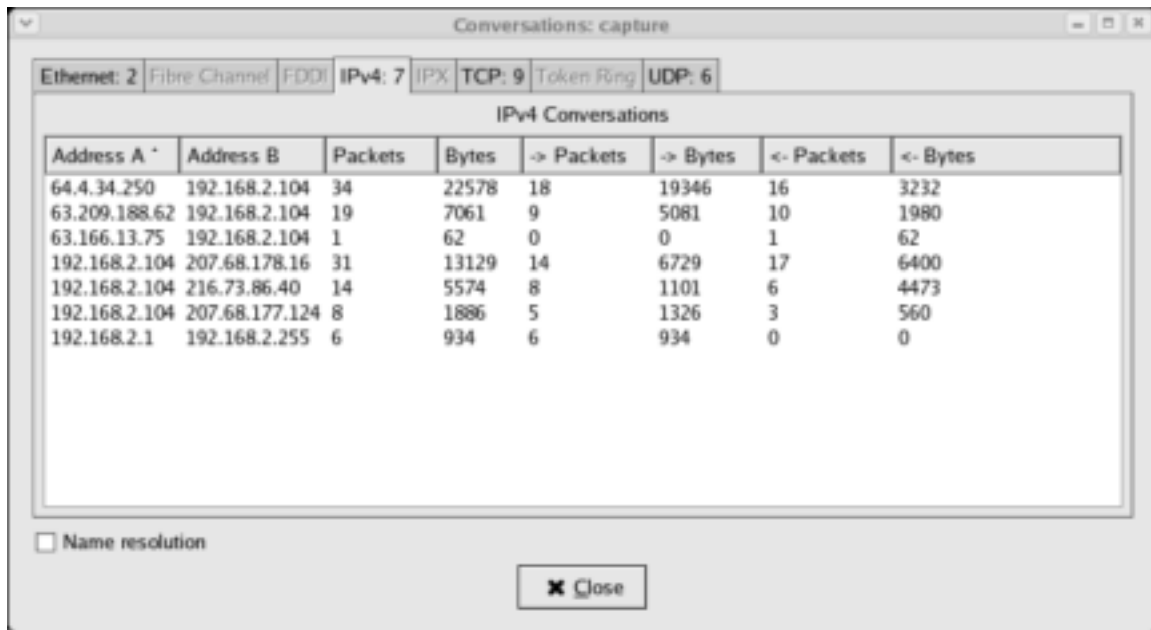


Figure 2.32: IPv4 network conversations in the network traffic captured

# TCP	# SNMP	Src IP	Src port	Dst IP	Dst port	HTTP request
1	2	192.168.2.104	2038	64.4.34.250	80	POST /cgi-bin/premail/2452
28	29	192.168.2.104	2039	207.68.178.16	80	GET /ADSAdClient31.dll? ... (3)
44	45	192.168.2.104	2041	207.68.177.124	80	GET /c.gif?...
52	53	192.168.2.104	2042	63.209.188.62	80	GET /ads/363/..., GET /ads/9073/...
82	83	192.168.2.104	2044	216.73.86.40	80	"RSTed by client"
111	112	192.168.2.104	2046	63.166.13.75	80	"Only SYN"
30		192.168.2.104	2040	207.68.178.16	80	GET /ADSAdClient31.dll?... (2)
67		192.168.2.104	2043	63.209.188.62	80	GET /ads/8707/...
84		192.168.2.104	2045	216.73.86.40	80	GET /adi/N1057.networksite...

Table 2.4: Listing of all the SNMP Trap packets and TCP sessions captured

"@out" for all them, and source and destination IP addresses and ports (see table 2.4<sup>41</sup>). There are three additional TCP establishment packets (see the bottom of table 2.4), but addressed to a destination IP address previously used, so no SNMP Trap was generated for them.

Therefore, as can be seen in figure 2.29, a TCP session, HTTP-based, from IP address 192.168.2.104 (port 2038) to IP 64.4.34.250 ("www.bay12.hotmail.com"), port 80, was initiated, accessing the Hotmail Web service<sup>42</sup>. This request is part of a Webmail Hotmail session, a free mail service owned by Microsoft<sup>43</sup> (see figure 2.33)<sup>44</sup>.

In order to analyze the 7 relevant TCP connections<sup>45</sup>, only the establishing packets

<sup>41</sup>First table columns reference the TCP and SNMP packet numbers within the capture file.

<sup>42</sup><http://www.bay12.hotmail.com/cgi-bin/premail/2452>

<sup>43</sup>This service requires in the client browser Javascript and Cookies support.

<sup>44</sup>When you access Hotmail (<http://www.hotmail.com>) your browser is redirected to a Microsoft login page, <http://login.passport.net/ui/login.srf?id=2>, belonging to its Passport account management solution, <http://www.passport.net>.

<sup>45</sup>Sessions starting with packets 82 (ad.doubleclick.net) and 111 don't have any content (see table 2.4).

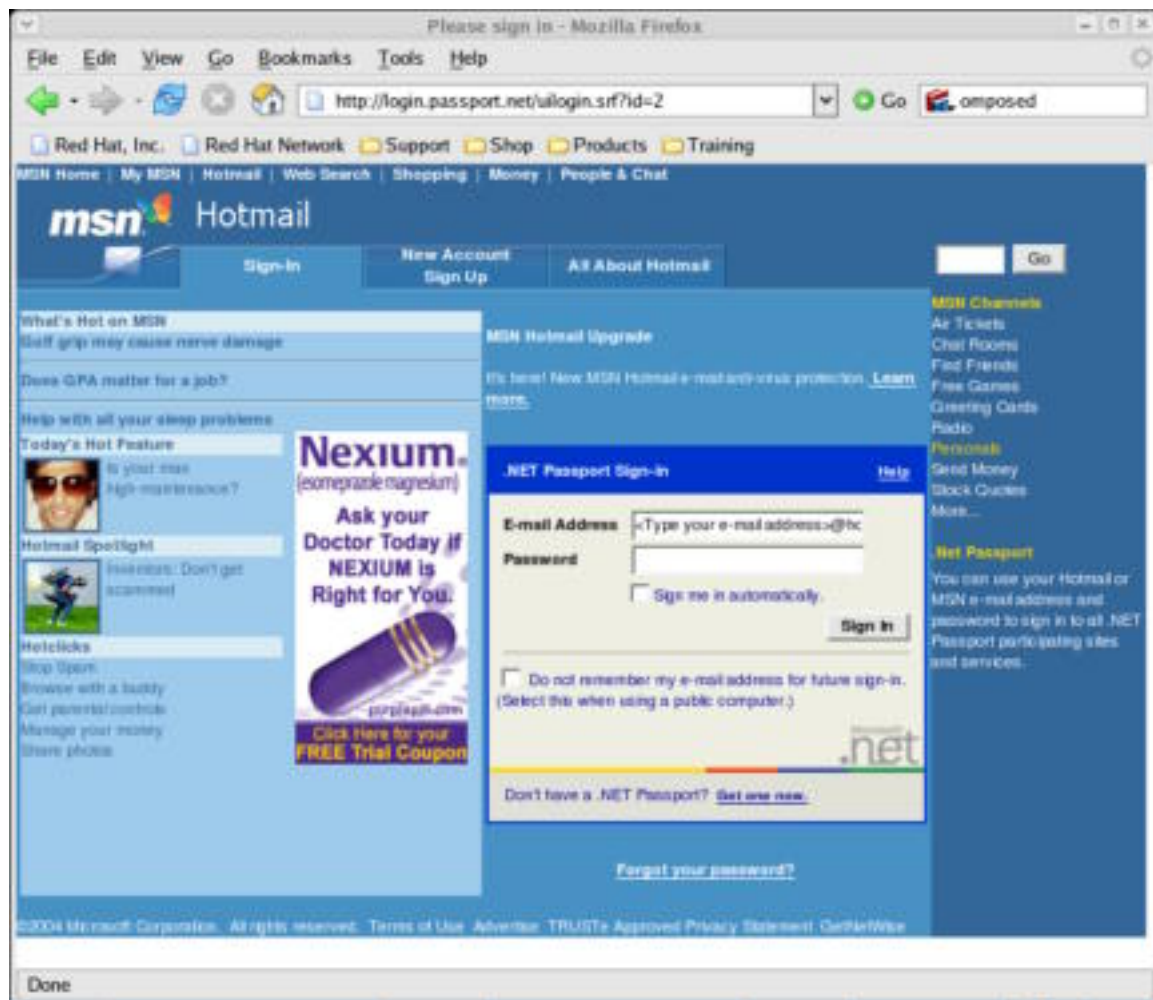


Figure 2.33: Hotmail default webmail login page

were obtained <sup>46</sup>, applying a new filter <sup>47</sup>. For each of them, the complete session was reconstructed and saved <sup>48</sup> (see figure 2.34 for the first TCP session). An in depth analysis of the 7 sessions has been performed inspecting its contents with an editor (*vi*) in the case of the HTTP/HTML sessions and using and hexadecimal editor (*ghex2*, v.2.8.1 <sup>49</sup>) for the HTTP/GIF transfers, in order to edit the session and extract only the GIF binary files <sup>50</sup>.

First TCP session (packet 1) contains the most relevant data; it corresponds to the Hotmail HTML web page confirming that a message has been sent from “flowergirl96@hotmail.com” to “SamGuarillo@hotmail.com” <sup>51</sup>.

The main evidence is not located in the HTML itself, but in the HTTP POST request that originated the HTML response. Based on the way the Hotmail service works, the

<sup>46</sup>Packets numbers: 1, 28, 30, 44, 52, 67, and 84.

<sup>47</sup>`'tcp.flags.syn == 1 and tcp.flags.ack == 0'`.

<sup>48</sup>“Analyze – Follow TCP Stream: Save As”.

<sup>49</sup><http://dag.wieers.com/packages/ghex/ghex-2.8.1-1.1.fc2.rf.i386.rpm>

<sup>50</sup>The contents of all these sessions has been detailed below.

<sup>51</sup>The session traffic contains some TCP duplicated packets: 19, 21, 23 and 25.

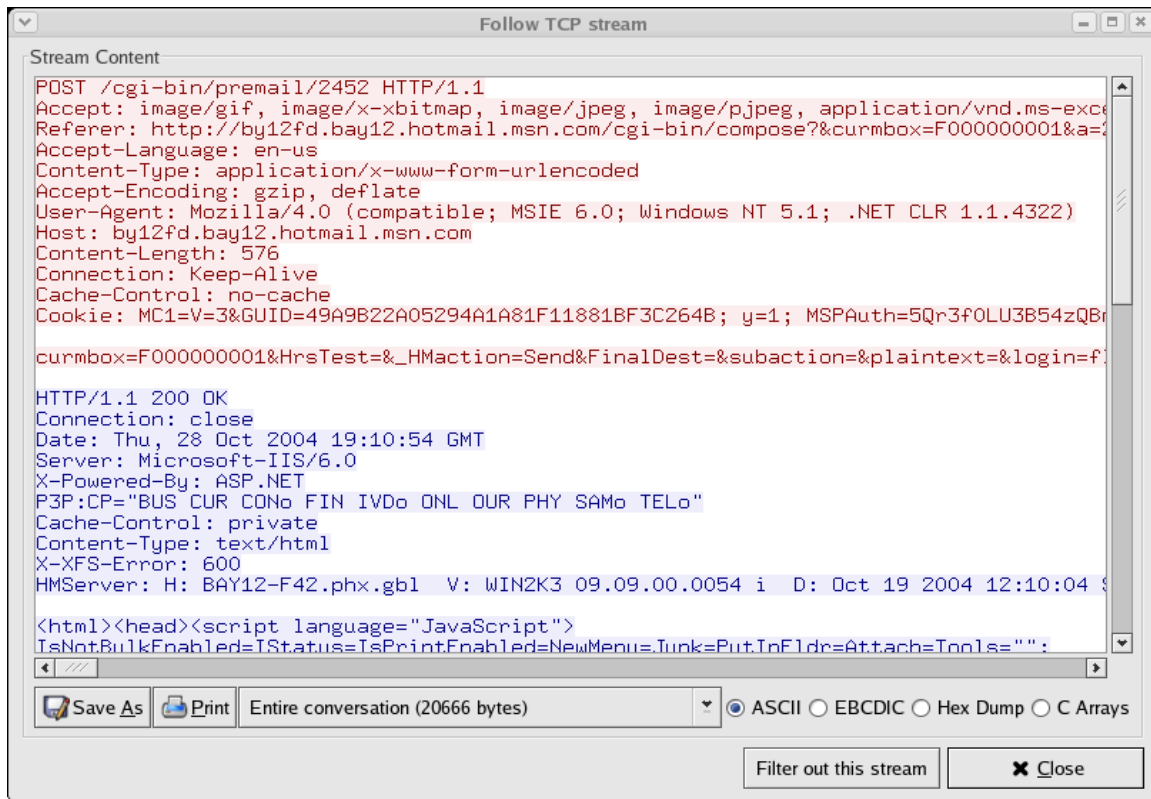


Figure 2.34: HTTP stream for the first TCP session (packet # 1 and related)

HTTP request contains a field called `curmbox` with the e-mail relevant information <sup>52</sup> <sup>53</sup>: `login="flowergirl96"`, `To="SamGuarillo@hotmail.com"`, `Subject="RE: coffee"` and `Body` (below) (see figure 2.35). The real e-mail body contents are in figure 2.36.

```
curmbox=F000000001&HrsTest=&_HMAction=Send&FinalDest=&subaction=&plaintext=&login=flowergirl96&
msg=&start=&len=&attfile=&attlistfile=&eurl=&type=&src=&ref=&ru=&
msghdrid=b16479b18beec291196189c78555223c_1098692452&RTBgcolor=&encodedto=SamGuarillo@hotmail.com&
encodedcc=&encodedbcc=&deleteUponSend=0&importance=&sigflag=&newmail=new&to=SamGuarillo@hotmail.com&
cc=&bcc=&subject=RE%3A+coffee&body=Sure%2C+coffee+sounds+great.++Let%27s+meet+at+the+coffee+shop+
on+the+corner+Hollywood+and+McCadden.++It%27s+a+nice+out+of+the+way+spot.%0D%0A%0D%0ASee+you+at+
7pm%21%0D%0A%0D%0A-Leila
```

Figure 2.35: HTTP header field: `curmbox`

```
Sure, coffee sounds great. Let's meet at the coffee shop on the corner \
Hollywood and McCadden. It's a nice out of the way spot.

See you at 7pm!

-Leila
```

Figure 2.36: Real e-mail message sent by Leila

To sum up, the traffic analysis has confirmed that an e-mail was sent from Leila's (potential) personal e-mail address, "flowergirl96@hotmail.com" to "SamGuarillo@hotmail.com",

<sup>52</sup>This field was also found searching for "Leila" in the "String Analysis" section, 2.6.

<sup>53</sup>The data is URL encoded [URL1].



(potentially) the friend with who she met at the coffee shop. The mail provided information to the suspect about where and when was Leila's appointment.

Session 28 contains three HTML files, two ads from <http://rad.msn.com> and a redirection to a <http://ad.doubleclick.net> ad. The same applies to the two HTML files of session 30, more MSN ads, and to session 84, another HTML-based doubleclick ad. Additionally, session 44 includes a small GIF file from <http://h.msn.com> and session 52 (two) and session 67 (one) GIF ads from <http://global.msads.net>.

## 2.6 Strings Analysis

---

**From:** 29 Nov, 2004 (14:00) **To:** 30 Nov, 2004 (18:00)

---

The string analysis could have been performed before, during the Autopsy "File Analysis" session, using the "Strings" links (see figure 2.14), but instead it has been performed in a final stage trying to follow an strict structured methodology.

The string search process (through keywords or regular expressions) should determine what keywords to look for and why. During the investigation a list of items that are specific to this case was created, called a "Dirty Word List", helping to prioritize the pieces of evidence to focus the efforts on<sup>54</sup>. This list includes specific keywords related to the case and was modified during the investigation based on other evidences discovered. A very basic Dirty Word List to start with for this case, based in the initial clues, would be<sup>55</sup>:

- People names: Leila Conlay and Robert Lawrence.
- Leila's computer IP address: 192.168.2.104.
- Coffee shop name or location: on the corner of Hollywood and McCadden.
- Leila's personal e-mail address: flowergirl96@hotmail.com.
- Dates: Thursday October 28th.

The visualization of the strings would also allow a quick analysis of the files contents and, for example, would have helped to determine the text contents of the three Microsoft Word documents found in this case if they would have been in a corrupted state.

In order to perform an string analysis of a Windows-based evidence from a Linux-based forensic workstation, like the USB flashdrive, the usage of special tools is required due to the different character sets used in both OS. Linux/Unix uses ASCII characters while Windows uses both, ASCII and Unicode. The processing of Unicode characters can be performed using Unicode-aware forensic tools, like Autopsy or the TSK's [SLEU1] sstrings command, or a Windows-based forensic workstation and its tools.

Up to this point there is only one piece of the investigation not clearly identified yet, sector 550, part of the slack space containing data. This is a CSV list of the strings contained in it, extracted from the "Data Unit" menu of Autopsy: i:VJ, 3wS, YYn , 0%A1, jMquw, r7 9, Msy:, jr)\_ , 5)Ks, voko.

Based on the result, it seems a fragment of an old, deleted binary file. Using the Autopsy search capabilities (explained below) over the entire image, the string "voko" was only found on sector 550, so it doesn't seem to be related with (or be a copy of) any other filesystem unit (see figure 2.37) currently available on the disk.

<sup>54</sup>Autopsy includes predefined searches for IP addresses and dates.

<sup>55</sup>Some of them have been found during the investigation but in a real case could have been obtained in advance from the victim, in this case the person being harassed, Leila.

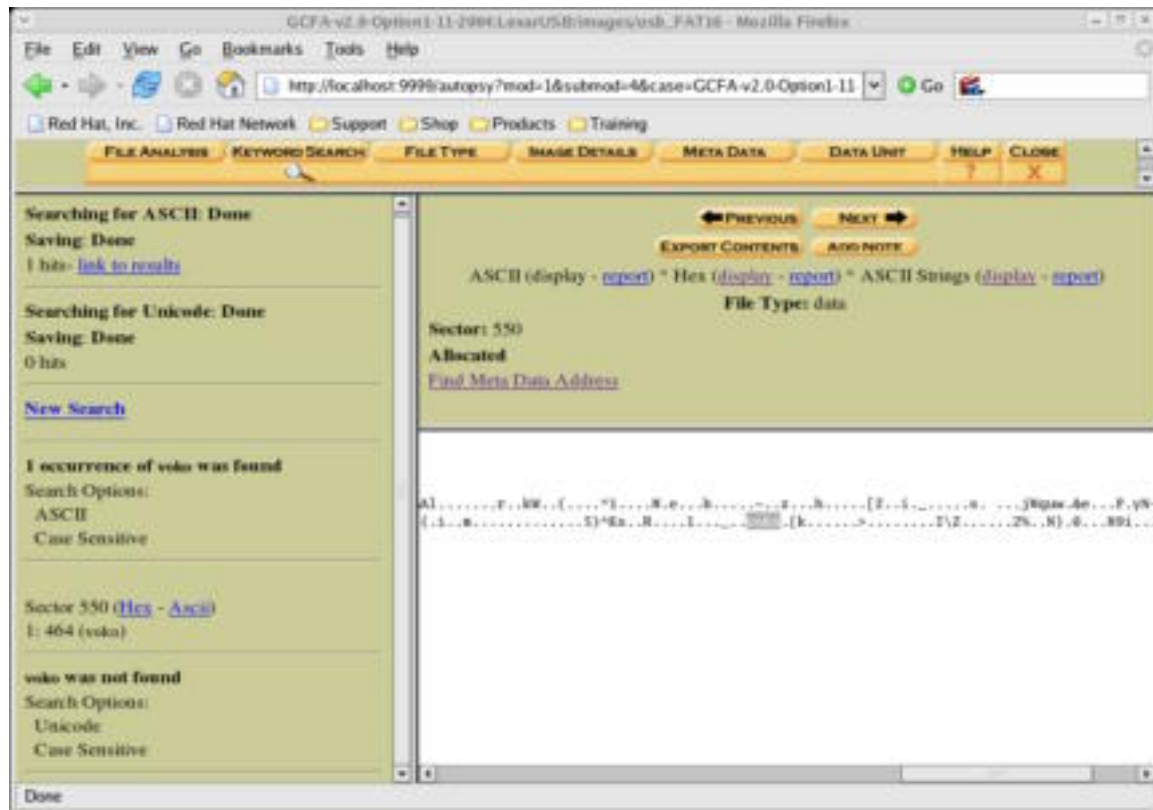


Figure 2.37: Autopsy string search for “voko” in the entire image

Selecting the “details” link in the Autopsy “Host Manager” menu it is possible to access the Autopsy capabilities to extract the strings of the entire image (see figure 2.18). Once the unallocated sectors have been obtained (see figure 2.19) then it is possible to extract<sup>56</sup> and look for the strings in both areas, the entire image and the unallocated space (see figure 2.38).

Due to the fact that all the evidence pieces have been already retrieved, providing all the case information required, instead of analyzing the whole list of strings corresponding to the entire image, we will only focus the search on the unallocated space using a very specific string from the Dirty Word List as an example, the keyword “Leila” (see figure 2.39).

The term “Leila” was found in sector 1795 of the unallocated space (sector 2426 of the original image<sup>57</sup>), that corresponds to the Directory Entry 15<sup>58</sup>, that is, the deleted “\_apture” file<sup>59</sup>.

It is also possible to perform string searches in libpcap traffic captured files using ethereal [ETHE1] using the “frame contains ...” display filter. For this case, the filter “frame contains “@hotmail”>” was used to find all packets including references to e-mail addresses associated to the mail service used by the victim. As a result, three TCP/HTTP packets were found, numbered 7, 16 and 24, all them part of the main TCP

<sup>56</sup>The performance of the searches is notably improved in this way.

<sup>57</sup>Obtained through the Autopsy “View original” link (see figure 2.39).

<sup>58</sup>Obtained through the Autopsy “Find Meta Data Address” link.

<sup>59</sup>This search would have found the e-mail contents in the Hotmail HTTP header in case it wouldn’t have been discovered while analyzing the “capture” file.

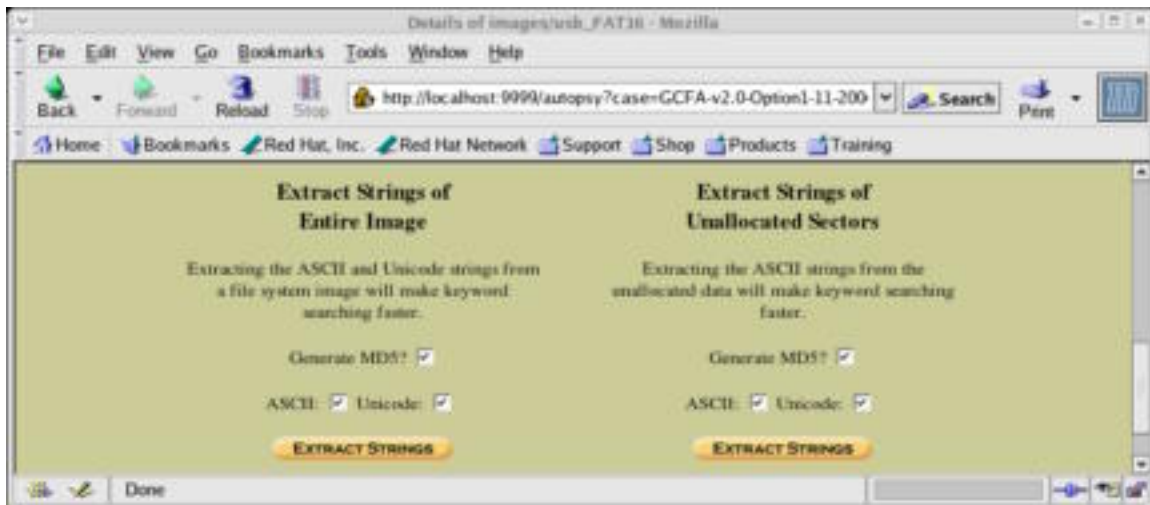


Figure 2.38: Autopsy string search capabilities

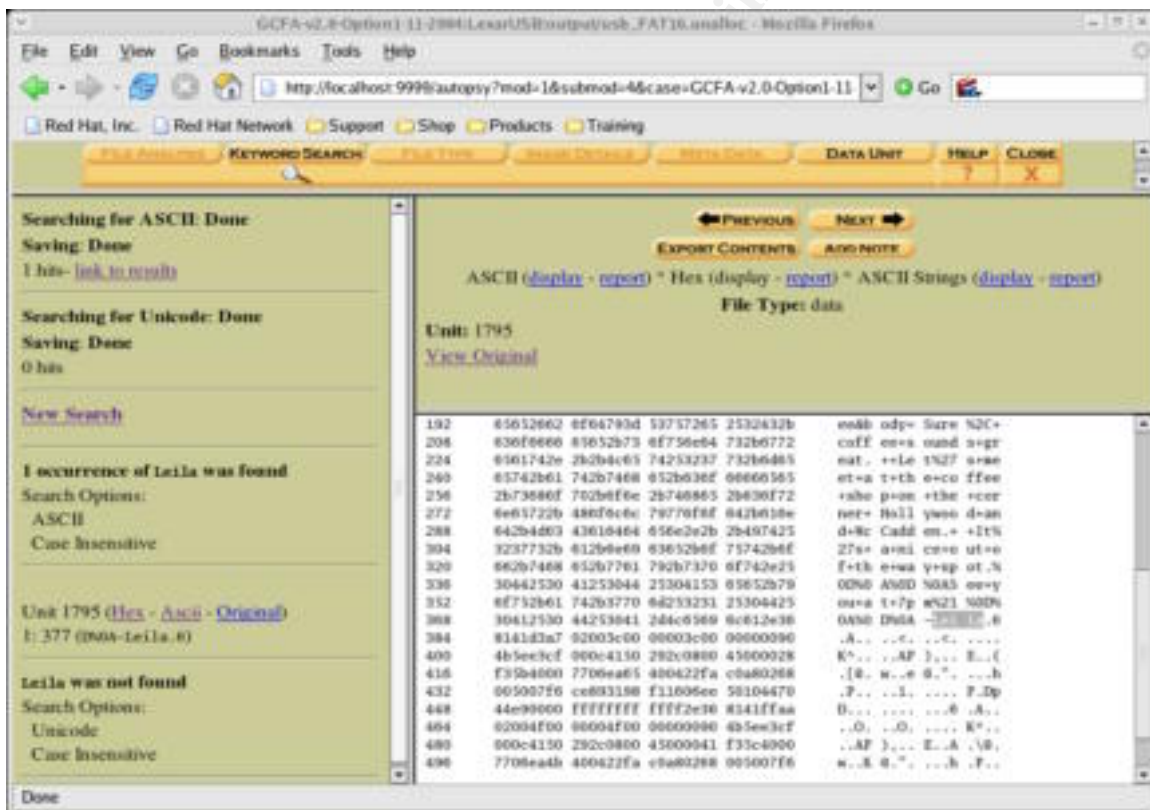


Figure 2.39: Autopsy string search results for “Leila” in the unallocated space

session described in a section 2.5.

## 2.7 Examination summary

*“What Mr. Lawrence did and how it relates to the concerns expressed by Ms. Conlay?”*

To sum up, the morning of Monday October 25th, Robert Lawrence created a file, called “her.doc” (see figure 2.24). This is the first of three files found that were supposedly sent as e-mails to Leila Conlay. The next morning, October 26th, the second file called “hey.doc” was created (see figure 2.25) using a more aggressive language <sup>60</sup>.

One day later, October 27th, the suspect obtained the WinDump/WinPcap sniffer that was used on Thursday, October 28th at 11:08, to capture the network traffic associated to a victim’s personal Hotmail session. The contents of the e-mail captured included a date with another person the same day in a coffee shop located on “Hollywood Blvd” and “McCadden” streets. Immediately after this, the suspect, having the previous information, obtained a map image of the coffee location (see figure 2.27).

The evening of October 28th, at 7pm, the victim met with her friend at the coffee shop and the suspect also was there. Fact that confirms why 25 minutes later he wrote a new message including his complaints about this meeting, “coffee.doc” (see figure 2.26), that would be sent as a third e-mail to the victim <sup>61</sup>.

This sequence of events matches the victim’s concerns made on the next afternoon, Friday October 29th, to CC Terminals corporate security, confirming that she was being harassed by the suspect, some of the numerous attempts the suspect made to meet her, the contents an nature (very aggressive) of the e-mails received and the scene at the coffee shop.

---

<sup>60</sup>The fact that these MS Word files have been sent as e-mails is based on the victim’s declaration. See chapter 7 for methods to confirm these facts.

<sup>61</sup>Due to the typical working hours, probably, the victim read it the next day.

# 3 IMAGE DETAILS

Table 3.1 shows all the files that are or were in the USB flashdrive image, including all their properties <sup>1</sup>. All this information has been obtained during the “Timeline” and “Media Analysis and Data Recovery” phases (see sections 2.3 and 2.4).

FAT filesystems don’t have any security mechanism to control the access (read, write, execute, delete...) to files and directories. For this reason the concepts of user, group and permissions don’t exist and this type of information cannot be extracted from this USB flashdrive.

The size, type and MD5 values (see figure 3.2, <sup>2</sup>) of the files obtained from the disk during the analysis can be found on figure 3.1. Based on the analysis, only the 0.098% of the partition space is currently occupied <sup>3</sup> and only the 1.674% was occupied anytime in

<sup>1</sup>“DE” means FAT **D**irectory **E**ntry.

<sup>2</sup>An screen shot has been taken because it is much more believable than typed text. A picture of the screen would also be valid [MURR1].

<sup>3</sup>Available data size: 121408 - 32 = 121376 sectors. Current files: 120 sectors (511-630).

Filename	True name	MAC times	Size (bytes)	# DE
her.doc	her.doc	C: Mon Oct 25 2004 08:32:06 M: Mon Oct 25 2004 08:32:08 A: Mon Oct 25 2004 00:00:00	19968	3
hey.doc	hey.doc	C: Mon Oct 26 2004 08:48:06 M: Mon Oct 26 2004 08:48:10 A: Mon Oct 26 2004 00:00:00	19968	4
._INPCA 1.EXE	WinPcap_3.1_beta.3.exe	C: Mon Oct 27 2004 16:23:54 M: Mon Oct 27 2004 16:23:56 A: Mon Oct 27 2004 00:00:00	19968	7
._INPCA 1.EXE	WinPcap_3.1_beta.3.exe	M: Mon Oct 27 2004 16:23:50 C: Mon Oct 27 2004 16:23:54 A: Mon Oct 28 2004 00:00:00	19968	10
._INDUMP.EXE	WinDump.exe	C: Mon Oct 27 2004 16:24:04 M: Mon Oct 27 2004 16:24:06 A: Mon Oct 27 2004 00:00:00	19968	12
._INDUMP.EXE	WinDump.exe	M: Mon Oct 27 2004 16:24:02 C: Mon Oct 27 2004 16:24:04 A: Mon Oct 28 2004 00:00:00	19968	14
._apture	capture	C: Mon Oct 28 2004 11:08:24 M: Mon Oct 28 2004 11:11:00 A: Mon Oct 28 2004 00:00:00	53056	15
._ap.gif	map.gif	C: Mon Oct 28 2004 11:17:44 M: Mon Oct 28 2004 11:17:46 A: Mon Oct 28 2004 00:00:00	8814	16
._ap.gif	map.gif	C: Mon Oct 28 2004 11:17:44 M: Mon Oct 28 2004 11:17:46 A: Mon Oct 28 2004 00:00:00	8814	17
coffee.doc	coffee.doc	C: Mon Oct 28 2004 19:24:46 M: Mon Oct 28 2004 19:24:48 A: Mon Oct 28 2004 00:00:00	19968	18

Table 3.1: Listing of all the files within the USB flashdrive



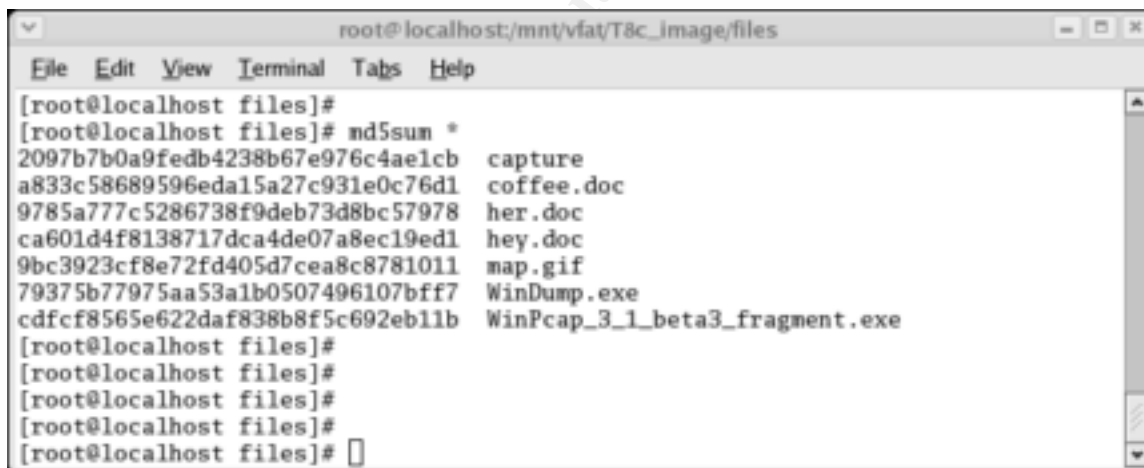
the partition life <sup>4</sup> (see table 2.3).

```
# ll
-r----- 1 root root 53056 Nov 27 13:03 capture
-r----- 1 root root 19968 Nov 27 12:48 coffee.doc
-r----- 1 root root 19968 Nov 27 12:50 her.doc
-r----- 1 root root 19968 Nov 27 12:51 hey.doc
-r----- 1 root root 8814 Nov 27 12:59 map.gif
-r----- 1 root root 450560 Nov 27 13:20 WinDump.exe
-r----- 1 root root 465408 Nov 28 09:52 WinPcap_3_1_beta3_fragment.exe

# file *
capture:                                tcpdump capture file (little-endian) - version 2.4
                                         (Ethernet, capture length 4096)
coffee.doc:                            Microsoft Office Document
her.doc:                                Microsoft Office Document
hey.doc:                                Microsoft Office Document
map.gif:                                GIF image data, version 89a, 300 x 200
WinDump.exe:                            MS-DOS executable (EXE), OS/2 or MS Windows
WinPcap_3_1_beta3_fragment.exe: data

# md5sum *
2097b7b0a9fedb4238b67e976c4ae1cb  capture
a833c58689596eda15a27c931e0c76d1  coffee.doc
9785a777c5286738f9deb73d8bc57978  her.doc
ca601d4f8138717dca4de07a8ec19ed1  hey.doc
9bc3923cf8e72fd405d7cea8c8781011  map.gif
79375b77975aa53a1b0507496107bff7  WinDump.exe
cdfcf8565e622daf838b8f5c692eb11b  WinPcap_3_1_beta3_fragment.exe
```

Figure 3.1: Size, type and MD5 value of all the image files



```
root@localhost:/mnt/vfat/T8c_image/files
File Edit View Terminal Tabs Help
[root@localhost files]#
[root@localhost files]# md5sum *
2097b7b0a9fedb4238b67e976c4ae1cb  capture
a833c58689596eda15a27c931e0c76d1  coffee.doc
9785a777c5286738f9deb73d8bc57978  her.doc
ca601d4f8138717dca4de07a8ec19ed1  hey.doc
9bc3923cf8e72fd405d7cea8c8781011  map.gif
79375b77975aa53a1b0507496107bff7  WinDump.exe
cdfcf8565e622daf838b8f5c692eb11b  WinPcap_3_1_beta3_fragment.exe
[root@localhost files]#
[root@localhost files]#
[root@localhost files]#
[root@localhost files]#
[root@localhost files]#
```

Figure 3.2: Screen shot of the MD5 values of all files recovered from the USB disk

### 3.1 Keywords associated with the programs/files

Apart from the string searching steps of section 2.6, in this case, the TSK's [SLEU1] sstrings (v1.72) command was used individually over each recovered file to extract its most relevant ASCII and Unicode (" -e 1") strings [INFOR16]. A summary list of the most relevant strings found is in table 3.2 (see figure 3.3 as an example).

<sup>4</sup>From sector 511 to 2542, that is 2032 sectors.

Filename	# DE	Strings
her.doc	3	Robert Lawrence, Normal.dot, Microsoft Word 10.0, Microsoft Word Document, MSWordDoc, Word.Document.8, Times New Roman, Symbol, Arial, Courier New... plus the document contents.
hey.doc	4	"Similar to "her.doc"
.INPCA 1.EXE	10	Portions of Windows registry keys: "ULM,EY_CLASSES.RO,URRENT_UA, LOCAL_MACHINE", lots of libraries referenced, such as OLEAUT32.DLL or KERNEL32.DLL, and functions, like GetProcAddress or RegCloseKey, and file references as "data/Main/12/NetMonInstaller.exe", PACKAGEINFO...
.INDUMP.EXE	14	!This program cannot be run in DOS mode., .text, .data, .rdata, Source code references, such as, "Header: /tcpdump/master/tcpdump/addrtoname.c ,v 1.96.2.6 2004/03/24 04:14:31 guy Exp", Copyright messages and other program messages, like "not enough space for stdio initialization", Microsoft Visual C++ Runtime Library, KERNEL32.dll, WSOCK32.dll, lots of "pcap" functions references, wpcap.dll, Windows functions, such as "GetTimeZoneInformation", protocols and technologies it manages, like FDDI or IPv6, and its fields and the program usage help message: Usage: %s [-aAdDeflLnNOpqRStuUvxX] [-B size] [-c count] [-C file_size]
.apture	15	SNMP Trap messages text, HTTP headers, HTTP responses, HTTP Cookies, HTML responses and GIF images... (see section 2.5) all from the network traffic sessions captured with WinDump.
.ap.gif	17	GIF89a ("The rest are strings from binary data")
coffee.doc	18	"Similar to "her.doc"

Table 3.2: Listing of the most relevant strings from the files found in the USB flashdrive

For the "her.doc" file, the strings provided the contents of the file (see figure 2.24) as well as the program and file template (.dot) used to create it, its title and the font types used. The same information is available for the other two ".doc" files. The GIF image only contains binary data apart from the header string type, while the capture file includes all the communication sessions analyzed in section 2.5. Finally, the two binary files contain strings associated to the libraries (".dll") and functions they use, the program messages and other information very useful to identify the type of binary they are, such as the fact that WinDump was compiled with Visual C++<sup>5</sup>.

When searching strings in binary files, such as "map.gif" or "WinDump.exe", it is recommended to filter the strings displayed based on the number of characters. Using the `sstrings -7` option, it will only display strings with 7 or more characters (instead of the default of 4 characters)<sup>6</sup>.

<sup>5</sup>This data corroborates the information found during the analysis and confirms the exact binary files found (see chapter 4 and 5).

<sup>6</sup>It is recommended to adjust this number based on the file nature, but a value between 7 and 10 is a good starting point.

```
# sstrings her.doc
bjbj
Hey I saw you the other day. I tried to say "hi", but you disappeared??? That
was a nice blue dress you were wearing. I heard that your car was giving you some \
trouble. Maybe I can give you a ride to work sometime, or maybe we can get dinner sometime?
Have a nice day
Hey I saw you the other day
Robert Lawrence
Normal.dot
Robert Lawrence
Microsoft Word 10.0
Hey I saw you the other day
Title
Microsoft Word Document
MSWordDoc
Word.Document.8
#
# sstrings -e l her.doc
Normal
Default Paragraph Font
Table Normal
No List
Unknown
Times New Roman
Symbol
Arial
Courier New
Hey I saw you the other day
Robert Lawrence
Robert Lawrence
Root Entry
1Table
WordDocument
SummaryInformation
DocumentSummaryInformation
CompObj
```

Figure 3.3: Strings (ASCII and Unicode) from "her.doc".



## 4 FORENSIC DETAILS

This chapter goal is the identification of the program used by Robert Lawrence, its type and usage. The conclusions and a complete description of how they were obtained have been detailed in chapter 2, mostly during the “Timeline” (when?, see section 2.3), “Media Analysis and Data Recovery” (what & type?, see section 2.4), and “Data Examination” (usage?, see section 2.5) phases <sup>1</sup>.

### 4.1 Program used by the suspect

Initially, based on the filenames found during the analysis, the main program used by Robert Lawrence was **WinDump** (<http://windump.polito.it>) <sup>2 3</sup>, a text-based network traffic capture and analysis tool, also known as a network sniffer <sup>4</sup>. This tool requires a traffic capture library to run properly, also found in the USB flashdrive, **WinPcap**, the Free Packet Capture Library for Windows (<http://winpcap.polito.it>), which is derived from the Unix libpcap library (also available in the tcpdump Web page).

This tool (plus the library) can be used to capture all the network traffic coming to or leaving the system when it is running, or all the traffic <sup>5</sup> crossing the network where the system it is running on is plugged to <sup>6</sup>. In this investigation, it was used by the suspect to capture the victim traffic, more specifically, a Hotmail session including a private e-mail message.

Based on the timeline analysis (see section 2.3) the last time the tool was executed was the 28th of October, 2004, at 11:08:24 <sup>7</sup>, when the network traffic “capture” file was last saved (found also in the USB flashdrive).

The files nature was also confirmed executing the binary file (plus its library) in a controlled environment (see next section), considering the details extracted in chapter 5.

### 4.2 How the program used works?

---

**From:** 12 Dec, 2004 (15:30) **To:** 13 Dec, 2004 (22:00)

---

To understand how the program works, a basic reverse engineering binary analysis was performed using the workstation detailed on appendix B. The first step taken was transferring the binary files, the program and the official library, using an USB flashdrive, to the “C:\GCFA” folder of the analysis workstation. Then, their MD5 values were verified using the Windows version of md5deep (v1.5) [MD5D1] (see figure 4.5).

---

<sup>1</sup>The details have not been included here, separately from the whole analysis process, trying not to repeat contents material and to improve the paper readability.

<sup>2</sup>What program?

<sup>3</sup>tcpdump (<http://www.tcpdump.org>) for Windows.

<sup>4</sup>Type of program?

<sup>5</sup>Running in promiscuous mode (by default, “-p” option).

<sup>6</sup>Usage?

<sup>7</sup>When it was used the last time?

Prior to executing the WinDump.exe binary, the following tools were started and ready to capture all the events occurring in the system (see figure 4.1):

- A VMWare snapshot (Snapshot - Save Snapshot) was taken in order to be able to easily revert the system to the current, “non-infected”, status.
- RegShot (v1.61e1) [REGS1] was run before executing the binary to get a current system snapshot (1st shot) of files and registry keys that could be compared later on, to check the changes introduced in the system once the binary has been executed.
- FileMon (v4.34) [FILEM1], RegMon (v4.34) [REGM1] and TDIMon (v1.0) [TDIM1] were executed to get the system activity related with files, the Windows registry and network connection respectively. The capture was stopped in the three tools (Ctrl + E) until launching the program execution.
- ProcExp, Process Explorer (v 5.23) [PROCE1], was launched to monitor the process activities in the system while the program is running.
- Additionally, tcpdump was executed in promiscuous mode in the Linux VM to monitor all the traffic generated by the tool on the lab virtual network: `tcpdump -nnvXe -s0 -w /tmp/lab_capture`.
- Finally, as the main tool purpose was known in advance (see chapter 5), in another terminal of the Linux VM, the command `ping -c 2 10.10.10.1` was ready to be executed and generate traffic in the network (not addressed to the Windows analysis VM, to test its promiscuous behaviour).

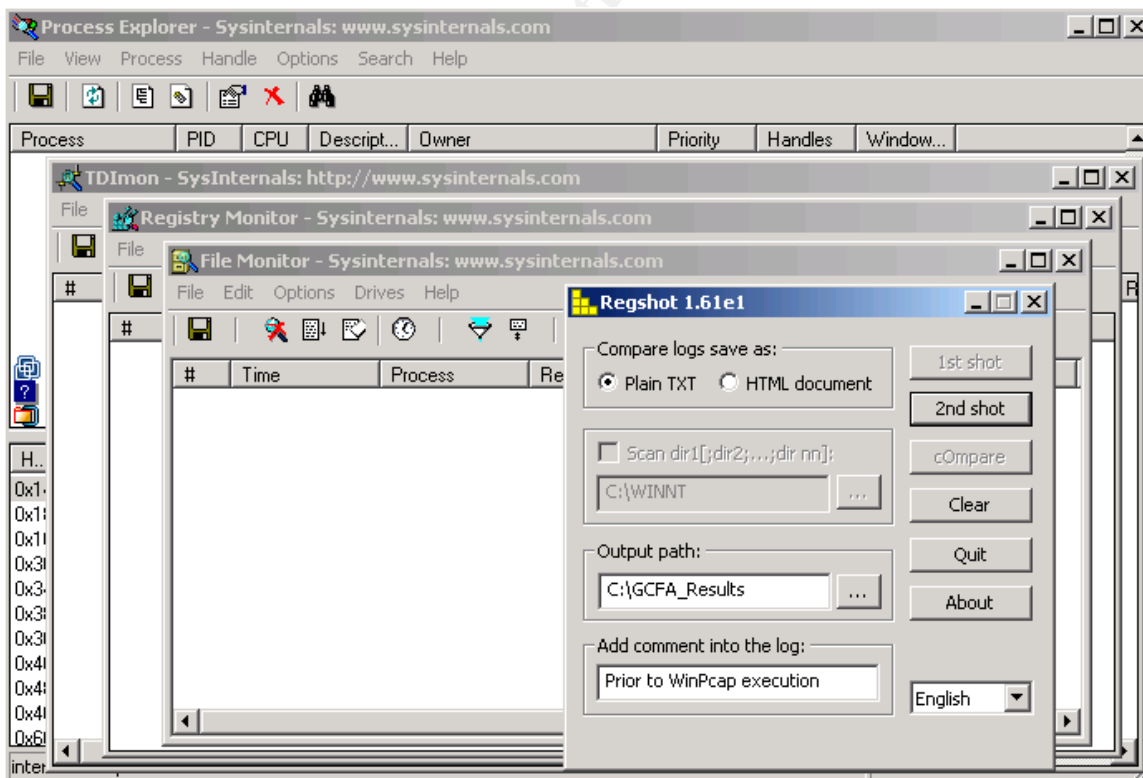


Figure 4.1: Windows binary analysis tools preparation

If the WinDump.exe program is executed without having installed the WinPcap library first, a Windows error message is generated indicating that the dynamic link library “wpcap.dll”

cannot be found. If it is run when a different WinPcap version is installed, such as WinPcap v3.0, then a Windows popup error message is generated: "The procedure entry point pcap\_dump\_flush could not be located in the dynamic link library wpcap.dll" <sup>8</sup>.

Therefore, as mentioned before, the WinPcap library is required for WinDump to run properly. The WinPcap fragment recovered from the disk in chapter 2 obviously cannot be executed. If the official WinPcap\_3\_1\_beta\_3.exe file is executed instead <sup>9</sup>, then the WinPcap Installation Wizard is launched and the library is installed in the system <sup>10</sup>.

The library default installation process <sup>11</sup> has been monitored using the tools and methods described for the WinDump program analysis explained below (see figure 4.1). The main conclusions obtained are:

- A new process associated to the library installation program, child of the "cmd.exe" process, was created, called "WinPcap\_3\_1\_beta\_3.exe". During the execution, a new child process was launched, "NetMonInstaller".
- At the file level, the installation used the "unpack.dll" and "Resume.exe" files, created in a temporary directory inside the user profile folder. Besides it created inside another temporary "WinPcap\_3\_1\_beta\_3" folder and saved there some files, such as "splash.bmp", "presetup.rgn", "presetup.bmp", "db.pdb", "main.pdb" and "Uninstall.exe"; and subfolders, like "presetup", "plugins" and "lng".
- Finally, it created a new folder, "C:\Program Files\WinPcap", with some files inside (see figure 4.3) and 4 new libraries in "C:\WINNT\system32" (%SystemRoot%), called "pthreadVC.dll", "wanpacket.dll", "packet.dll" and "wpcap.dll". Right clicking in them and selecting "Properties - Version" it is possible to check their descriptions ("WanPacket", "Packet" and "wpcap - Based on libpcap 0.8.1"), version (3.1.0.23) and company ("NetGroup - Politecnico di Torino"). The basic information of these new relevant files created was obtained (see figure 4.2), because these are the WinPcap itself <sup>12</sup>.
- It also installed a new Windows packet capture driver in "C:\WINNT\system32\drivers", called "npf.sys", "WinPcap Netgroup Packet Filter Driver", and used previous definitions files from "C:\WINNT\INF", such as "netnm.PNF", "layout.PNF" or "netrasa.PNF", and lots of Windows DLL libraries.
- This is a standard Windows installation process, where the digitally signed files hashes of the Windows WFP (Windows File Protection) catalog files (.cat) are also queried, in "C:\WINNT\system32\CatRoot".
- The generic Windows software logs and config files were modified too <sup>13</sup>: "software.LOG", "system" and "SYSTEM.ALT"; and "C:\WINNT\setupapi.log".
- No network connections were established during this process.
- Analyzing the registry, the most relevant branches, keys and values modified are in figure 4.4, such as the information required to uninstall the WinPcap library. It also

<sup>8</sup>It was verified installing a previous WinPcap version in a "clean" VMWare snapshot.

<sup>9</sup>Based on the identification performed in chapter 5.

<sup>10</sup>If a previous WinPcap version is detected, the system must be rebooted so the changes will take effect.

<sup>11</sup>The default installation was followed, pressing "Next" and accepting the terms of the license.

<sup>12</sup>The timestamp of these files matches the official release dates of this WinPcap version, mentioned in chapter 5, one day before May 15th.

<sup>13</sup>From "C:\WINNT\system32\config".

created several others related with the current system configuration <sup>14</sup> <sup>15</sup>, like the new network driver (NDI, Network Driver Installer <sup>16</sup>) and its definitions.

```

C:\WINNT\system32>md5deep wanpacket.dll packet.dll wpcap.dll
408e8ef3c17501144f1adcd2ec11b001 C:\WINNT\system32\wanpacket.dll
28f57308b067836d715aa070d29c3232 C:\WINNT\system32\packet.dll
5c5561185a8751711156934585f002e8 C:\WINNT\system32\wpcap.dll

C:\WINNT\system32>dir wanpacket.dll packet.dll wpcap.dll
Volume in drive C has no label.
Volume Serial Number is A493-6352

Directory of C:\WINNT\system32

05/14/2004  11:30a                61,440 wanpacket.dll

Directory of C:\WINNT\system32

05/14/2004  11:30a                81,920 packet.dll

Directory of C:\WINNT\system32

05/14/2004  01:02p                225,280 wpcap.dll
               3 File(s)              368,640 bytes
               0 Dir(s)              913,129,472 bytes free

C:\WINNT\system32>

```

Figure 4.2: Library (.dll) files created by WinPcap

```

C:\Program Files\WinPcap>dir
Volume in drive C has no label.
Volume Serial Number is A493-6352

Directory of C:\Program Files\WinPcap

12/13/2004  06:04a    <DIR>          .
12/13/2004  06:04a    <DIR>          ..
05/14/2004  11:36a           49,152 daemon_mgm.exe
12/13/2004  06:04a           5,105 INSTALL.LOG
05/14/2004  11:38a           6,656 NetMonInstaller.exe
05/14/2004  11:37a           49,152 npf_mgm.exe
05/14/2004  01:02p           86,016 rpcapd.exe
08/30/2003  05:50p          199,168 Uninstall.exe
               6 File(s)              395,249 bytes
               2 Dir(s)              913,129,472 bytes free

C:\Program Files\WinPcap>md5deep *
5df2055815aa72ac84e0fe4466f8b295 C:\Program Files\WinPcap\daemon_mgm.exe
4955bed34075b392749bbeb0c884ccc C:\Program Files\WinPcap\INSTALL.LOG
87c1716cf63a2522e8d1fc123a1dd9df C:\Program Files\WinPcap\NetMonInstaller.exe
07382671a64e2b63638aa8ea93390c82 C:\Program Files\WinPcap\npf_mgm.exe
93322bf342adbb522dfa7bc95f5e167c C:\Program Files\WinPcap\rpcapd.exe
62da2c201bc09a55c97c46f0ad73c28a C:\Program Files\WinPcap\Uninstall.exe

C:\Program Files\WinPcap>

```

Figure 4.3: Program files created by WinPcap

Once the library has been installed and the analysis lab was ready, WinDump was executed without options from a new MS-DOS window (see figure 4.5), then, the network

<sup>14</sup>The meaning of some of the keys and files has been mainly obtained through personal experience or searching in Google.

<sup>15</sup>HKEY\_LOCAL\_MACHINE\SYSTEM\CurrentControlSet\Services and HKEY\_LOCAL\_MACHINE\SYSTEM\ControlSet001\Services.

<sup>16</sup>Lots of keys containing the branch “Ndi” and others, like “MS\_NDISWANBH” and “MS\_NDISWANIP”.

```

HKEY_LOCAL_MACHINE\SYSTEM\ControlSet001\Services\nm
HKEY_LOCAL_MACHINE\SYSTEM\ControlSet001\Services\NPF
HKEY_LOCAL_MACHINE\SYSTEM\ControlSet001\Services\rpcapd
HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Services\nm
HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Services\NPF
HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Services\rpcapd
...
HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Windows\CurrentVersion\Uninstall\WinPcapInst
HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Windows\CurrentVersion\SharedDLLs\C:\Program Files\WinPcap\
Uninstall.exe: 0x00000001
HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Windows\CurrentVersion\Uninstall\WinPcapInst\DisplayName:
"WinPcap 3.1 beta3"
HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Windows\CurrentVersion\Uninstall\WinPcapInst\UninstallString:
""C:\Program Files\WinPcap\Uninstall.exe" "C:\Program Files\WinPcap\install.log""
HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Windows\CurrentVersion\Uninstall\WinPcapInst\InstallLocation:
"C:\Program Files\WinPcap"
HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Windows\CurrentVersion\Uninstall\WinPcapInst\InstallSource:
"C:\GCFA"
HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Windows\CurrentVersion\Uninstall\WinPcapInst\InstallSourceFile:
"C:\GCFA\WinPcap_3_1_beta_3.exe"
HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Windows\CurrentVersion\Uninstall\WinPcapInst\InstallDate:
"12/13/2004"
HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Windows\CurrentVersion\Uninstall\WinPcapInst\Publisher:
"Politecnico di Torino"

```

Figure 4.4: Main registry branches and keys used by WinPcap

```

C:\GCFA>md5deep *
79375b77975aa53a1b0507496107bfff? C:\GCFA\WinDump.exe
4511ee3b4e5d8150c035a140dfba72c0 C:\GCFA\WinPcap_3_1_beta_3.exe

C:\GCFA>WinPcap_3_1_beta_3.exe

C:\GCFA>WinDump.exe
WinDump.exe: listening on \Device\NPF_GenericNdisWanAdapter
08:29:48.236625 04:d3:20:52:41:53 802.1b-gsap > 03:00:00:00:00:02 802.1b-isap ui
/C len=180

1 packets captured
1 packets received by filter
0 packets dropped by kernel

C:\GCFA>WinDump.exe -i 2 -nn
WinDump.exe: listening on \Device\NPF_{5A2B252B-0A24-4014-A92A-1896B52A418D}
08:30:10.608462 IP 10.10.10.10 > 10.10.10.1: icmp 64: echo request seq 0
08:30:10.608513 IP 10.10.10.1 > 10.10.10.10: icmp 64: echo reply seq 0
08:30:11.642182 IP 10.10.10.10 > 10.10.10.1: icmp 64: echo request seq 1
08:30:11.642194 IP 10.10.10.1 > 10.10.10.10: icmp 64: echo reply seq 1

4 packets captured
4 packets received by filter
0 packets dropped by kernel

C:\GCFA>_

```

Figure 4.5: MD5 values and initial execution of the program

interface the traffic will be captured from is displayed, and it started capturing all the network traffic (by default it works in promiscuous mode) from the default network interface and showed it through the program standard output (the MS-DOS window itself) <sup>17</sup>.

After analyzing the status of the ProcExp tool, the binary was left running for about 30 more seconds and its execution was aborted pressing Ctrl + C in the MS-DOS Windows. Then, the capture of all three monitoring tools was stopped (Ctrl + E) as well as the Linux

<sup>17</sup>In this case it capture one layer 2, 802.1b packet, from the Windows generic adapter.



tcpdump traffic capturing tool. Finally, a new snapshot (2nd shot) was taken using RegShot and its “cOmpare” function invoked to check the system status.

The result obtained was not very useful due to the fact that the default interface it is not the real one. The system network interfaces list can be obtained running WinDump.exe -D. The default interface is the first one, numbered with “1”. Another interface can be specified using the “-i” option. Therefore, the analysis process was repeated specifying the valid network adapter, number 2 (see figure 4.5). Additionally, the “-nn” flag was set to avoid name and port resolution.

This time, the ready to use ping command was executed, confirming the promiscuous interception of traffic (see figure 4.5). If any traffic is generated over the network the tool is listening on, the traffic will be immediately captured by it.

The conclusions obtained from the program analysis are:

- A new process was created, child of the “cmd.exe” process, called “WinDump.exe”.
- The program uses several registry keys (see figure 4.7 for the generic branches accessed) to access the network driver (npf.sys) and other network and TCP/IP parameters and configurations in the current system.
- Through TDImon it is possible to visualize the IOCTL operations performed by the program over the network interface (see figure 4.6) in order to capture the traffic.
- Finally, the program uses lots of standard Windows networking libraries (.dll) from “C:\WINNT\system32”, such as “WSOCK32”, “WS2\_32”, “WS2HELP”, “NPPTools”, “ICMP”, “NETAPI32”, “SECUR32”, “NETTRAP”, “DNSAPI”.... The most important accesses, related with WinPcap, are “wpcap.dll”, “wanpacket.dll” and “packet.dll”, as well as the “ntp.sys” capture driver previously mentioned. All them are required to interact with the network adapters, capture the traffic and interpret the different communication protocols.

#	Time	Process	Object	Request	Local	R.	Result	Other
1	0.000...	WinDump.exe:1...	81126858	IRP_MJ_CREATE	TCP:Control obj		SUCCESS	
2	0.006...	WinDump.exe:1...	810D25F8	IRP_MJ_CREATE	TCP:Control obj		SUCCESS	
3	0.006...	WinDump.exe:1...	81126858	IRP_MJ_DEVICE_CONTROL	TCP:Control obj		SUCCESS	IOCTL_TCP_QUERY_INFORMATION_EX
4	0.007...	WinDump.exe:1...	81126858	IRP_MJ_DEVICE_CONTROL	TCP:Control obj		SUCCESS	IOCTL_TCP_QUERY_INFORMATION_EX
5	0.007...	WinDump.exe:1...	81126858	IRP_MJ_DEVICE_CONTROL	TCP:Control obj		SUCCESS	IOCTL_TCP_QUERY_INFORMATION_EX
6	0.007...	WinDump.exe:1...	81126858	IRP_MJ_DEVICE_CONTROL	TCP:Control obj		SUCCESS	IOCTL_TCP_QUERY_INFORMATION_EX
7	0.007...	WinDump.exe:1...	81126858	IRP_MJ_DEVICE_CONTROL	TCP:Control obj		SUCCESS	IOCTL_TCP_QUERY_INFORMATION_EX
8	0.007...	WinDump.exe:1...	81126858	IRP_MJ_DEVICE_CONTROL	TCP:Control obj		SUCCESS	IOCTL_TCP_QUERY_INFORMATION_EX
9	0.660...	WinDump.exe:1...	81126858	IRP_MJ_DEVICE_CONTROL	TCP:Control obj		SUCCESS	IOCTL_TCP_QUERY_INFORMATION_EX
10	0.666...	WinDump.exe:1...	81126858	IRP_MJ_DEVICE_CONTROL	TCP:Control obj		SUCCESS	IOCTL_TCP_QUERY_INFORMATION_EX

Figure 4.6: TDImon IOCTL operations performed by WinDump

This analysis process was repeated several times, checking for the program behaviour when different command line options were used. A brief list of available options can be obtained through WinDump -h. The whole list of command line options to control its execution is available from its homepage <sup>18</sup>.

In order to save the traffic captured to a file, as in the case of this investigation, it must be executed using the command “WinDump.exe -w capture”. If WinDump has been directly executed from the USB flashdrive, the capture file, “capture” will be directly saved

<sup>18</sup><http://windump.polito.it/docs/manual.htm>



```
HKEY_LOCAL_MACHINE\SYSTEM\ControlSet001\Enum\Root\LEGACY_NM\  
HKEY_LOCAL_MACHINE\SYSTEM\ControlSet001\Enum\Root\LEGACY_NPF\  
HKEY_LOCAL_MACHINE\SYSTEM\ControlSet001\Services\nm\  
HKEY_LOCAL_MACHINE\SYSTEM\ControlSet001\Services\NPF\  
HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Enum\Root\LEGACY_NM\  
HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Enum\Root\LEGACY_NPF\  
HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Services\nm\  
HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Services\NPF\  
...  
HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Services\Tcpip\Parameters  
HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Services\Tcpip\Linkage  
HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Services\WinSock2\Parameters  
HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Control\ComputerName  
HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Tracing  
HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Cryptography
```

Figure 4.7: Main registry branches and keys used by WinDump

also in the USB flashdrive; it is saved in the directory from which the program has been executed.

Due to the fact that, nor the library neither the program, generate any traffic themselves, no other complex actions, such as service simulation using the Linux VM were performed.

© SANS Institute 2005, Author retains full rights.

# 5 PROGRAM IDENTIFICATION

---

**From:** 27 Nov, 2004 (22:00) **To:** 28 Nov, 2004 (14:30)

---

This section goal is the verification of the main program used by the suspect through reliable binary/data fingerprinting methods; this would help to demonstrate its purpose and usage detailed in chapter 4.

The WinDump homepage is <http://windump.polito.it> and the WinPcap homepage is <http://winpcap.polito.it>. At the time this forensic analysis was performed the latest downloadable versions was WinDump 3.8.3 beta <sup>1</sup>, available since the 3rd of May, 2004 <sup>2</sup> and WinPcap 3.1 beta 4 <sup>3</sup>, available since the 4th of November, 2004. Its evident that the later was not the version used during the incident (dated the 28th of October, 2004); however, WinPcap 3.1 beta 3 was released the 15th of May, 2004 (and beta 2, the 3rd of May, 2004), so both are potential candidates to be the library used.

Based on the forensic analysis filenames, the WinPcap version used by Robert Lawrence was 3.1 beta 3 but the exact WinDump version is unknown. Trying to guess it, it must be considered that the previous WinDump main version branch, 3.6.x, only worked with the 2.3 versions of WinPcap, therefore and based on the WinPcap version, it seems the WinDump version used was from the 3.8.x branch. Explicitly the following message is in the WinDump homepage:

*“BEFORE running WinDump 3.8.3 beta you have to download and install WinPcap 3.1 beta2 or beta3.”*

The evolution of all the WinDump and WinPcap versions can be analyzed through their change logs: <sup>4</sup> and <sup>5</sup>. The WinPcap version mentioned was not available in the WinDump/WinPcap download repository, <http://windump.polito.it/misc/bin/> or in the mirror repositories <http://winpcap.polito.it/misc/mirrors.htm> at the time of this writing.

The WinPcap version used in the incident was found searching through Google <sup>6</sup> the 28th of November, 2004, using the term “WinPcap\_3.1\_beta.3.exe” (see figure 5.1, 4th entry). I found a wget mirror log at <http://ftp.cs.pu.edu.tw/Windows/Security/mirror.log> with the “official” reference of the searched file <sup>7</sup>. The same log shows the reference to the 3.1 beta 3 source code package <sup>8</sup>.

All these files were downloaded to perform a deepest file and version analysis. As the first step, it was confirmed that the file sizes of the WinDump 3.8.3 beta (450560 bytes) and the WinPcap 3.1 beta 3 (485810 bytes) match with the ones found during the USB flashdrive in the timeline analysis (see figure 5.2 and appendix C).

To be able to compare the MD5 values of the USB evidence files and the official files a copy of the deleted files from the USB flashdrive must be restored (see section 2.4.3 and section 2.4.4). The “WinDump\_3.8.3beta.exe” is the file downloaded from the WinDump

---

<sup>1</sup><http://windump.polito.it/install/default.htm>

<sup>2</sup><http://windump.polito.it/news.htm>

<sup>3</sup><http://winpcap.polito.it/install/default.htm>

<sup>4</sup><http://windump.polito.it/misc/changelog.htm>

<sup>5</sup><http://winpcap.polito.it/misc/changelog.htm>

<sup>6</sup><http://www.google.com>

<sup>7</sup>[http://winpcap.polito.it/install/bin/WinPcap\\_3\\_1\\_beta\\_3.exe](http://winpcap.polito.it/install/bin/WinPcap_3_1_beta_3.exe)

<sup>8</sup>[http://winpcap.polito.it/install/bin/wpcapsrc\\_3\\_1\\_beta\\_3.zip](http://winpcap.polito.it/install/bin/wpcapsrc_3_1_beta_3.zip)



```
# ll WinDump_3.8.3beta.exe
-r----- 1 root root 450560 Nov 28 00:38 WinDump_3.8.3beta.exe
# ll /T8c/files/WinDump.exe
-r----- 1 root root 450560 Nov 27 13:20 /T8c/files/WinDump.exe
#
# md5sum WinDump_3.8.3beta.exe
79375b77975aa53a1b0507496107bff7 WinDump_3.8.3beta.exe
# md5sum /T8c/files/WinDump.exe
79375b77975aa53a1b0507496107bff7 /T8c/files/WinDump.exe
```

Figure 5.3: MD5 comparison of the official and recovered WinDump file

last 909 blocks of 512 bytes were extracted from the WinPcap file downloaded from the WinPcap official Web page, “WinPcap\_3\_1\_beta\_3.exe”. This new official fragment file was called “WinPcap\_3\_1\_beta\_3\_fragment\_official.exe” (see figure 5.4).

```
# dd if=WinPcap_3_1_beta_3.exe of=WinPcap_3_1_beta_3_fragment_official.exe \
    bs=512 skip=40
908+1 records in
908+1 records out
#
# ll /T8c/files/WinPcap_3_1_beta3_fragment.exe
-r----- 1 root root 465408 Nov 28 09:52 /T8c/files/WinPcap_3_1_beta3_fragment.exe
# ll WinPcap_3_1_beta_3_fragment_official.exe
-r----- 1 root root 465330 Nov 28 10:07 WinPcap_3_1_beta_3_fragment_official.exe
```

Figure 5.4: MD5 comparison of the official and recovered WinPcap file fragment

The official WinPcap file is 485810 bytes in length. If the first 40 blocks of 512 bytes are removed (20480 bytes), we get 465330 bytes, that is, 908 blocks (464896) plus 434 bytes. However, when the USB flashdrive version was recovered (in section 2.4.4) we used Autopsy working at the disk level, so the whole last sector was recovered (including the additional 78 bytes) for a total of 465408 bytes. These bytes were filled with zero and it can be verified using the hexedit editor (see figure 5.5). That's the reason why there is a difference of 78 bytes between both files.

```
...
00071998  00000000 14EC0700 00000000 BADF0C00 00000000 47495045 4E440000 ...GIPEND..
000719B4  00000000 00000000 00000000 00000000 00000000 00000000 00000000 .....
000719D0  00000000 00000000 00000000 00000000 00000000 00000000 00000000 .....
000719EC  00000000 00000000 00000000 00000000 00000000 00000000 00000000 .....
```

Figure 5.5: Last bytes of the last WinPcap file sector

Therefore, the fragment obtained from the official file must be filled up with 78 bytes containing the 0x00 value. To do so the “conv=sync” dd command option was used. Once filled, their size will be the same and its MD5 value could be verified (see figure 5.6).

In this case, the MD5 value of the latest 465408 bytes of the WinPcap library match, fact that allows identifying in a reliable way that WinPcap version 3.1 beta 3 was the version used by the suspect.

## 5.1 Compiling the WinPcap and WinDump sources

---

**From:** 12 Dec, 2004 (15:30) **To:** 16 Dec, 2004 (22:00)

---

```
# mv WinPcap_3_1_beta_3_fragment_official.exe \
    WinPcap_3_1_beta_3_fragment_official_truncated.exe
# dd if=WinPcap_3_1_beta_3.exe of=WinPcap_3_1_beta_3_fragment_official.exe \
    bs=512 skip=40 count=909 conv=sync
908+1 records in
909+0 records out
# ll WinPcap_3_1_beta_3_fragment_official*
-r----- 1 root root 465330 Nov 28 10:07 \
    WinPcap_3_1_beta_3_fragment_official_truncated.exe
-r----- 1 root root 465408 Nov 28 10:28 WinPcap_3_1_beta_3_fragment_official.exe
# ll /T8c/files/WinPcap_3_1_beta3_fragment.exe
-r----- 1 root root 465408 Nov 28 09:52 /T8c/files/WinPcap_3_1_beta3_fragment.exe
#
# md5sum /T8c/files/WinPcap_3_1_beta3_fragment.exe
cdfcf8565e622daf838b8f5c692eb11b /T8c/files/WinPcap_3_1_beta3_fragment.exe
# md5sum WinPcap_3_1_beta_3_fragment_official.exe
cdfcf8565e622daf838b8f5c692eb11b WinPcap_3_1_beta_3_fragment_official.exe
```

Figure 5.6: Verification of the MD5 value of WinPcap

In case it wouldn't be possible to arrive to the previous conclusive results, it would have been required to use other methods, such as the compilation and detailed analysis of the binaries contents <sup>9</sup>.

There are two options available <sup>10</sup> to compile these packages, using Microsoft Visual C++ 6.0 <sup>11</sup> plus the Microsoft Software Development Kit (SDK) <sup>12</sup> or using Cygwin <sup>13 14</sup>. The source code download and Cygwin compilation process is available in appendix G.

As expected, the files didn't match (even in size) because there are multiple dependencies, external to the sources themselves, that can affect the end result, such as the compiler used (MS Visual C++ or Cygwin) and its specific version, or its complementary tools and versions, such as the linker (ld) or other parsing tools, like bison and flex in Cygwin, or the DDK <sup>15</sup> and SDK <sup>16</sup> in Visual C++. Besides, based on the documentation of WinPcap <sup>17</sup>, for the Visual C++ project, there are six different built configurations <sup>18</sup>. Based on the WinDump documentation <sup>19</sup>, there are two versions of the tool, "release" and "debug". There are also other OS library dependencies (see appendix G).

The main reason confirming the difference observed is that the file used by the suspect, that is, the official WinDump executable, has been compiled with MS Visual C++. Inspecting the source code the MS Visual C++ project result is a file called "WinDump.exe", while the result of the Cygwin project is a file called "windump.exe"; note the difference between upper and lowercase. This is also confirmed in figure 5.7 due to the linker versions.

Both WinDump files were compared using LordPE RoyalTS <sup>20</sup> (PE Editor - Compare), which allows to see the huge effects the different compilers have over the same source

<sup>9</sup>As requested by the GCFA certification assignment v2.0.

<sup>10</sup>Based on the documentation files inside both source code packages.

<sup>11</sup><http://msdn.microsoft.com/visualc/>

<sup>12</sup><http://www.microsoft.com/msdownload/platformsdk/sdkupdate/downlevel.htm>

<sup>13</sup><http://www.cygwin.com>

<sup>14</sup>This was the option selected because the forensic team didn't have a copy of the commercial MS Visual C++ compiler for this investigation.

<sup>15</sup>Microsoft Driver Developer Kit (DDK).

<sup>16</sup>Microsoft Software Development Kit (SDK).

<sup>17</sup>The "readme-visualc" and "readme-cygwin" files.

<sup>18</sup>Besides, WinPcap can also be compiled for Win9x or WinNT (see appendix G).

<sup>19</sup>The "Readme.Win32" file.

<sup>20</sup><http://mitglied.lycos.de/yoda2k/LordPE/info.htm>



code (see figure 5.7, where the different linker versions are showed, 6.0 for Visual C++ (6.0) and 2.38 for Cygwin). They even generate different binary sections: Visual C++ uses 3 sections (.text, .rdata and .data) while Cygwin uses 5 (.text, .data, .rdata, .bss and .idata).

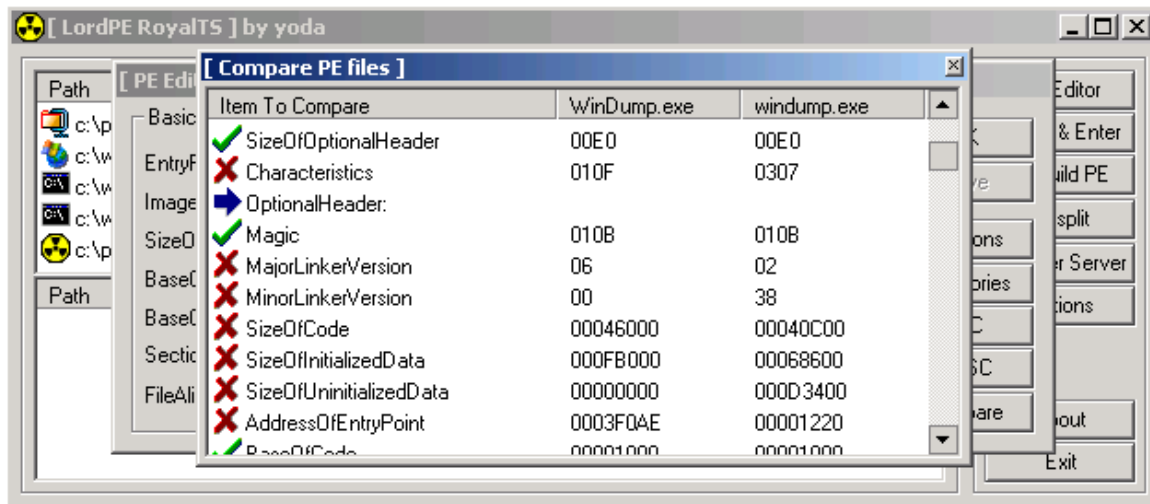


Figure 5.7: Comparison of both WinDump binaries through LordPE

Therefore, other methods different from the MD5 comparison must be used to compare the files, mainly based on the reverse engineering inspection methods of chapter 4, such as extracting the files human readable strings, the libraries they use and, if required <sup>21</sup>, a complete behavioral and code base analysis. The efforts will be only focused on the WinDump tool, due to the fact the WinPcap library was not entirely recovered from the flashdrive.



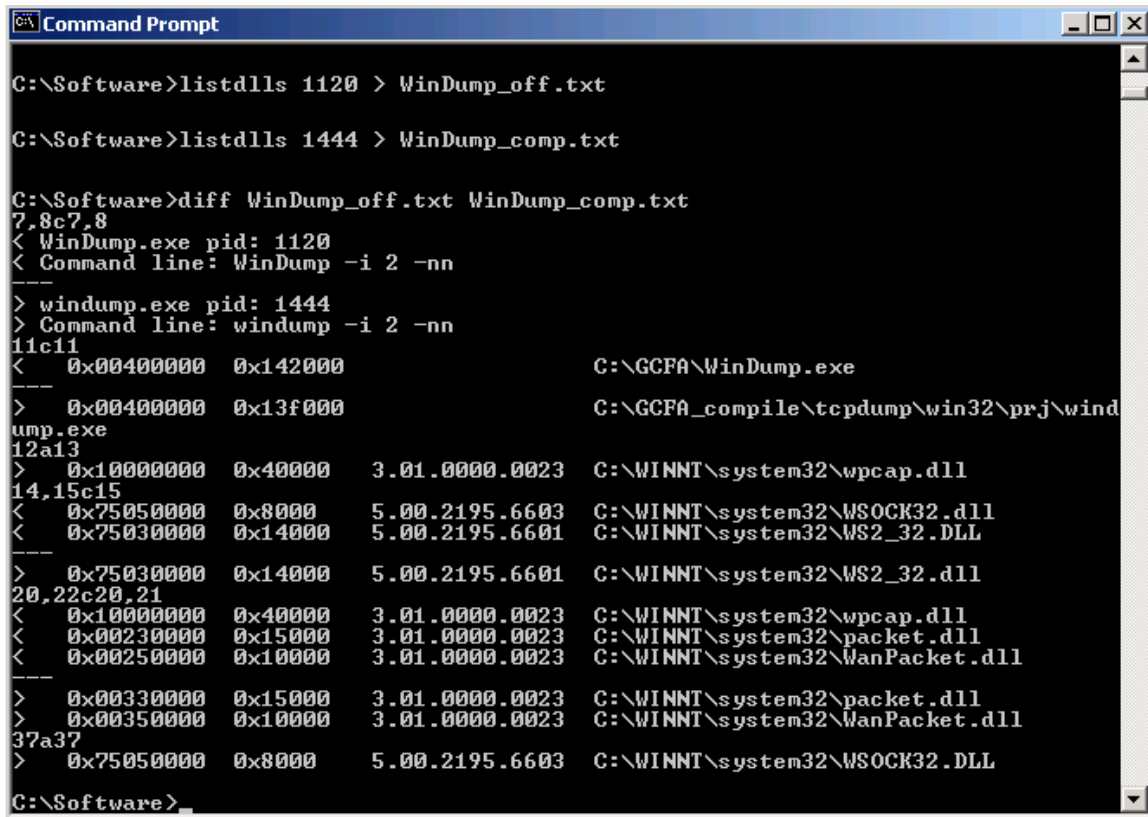
Figure 5.8: Comparison of the strings of both WinDump binaries

The strings of both files were compared using Bintext (v3.00) [BIN1], and most of them matched although in different file offsets (see figure 5.8 for the PCAP strings section). Additionally, the tools were run and their dynamic libraries (DLLs) were obtained using

<sup>21</sup>A more accurate but elaborate way of verifying the program is testing its functionality



ListDLLs (v.2.23) [LIST1]<sup>22</sup>. Then, the DLLs of the compiled and the recovered versions were compared, and again, they use exactly the same DLLs except that the PCAP libraries were loaded in a different memory offset (see figure 5.9).



```

C:\Software>listdlls 1120 > WinDump_off.txt

C:\Software>listdlls 1444 > WinDump_comp.txt

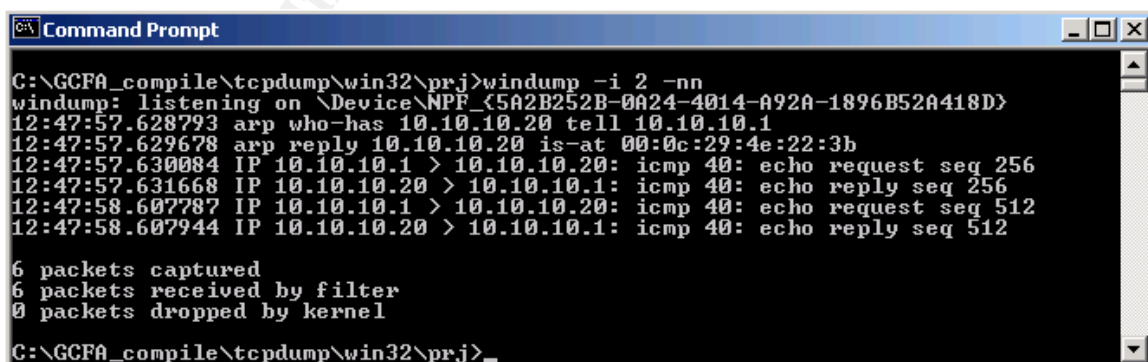
C:\Software>diff WinDump_off.txt WinDump_comp.txt
7,8c7,8
< WinDump.exe pid: 1120
< Command line: WinDump -i 2 -nn
---
> windump.exe pid: 1444
> Command line: windump -i 2 -nn
11c11
< 0x00400000 0x142000 C:\GCFA\WinDump.exe
---
> 0x00400000 0x13f000 C:\GCFA_compile\tcpdump\win32\prj\wind
ump.exe
12a13
> 0x10000000 0x40000 3.01.0000.0023 C:\WINNT\system32\wpcap.dll
14,15c15
< 0x75050000 0x8000 5.00.2195.6603 C:\WINNT\system32\WSOCK32.dll
< 0x75030000 0x14000 5.00.2195.6601 C:\WINNT\system32\WS2_32.DLL
---
> 0x75030000 0x14000 5.00.2195.6601 C:\WINNT\system32\WS2_32.dll
20,22c20,21
< 0x10000000 0x40000 3.01.0000.0023 C:\WINNT\system32\wpcap.dll
< 0x00230000 0x15000 3.01.0000.0023 C:\WINNT\system32\packet.dll
< 0x00250000 0x10000 3.01.0000.0023 C:\WINNT\system32\WanPacket.dll
---
> 0x00330000 0x15000 3.01.0000.0023 C:\WINNT\system32\packet.dll
> 0x00350000 0x10000 3.01.0000.0023 C:\WINNT\system32\WanPacket.dll
37a37
> 0x75050000 0x8000 5.00.2195.6603 C:\WINNT\system32\WSOCK32.DLL

C:\Software>

```

Figure 5.9: Comparison of the DLLs used by both WinDump binaries

The new compiled program was executed in a controlled environment and its analysis was performed (like in chapter 4), confirming that its output and behaviour is equal to the one extracted from the USB flashdrive (see figure 5.10)<sup>23</sup>.



```

C:\GCFA_compile\tcpdump\win32\prj>windump -i 2 -nn
windump: listening on \Device\NPF_{5A2B252B-0A24-4014-A92A-1896B52A418D}
12:47:57.628793 arp who-has 10.10.10.20 tell 10.10.10.1
12:47:57.629678 arp reply 10.10.10.20 is-at 00:0c:29:4e:22:3b
12:47:57.630084 IP 10.10.10.1 > 10.10.10.20: icmp 40: echo request seq 256
12:47:57.631668 IP 10.10.10.20 > 10.10.10.1: icmp 40: echo reply seq 256
12:47:58.607787 IP 10.10.10.1 > 10.10.10.20: icmp 40: echo request seq 512
12:47:58.607944 IP 10.10.10.20 > 10.10.10.1: icmp 40: echo reply seq 512

6 packets captured
6 packets received by filter
0 packets dropped by kernel

C:\GCFA_compile\tcpdump\win32\prj>

```

Figure 5.10: Execution of the WinDump compiled version

<sup>22</sup>To run this tool it is required to know the PID number of the WinDump process, extracted through the Windows Task Manager.

<sup>23</sup>This analysis has not been included due to GCFA certification paper length constraints.

All these steps allow to confirm reliably that both files have a similar functionality, but not to a 100%, as with the initial MD5 verification. Therefore, the last recommendation would be to perform a complete reverse engineering code analysis over both files<sup>24</sup> to verify they exactly implement the same functionality and to confirm there are no differences between them, such as a hidden backdoor.

---

<sup>24</sup>Not included because it is out of the scope of this GCFA practical and the MD5 values were verified with the official versions, extracted from where the source code was also downloaded.

## 6 LEGAL IMPLICATIONS

The Spanish law, regulated under the general statutes, “Código Penal (C.P) en España” since 1995 [CP1], defines in its Title 10, first chapter <sup>1 2</sup> [TIT10], the criminal offenses of the acquisition and revelation of secrets or confidential data associated to the privacy right.

In Spain, the penal treatment of the interception of communications cases varies based on the type of the data accessed and the passive subject to whom the criminal action is directed. The privacy of individuals is protected by the “Article 197.1 - Second paragraph”, and it applies to this investigation because it regulates that no one can intercept other’s telecommunications (including electronic ones) or use technical listening/recording communication devices to discover the secrets or harm the privacy of a third party without its consent <sup>3</sup>. In Spain, the exceptions to this article are the consent of the other party, that can be directly accepted or indirectly determined by the court, or a company accessing the employees communications, regulated by the workers’ statute [ESTA1] (“Article 20.3”).

For this criminal offense, the law focuses on the interception of the data itself, not in reviewing its contents or even distributing them. The punishment defined in the law imposes a 1 to 4 years prison and a 12 to 24 months penalty fine <sup>4</sup>. If the illegally intercepted data is disclosed or distributed, then the penalty ascends to a 2 to 5 year prison. Article 197 is focused on protecting, as an asset, the privacy right related to the secrecy of the communications, and as a subjective element, the potential damage over the person affected.

In order to prosecute an individual by these crimes, “Article 201.1” [TIT10] <sup>5</sup> dictates that the victim (or its legal representative) must denounce these facts to initiate the prosecution. Most of the legal details associated to the privacy right in the Spanish law, focusing on computer offenses and crimes, are covered in [CUER1].

For companies and organizations secrets there is another article regulating this behaviours, “Article 278” [TIT10], associated to industrial computer espionage. It also covers new espionage techniques, such as adware or spyware [SPY1].

Additionally, based on the victim harassment concerns, “Article 620 - Second paragraph” <sup>6</sup> (modified by the 15/2003 law, up to 6 months <sup>7</sup>) states that a 10 to 20 days penalty fine could be applied over those that threat, offend, insult or abuse others in an unjustified way <sup>8</sup>. This is considered a minor offense and article 201.1 applies again <sup>9</sup>.

---

<sup>1</sup>“TITULO X. Delitos contra la intimidad, el derecho a la propia imagen y la inviolabilidad del domicilio. - CAPITULO I. Del descubrimiento y revelación de secretos”.

<sup>2</sup><http://delitosinformaticos.com/legislacion/espana.shtml>

<sup>3</sup>The Spanish law is similar to the U.S. Code (Title 18 2511(1)), prohibiting the eavesdropping, use and disclosure of other’s communications.

<sup>4</sup>The Spanish law defines the penalty fines in days; then these are translated to an economic daily quote value (in euros) by the judge for each specific sentence following the principles defined in “Article 50.5” [ART50].

<sup>5</sup>[http://noticias.juridicas.com/base\\_datos/Penal/lo10-1995.l2t10.html#c1](http://noticias.juridicas.com/base_datos/Penal/lo10-1995.l2t10.html#c1)

<sup>6</sup>[http://www.igsap.map.es/cia/dispo/7734\\_4.htm#tit1](http://www.igsap.map.es/cia/dispo/7734_4.htm#tit1)

<sup>7</sup><http://www.igsap.map.es/cia/dispo/27892.htm>

<sup>8</sup>The information found in the USB flashdrive only confirms this offense partially; it should be complemented by the victim’s declaration.

<sup>9</sup>Cases related with sexual harassment (not probed by this investigation) are covered by different articles

There have been several legal cases in Spain involving the usage of the new technologies (e-mail, SMS messages. . . ) to threat and offend other people <sup>10</sup>. They have established jurisprudence and typically are considered as offenses, not as crimes.

At the European level, due to the fact that Spain is part of the European Union, the resolutions of the “Convention on Cybercrime” [EU1] <sup>11</sup> will be applicable in the near future, when Spain will ratify them. Specifically, this case is affected by “Chapter II - Section 1 - Article 3 - illegal interception” that establishes the interception without right by any mean as a criminal offense. On “Article 13” it is determined that each country should adopt the legislative sanctions and measures against this punishable acts, that can include deprivation of liberty as well as other dissuasive sanctions.

The new Spanish law, LSSI <sup>12</sup> doesn't apply in this case because in regulates the rights and obligations of the Information Society electronic service providers (not individuals).

Finally, this investigation is also affected by the CC Terminals company policies, that apart from following the law in relation with the wiretapping acts, should clarify the official allowed usage of the company electronic resources, such as the computers, networks or, specifically, the e-mail service. This affects the victim if the policy specifies that company resources cannot be used for personal usage <sup>13</sup>, and without no doubt to sue the suspect, because the policy should enforce (and condemn) that the company assets are not used for illegal acts, such as harassment or eavesdropping. A very interesting essay covering the legitimate usage of the electronic mail and its legal consequences in Spain is [ALEJ1].

---

of the law.

<sup>10</sup><http://www.delitosinformaticos.com/delitos/injuriassms.shtml>

<sup>11</sup><http://conventions.coe.int/Treaty/en/Treaties/Html/185.htm>

<sup>12</sup>“Ley de Servicios de la Sociedad de la Información y de Comercio Electrónico, LSSI (Ley 34/2002, de 11 de julio)”: <http://delitosinformaticos.com/descarga/lssi.pdf>.

<sup>13</sup>Particularly, using the company computers and networks for sending/receiving private/personal e-mails.

# 7 RECOMMENDATIONS

The following list includes the follow-up actions recommended based on the results of this investigation. Its main goals are: validate and complement the information obtained from the USB flashdrive and, mitigate similar cases in the future.

## 7.1 Victim's confirmation of evidences

The different pieces of evidence found during the analysis must be confirmed directly or by the victim, Leila Conlay, such as her computer IP address, "192.168.2.104", her computer's OS version, her personal e-mail address, "flowergirl96@hotmail.com" or her friend e-mail address, "SamGuarillo@hotmail.com", to increase the results accuracy.

## 7.2 Suspect's computer forensic analysis

Although an exhaustive forensic analysis of the suspect's (Robert Lawrence) computer should be performed to confirm all the evidences found in the USB flashdrive, the following actions have been prioritized based on the files found, and they will notably speed up the computer analysis process.

The WinPcap library, version 3.1 beta 3, must be installed in the suspect's system in order for the WinDump tool to work properly (see chapter 4). This fact must be confirmed, corroborating the usage of an sniffer.

The suspect's mail client should be strictly analyzed. Supposing he uses MS Outlook, both, the current mailbox (Online) and all the locally stored ".pst" files should be found and inspected using the Outlook offline working capabilities, with the goal of finding copies of the e-mails sent to the victim. These mails could be located in multiple places, such as any of his personal folders, the "Deleted Items", the "Sent Items" or the "Draft" folders.

Additionally, the system disks should be reviewed trying to find a copy of all the 7 files found in the USB flashdrive: the MS Word files, the GIF map image and the binary and capture file. If these files were stored or copied to the suspect's PC, it could be possible to find some evidence of its deletion in the filesystem. It could be even possible to find some clues in the Windows Recycle Bin. There are several tools, IEHistory.exe<sup>1</sup> for Windows or rifiuti<sup>2</sup> for Linux, that allow to inspect the Recycle Bin contents and see what files were deleted, their complete path, when they were removed and the user account that did it.

If the suspect run the sniffer it would be possible to find other network capture files in the system disks.

Due to the fact that the e-mails contents have been found in MS Word documents, more specific Word forensic actions could be performed. MS Word keeps a trail of evidence, including the filename under which a file has been stored in the past (and where,

<sup>1</sup><http://www.phillipsponder.com/histviewer.htm>

<sup>2</sup><http://www.foundstone.com/resources/proddesc/rifiuti.htm>

filesystem path) and the Windows user that saved it (as we saw in section 2.5). For example, figure 7.1<sup>3</sup>) shows the effects of having the “Track changes” functionality active in the “coffee.doc” file, as in this investigation. MS Word also keeps track of the most recently opened files, so the 3 “.doc” files found could be listed there.

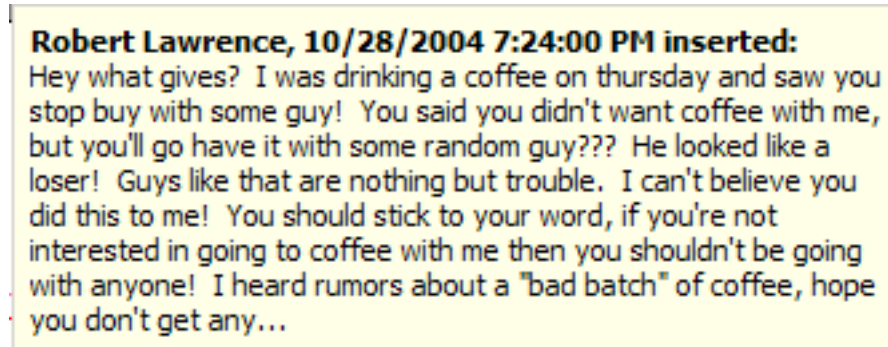


Figure 7.1: MS Word popup of who and when changes were made to this document

Based on the evidences found, it seems that a GIF file was obtained from MS MapPoint. If this hypothesis is correct, this tool, and other maps generated from it, could be found in the system.

The suspect had to access different Web pages to download WinDump and WinPcap. His Web browser should have these accesses saved in its history repository if they have not been explicitly deleted. Supposing he uses MS Internet Explorer, IE, the Windows IEhistory.exe tool (a similar tool, called pasco<sup>4</sup> can be used in Linux) can be used to inspect the IE user files, “.dat”. Not only the IE history will reflect this information, but also the IE cache could include a copy of the HTML pages visited.

Besides, the suspect's computer registry should be inspected, trying to find evidences of the actions discovered in the USB flashdrive in the Windows registry, such as the Most Recently Used files (MRU), including all the files opened, like the Word documents<sup>5</sup>, or executed, like WinDump<sup>6</sup>, the URLs typed (when accessing the WinDump/WinPcap homepages), last command executed, last files saved, last searched files. . .

There are other specific details that can be checked in the registry, based on the in-depth analysis of chapter 4, such as when WinPCap was installed in the suspect's computer, looking at the InstallDate registry key<sup>7</sup>, or from which file it was installed<sup>8</sup>.

Finally, if all these action are not successfully relevant, the computer should be analyzed performing a complete forensic analysis, looking for information related with the case, such as strings included in the “Dirty Word List”, in all its disks and filesystems (including unallocated and slack space), dynamic memory (RAM), Windows pagefile. . .

<sup>3</sup>Placing the pointer over the “coffee.doc” contents pop ups this information.

<sup>4</sup><http://www.foundstone.com/resources/proddesc/pasco.htm>

<sup>5</sup>There is an specific entry for the opened files through MS Word.

<sup>6</sup>There is also an specific entry for the files executed through “Start - Run”.

<sup>7</sup>HKLM\SOFTWARE\Microsoft\Windows\CurrentVersion\Uninstall\WinPcapInst\InstallDate

<sup>8</sup>HKLM\SOFTWARE\Microsoft\Windows\CurrentVersion\Uninstall\WinPcapInst\InstallSourceFile



## 7.3 CC Terminals infrastructure analysis

This subsection mainly focuses on the analysis that can be performed at the corporate level, based on the network traffic and files found in the USB flashdrive.

The company, CC Terminals, mail server logs should be inspected, trying to find events associated to the e-mails sent by Robert Lawrence to Leila Conlay (both, at her company and personal e-mail addresses). Besides, the corporate mail server backup should be analyzed looking for the same type of mail activities <sup>9</sup>.

The victim's (Leila) Hotmail account and company mailbox should be accessed (with the victim consent) to inspect the headers of the e-mails received from the suspect, if they are still available.

Due to the fact the e-mail was the main method used to harass the victim and that, technically speaking, the e-mail headers are hard to spoof, additional actions are focused on the e-mail events. The mail headers keep track of the originating IP address each time the message passes through an SMTP (mail) relay until its final destination. This would help to verify Robert Lawrence was indeed the person that sent the e-mails; although the e-mail address can be spoofed very easily, obtaining the whole path the mail has passed through would confirm which was the first mail server that received it and the IP address of the originating PC (probably the suspect's PC). It also will show when it was sent (based on the headers timestamps set in every intermediate mail relay, MTA) during its whole path to the destination.

The same inspection that can be made at the client level (suspect's PC) can be accomplished at the corporate level, looking into the company Web proxy for access to all the external URLs potentially used, like the WinDump/WinPcap homepages. The client IP address should be registered in the proxy logs with the timestamp of the access <sup>10</sup>. This could also be extended to the company content filtering devices, such as the e-mail antivirus server or the Web/URL filtering server. The logs of all these devices and the protection perimeter solution, such as the firewalls or the network IDSes, based on the way they would have been configured, could help to verify the evidences found: TCP Web traffic to port 80 and TCP SMTP traffic to port 25.

Based on the SNMP Traps registered in the "capture" file, the CC Terminals Network Enterprise Management system logged all Internet outbound connections. Its logs must be inspected, looking for events associated to the suspect's Internet activities, like (again) the Web connections required to obtain the WinDump/WinPcap packages. The logs will corroborate when and from which IP address and port these Web pages were accessed. To be able to correlate the events obtained from all these different sources, CC Terminals must have all its devices synchronized through NTP.

Finally, it would also be interesting to get case evidences from third parties, such as the ones associated to the free e-mail service usage, like Hotmail in this case. The Hotmail logs would confirm the reception of Robert's e-mails in Leila's Hotmail personal e-mail account. This is a hard task because it is required to work in conjunction with authorities to get the required information from the service provider, but it adds lots of credibility to the evidences because the involvement of a third party <sup>11</sup>.

<sup>9</sup>The inspection of backups also applies to the suspect's computer, if available.

<sup>10</sup>This checking methodology, client and corporate checking, applies to both, e-mail and Web usage.

<sup>11</sup>Besides, while some of the suggested checks could be manipulated by the suspect, having enough control of the CC Terminals infrastructure, it will be very difficult for him to modify other third party sources.

## 7.4 CC Terminals physical and policy analysis

In relation with the case events, the company physical access registry (controlling the physical access to the CC Terminals facilities) should be reviewed to verify what time the suspect entered and left the company building the day the victim was harassed at the coffee shop, that is, the 28th of October, 2004.

Again, with the support of the authorities, the coffee shop surveillance system could be reviewed confirming the victim's declaration about being there the afternoon of October, 28th, and even it could also confirm that in fact the suspect arrived at the coffee shop and when.

Due to the fact that the suspect made numerous attempts to meet her, probably some of them were through phone, and some additional evidences could be collected from the victim's company or home phones and personal cellular voice mail.

This time, the usage of a portable disk device, like the USB flashdrive, has helped to obtain lots of evidences related to the investigation, but CC Terminals should have a defined policy specifying if the usage of these devices is allowed. In many environments these devices are a very important threat associated to information leakage <sup>12</sup>.

## 7.5 Preventive actions

The main technical issue associated with this case is the possibility of capturing the network traffic of other parties within the company (in promiscuous mode). The following list of actions try to detect and avoid similar incidents in the future in CC Terminals.

Blocking the usage of a promiscuous sniffer in Windows is not an easy task; besides, it is not possible to limit what users can execute it, because no high privilege (Administrator level) is required in Windows (like in the Unix world) to run it and there is no right or permission definition to block it at the OS level. The Windows "Network Monitor Driver" is not required for all sniffers <sup>13</sup>, so removing the network card bindings for the built-in driver doesn't help. None of the Group Policy configuration settings apply to the network card promiscuous status either.

Additionally, there is no way to detect at the OS level changes in the network card status when it is set up in promiscuous mode; the Windows logging system doesn't save any event ("Event Viewer", `eventvwr.exe`) denoting these activities and there is no tool to manually check this status.

However, there are specific Windows programs (open-source and commercial) to detect systems whose network interface is in promiscuous mode, such as, proDETECT [PROD1], PromiScan [PROM1] or AntiSniff [ANTI1] <sup>14</sup>. Therefore CC Terminals should use this kind of solutions [SANA1] to log these activities. Generally speaking (it could vary between different Windows OS and SP versions and networks cards and hardware), technically, a Windows system running in promiscuous mode will generate a response to a packet send to the `ff:00:00:00:00:00` MAC address.

Based on the information available, it seems CC Terminals is using shared network

<sup>12</sup>It is relatively easy to extract company confidential information stored in them without being discovered.

<sup>13</sup>They have their own packet capture driver, such as the WinPcap "`npf.sys`".

<sup>14</sup>There are also similar solutions in the Unix world, but these are recommended due to the Windows nature of this investigation.

segments based on HUBs, what helped Robert to sniff Leila's traffic. They should use a switched environment instead, where although traffic can be sniffed using ARP spoofing methods [ARPS1], it is a much more complex attack and easy to detect, using strict ARP traffic monitoring devices.

CC Terminals company policy should enforce the usage of encrypted traffic for all the personal/private communications they don't want to be intercepted using any of the nowadays interoperable protocols, like SSL, IPsec, SSH. . . . Additionally, the company must ensure that the punishments defined in the policy for wiretapping activities and misuse of the company e-mail infrastructure are strictly applied to deter similar future events.

© SANS Institute 2005, Author retains full rights

# 8 ADDITIONAL INFORMATION

## 8.1 “Digital Investigation” Journal

“Digital Investigation” (“The International Journal of Digital Forensics & Incident Response”) is a technical journal focused on the research, best-practices, new developments and methodologies related to the art of digital forensics, published by Elsevier <sup>1</sup>. Although it is a subscription service, some of its articles are freely available in PDF format at <sup>2</sup>:

- “Time and date issues in forensic computing - a case study”. Chris Boyd, Pete Forster. Digital Investigation. Volume 1 Issue 1. (March 2004)
- “Forensic analysis of Windows hosts using UNIX-based tools”. Cory Altheide. Digital Investigation. Volume 1 Issue 3. (October 2004)

The former covers UTC timestamps translation issues, although mainly related with Internet Explorer, they are aligned to the timezone issues we dealt with in this case. The later is another example of how to use Linux to forensically analyze a Windows host.

## 8.2 National Institute of Justice (NIJ)

The National Criminal Justice Reference Service (NCJRS) of the National Institute of Justice (NIJ) <sup>3</sup> has published one report and one guide for the law enforcement community, containing recommendations about how to act in the incident response phase and how to handle the digital evidences. Both are very useful when merging the law enforcement and the technical worlds, as this report’s “Executive Summary”:

- “Forensic Examination of Digital Evidence: A Guide for Law Enforcement”:  
<http://www.ncjrs.org/pdffiles1/nij/199408.pdf> (April 2004)
- “Electronic Crime Scene Investigation: A Guide for First Responders”:  
<http://www.ncjrs.org/pdffiles1/nij/187736.pdf> (July 2001)

## 8.3 The Sleuth Kit Informer

The Sleuth Kit Informer is an electronic newsletter for The Sleuth Kit, Autopsy, and their related tools. Its goal is to increase awareness, knowledge, and documentation for these tools, going from tool design details to techniques on dissecting a disk image using them: <http://www.sleuthkit.org/informer/index.php>. (February 2003 - November 2004)

As an investigator, there is a lot you can learn from these technical research articles about the usage of TSK [SLEU1] and Autopsy [AUTO1]. You can subscribe at <sup>4</sup>.

<sup>1</sup>[http://www.elsevier.com/wps/find/journaldescription.cws\\_home/702130/description](http://www.elsevier.com/wps/find/journaldescription.cws_home/702130/description)

<sup>2</sup><http://www.compseconline.com/digitalinvestigation/tableofcontents.htm#issue1>

<sup>3</sup>The Office of Justice Programs, U.S. Department of Justice.

<sup>4</sup><http://lists.sourceforge.net/lists/listinfo/sleuthkit-informer>

## 8.4 Analyzing Windows binaries

The following list includes a set of the most relevant and advanced technical references for the analysis of Windows binaries of year 2004, like the one used by the suspect in this case, using complex reverse engineering methods. They are a reference to complement the basic behavioral analysis performed in this report:

- “Attacking Obfuscated Code with IDA Pro”. Chris Eagle. Black Hat USA 2004. <http://www.blackhat.com/presentations/bh-usa-04/bh-us-04-eagle.pdf>
- The latest two HoneyNet Project “Scan of the Month” challenges, SotM 32<sup>5</sup> and 33<sup>6</sup>, are both focused on reverse engineering Windows code.
- “Reverse Code Engineering: An In-Depth Analysis of the Bagle Virus”. Konstantin Rozinov. August 2004.<sup>7</sup>

## 8.5 FAT filesystem layout

Due to the fact that this case analysis focuses on a FAT filesystem evidence, it is a must to know as much as possible about this filesystem layout, its structures and elements; therefore, it is important for the reader to have the list of FAT references (see **H**) that have been used during the analysis, including the official Microsoft specification: **[DISK1]**, **[FAT1]**, **[KAMP1]** and **[MSFAT1]**.

## 8.6 E-Evidence Information and Resource

The “E-Evidence Information and Resource”<sup>8</sup> site contains a compilation of links to material related to all aspects of Digital Forensics and Electronic Evidence. It is owned by Christine Siedsma, a professor and Project Manager at the Computer Forensic Research and Development Center (CFRDC) at Utica College. Between others it contains a compilation of Digital Forensics resources<sup>9</sup> (very useful as a reference) and includes a section of programs and utilities (some for Windows) for data acquisition, recovery, auditing...<sup>10</sup>.

## 8.7 Sniffing (network wiretap, sniffer) FAQ

The “Sniffing (network wiretap, sniffer) FAQ” by Robert Graham is the most famous and complete Internet document about eavesdropping on computer networks<sup>11</sup>. This investigation’s main threat was the eavesdropping of victim’s traffic, therefore it is a must for an investigator to know as much as possible about this method: how it works, how to defend against it, the tools used, protocols knowledge... .

<sup>5</sup><http://honeynet.org/scans/scan32/index.html>

<sup>6</sup><http://honeynet.org/scans/scan33/index.html>

<sup>7</sup>[http://rozinov.sfs.poly.edu/papers/bagle\\_analysis\\_v.1.0.pdf](http://rozinov.sfs.poly.edu/papers/bagle_analysis_v.1.0.pdf)

<sup>8</sup><http://www.e-evidence.info>

<sup>9</sup><http://www.e-evidence.info/links.html>

<sup>10</sup><http://www.e-evidence.info/other.html>

<sup>11</sup><http://www.dsinet.org/textfiles/faqs/Sniffing-FAQ.html>

# A CHAIN OF CUSTODY FORM AND CUSTODY FACILITIES

The original chain of custody form (see figure A.1), provided by the security administrator, Mark Mawer, and associated to the USB flashdrive (see figure A.2, [LEXAR1]) image to be analyzed has been extended using the form of table A.1<sup>1</sup>.

- Tag #:	USBFD-64531026-RL-001
- Description:	64M Lexar Media JumpDrive
- Serial #:	JDSP064-04-5000C
- Image:	USBFD-64531026-RL-001.img
- MD5:	338ecf17b7fc85bbb2d5ae2bbc729dd5

Figure A.1: Original chain of custody form



Figure A.2: Evidence: Lexar Media 64 MB JumpDrive Portable USB Flash Drive

The chain of custody form reflects information that allows identifying and keeping track of the evidence, establishing each person who has had custody of it and when. This information establishes continuity of possession, proves the integrity of the handling of the evidence collected and will help to prove to court of law it has not been altered.

In essence, it tries to respond WHO has handled the evidence?, WHAT is the evidence?, WHEN was the evidence collected and transferred to another entity?, WHERE was the evidence collected and stored?, HOW was the evidence obtained and stored? and WHY was the evidence collected (the purpose)?. The forensic analysis complements this information responding to WHAT are the actions that have been performed on it? and WHEN each action took place?.

Table A.1 - "Technical Data" field:

*"An image of the evidence has been obtained for its forensic analysis. With the goal of not damaging the initial evidence (USB flashdrive) an exact copy (image) has been written to a CD-R media (non-writable), labeled accordingly (including file name, size and MD5*

<sup>1</sup>Its contents are based on information provided in the "SECURITY 508: System Forensics, Investigation & Response (Track 8)" SANS training: <http://www.sans.org>.



Evidence feature	Description
Date and time evidence was seized:	October 29th, 2004 - 22:30h.
Location and who it was obtained from:	Robert Lawrence's cubicle - Sales department.
Make and model:	64M Lexar Media JumpDrive.
Serial number:	JDSP064-04-5000C.
Name and signature of individual(s) who collected evidence:	Mark Mawer, security administrator ( <b>signature</b> )
Description:	Standard USB flashdrive (Capacity: 64MB).
Name and signature of person receiving evidence:	Raul Siles, forensic investigator ( <b>signature</b> )
Case number:	GCFA-v2.0-Option1-11-2004.
Number of evidence (tag):	USBFD-64531026-RL-001.
Hash values (MD5):	338ecf17b7fc85bbb2d5ae2bbc729dd5.
Technical data:	"See text outside this table"
Image name:	USBFD-64531026-RL-001.img
Image size:	62439424 bytes

Table A.1: Evidence chain of custody form (extended)

value) and saved together with the initial evidence this form describes. The USB drive is Windows 98SE, ME/XP and 2000 and Mac OS 8.6, 9.X and 10.X compatible [LEXAR1]."

The CC Terminals incident response and forensic investigation teams have access to an evidence locker room where the evidences of all the cases they are or have been involved in are safely stored. Inside the room there are individual locked cabinets owned, and only accessible by, a single investigator.

The company physical security controls ensure that the cabinets are not manipulated or replaced. The evidence locker room has strict access controls in place to only allow authorized personnel to get into. Additionally, every time an investigator enters the room he/she must fill and sign up the room registry book (a numbered paper book) that documents the purpose why the room has been accessed and when. This registry is periodically reviewed and audited.

To add an extra layer of security, the room has a camera (connected to a VHS recorder) with a motion detector installed. The camera would automatically turn on when someone entered the room and monitors anyone entering, moving within, or leaving the room. A battery backup was added to the camera, being able to run during a full weekend [ADDI1].

Finally, the forensic workstations used by the investigators are strictly hardened and protected, both from the physical and logical perspectives. While used in a case, they are not connected to any network minimizing the chance of being remotely compromised. Between cases they are updated (new tools and patches) following very strict and monitored procedures. These systems are built from an specific secured OS image that is periodically upgraded.

# B FORENSIC ANALYSIS ENVIRONMENT

This chapter's sections contain a very brief description of the forensic workstation used for the USB flashdrive image analysis and the Windows binary analysis workstation and lab.

## B.1 Forensic workstation

The forensic workstation is based on a Compaq Evo N610c laptop, with 1 GB of RAM, a DVD-RW and a 40 GB hard disk. It has 2 USB ports to plug external evidence hard drives or USB flashdrives (like the one analyzed). The OS installed is a Linux Fedora Core 2, with the latests upgrades up to November, 21th (11:00h), 2004. The system main drive has been imaged using PartImage v0.6.4 <sup>1</sup>, so a clean copy is always ready for every new incident. The goal of this system is to have a media analysis, flexible and portable laboratory.

A Linux based system was considered the best option because it allows multiplatform forensic analysis, being able to understand several filesystems: NTFS, FAT, VFAT, Solaris, ext2, ext3, MAC, BSD... The machine has been strictly hardened and `iptables` is used as the personal firewall, denying all incoming connections. During a case, the laptop is isolated; never is connected to any network. All data transfers are performed using reliable media, such as clean CD-R disks or USB flashdrives. Additionally, the laptop activities are monitored using a filesystem integrity checker, `Aide` (v0.10) <sup>2</sup> and a log analyzer, `Logwatch` (v5.1) <sup>3</sup>. Only one user with the maximum privileges (root) is configured in the system and it is only accessible by the system investigator (owner).

The following list is the directory structure used in the forensic workstation for this case (see the file MD5 values in appendix H), built over the ext3-based main system disk:

- `/T8c` (link to `/mnt/vfat/T8c.image`): Directory used to store the USB image and all the data derived from it, such as, the partition images, the files extracted from the USB flashdrive (`/T8c/files`), the timeline (`/T8c/timeline`), the network captures (`/T8c/captures`) and the Slack space (`/T8c/slack`).
- `/mnt/vfat/T8c_binaries`: Repository for the official binary files downloaded from Internet, such as the different WinDump and WinPcap versions.
- `/mnt/vfat/T8c`: Report repository - documentation, screen captures...
- `/opt/EvidenceLocker/GCFA-v2.0-Option1-11-2004/LexarUSB/`: Autopsy Evidence Locker directory where all the Autopsy case information is stored, being the `"images"` and the `"output"` subdirectories the most important ones.

## B.2 Reverse Engineering workstation

The selection of the Reverse Engineering workstation OS was partially based on the network forensic analysis of chapter 2, supposing the suspect had a similar Windows version

---

<sup>1</sup><http://www.partimage.org>

<sup>2</sup><http://sourceforge.net/projects/aide>

<sup>3</sup><http://www.logwatch.org>

as the victim (W2K SP2+ or WinXP SP1). The lab is composed of a single laptop running VMWare v.4.5.2 build-8848 over Windows XP (see the details in figure B.1). Two additional virtual machines (VM) have been configured, one based on Windows, where the binary to be analyzed will be executed and monitored, and another running Linux, responsible of monitoring the lab virtual network traffic and providing any software service required (HTTPd, SMTPd, IRCd...). The two virtual machines have been preloaded with lots of multiple software packages used to monitor and research all the actions performed by the programs analyzed. Only the ones used in this analysis will be mentioned in chapter 4.

Following strict isolation precautions, all these three OS are interconnected only using a host-based VMWare virtual network (10.10.10.0/24). There is no other physical network connected to the real system (laptop). Additionally the laptop has antivirus software (Symantec Antivirus v9.0) and a personal firewall (Sygate Security Agent v3.5) installed that are regularly updated in a controlled environment.

```
- Reverse Engineering analysis laptop:

HW model: HP nx9010
OS version: Windows XP SP1
IP address: 10.10.10.1/24
Default gateway: None
DNS servers: None
RAM memory: 1 GB
Disk size: 40 GB

- Windows binary analysis virtual machine:

OS version: Windows 2000 SP4
IP address: 10.10.10.20/24
Default gateway: 10.10.10.1
DNS servers: None
RAM memory: 128 MB
Disk size: 4 GB

- Linux services and traffic capture analysis virtual machine:

OS version: Linux Fedora Core 1 (2.4.22-1.2115.npt1)
IP address: 10.10.10.10/24
Default gateway: 10.10.10.1
DNS Servers: None
RAM memory: 128 MB
Disk size: 4 GB
```

Figure B.1: Reverse Engineering Lab: system's details

# C USB FLASHDRIVE TIMELINE

**From:** 25 Nov, 2004 (08:00) **To:** 25 Nov, 2004 (09:00)

Find below the contents of the "mactime.txt" file containing the complete timeline of events extracted from the USB flashdrive:

```
Mon Oct 25 2004 00:00:00 19968 .a. -/rwxrwxrwx 0 0 3 -r/her.doc
Mon Oct 25 2004 08:32:06 19968 .c. -/rwxrwxrwx 0 0 3 -r/her.doc
Mon Oct 25 2004 08:32:08 19968 m.. -/rwxrwxrwx 0 0 3 -r/her.doc
Tue Oct 26 2004 00:00:00 19968 .a. -/rwxrwxrwx 0 0 4 -r/hey.doc
Tue Oct 26 2004 08:48:06 19968 .c. -/rwxrwxrwx 0 0 4 -r/hey.doc
Tue Oct 26 2004 08:48:10 19968 m.. -/rwxrwxrwx 0 0 4 -r/hey.doc
Wed Oct 27 2004 00:00:00 485810 .a. -/rwxrwxrwx 0 0 7 -r/WinPcap_3_1_beta_3.exe (_INPCA~1.EXE) (deleted)
450560 .a. -/rwxrwxrwx 0 0 12 -r/WinDump.exe (_INDUMP.EXE) (deleted)
0 .a. -rwxrwxrwx 0 0 7 <usb_FAT16-_INPCA~1.EXE-dead-7>
0 .a. -rwxrwxrwx 0 0 12 <usb_FAT16-_INDUMP.EXE-dead-12>
Wed Oct 27 2004 16:23:50 485810 m.. -rwxrwxrwx 0 0 10 <usb_FAT16-_INPCA~1.EXE-dead-10>
485810 m.. -/rwxrwxrwx 0 0 10 -r/WinPcap_3_1_beta_3.exe (_INPCA~1.EXE) (deleted)
Wed Oct 27 2004 16:23:54 485810 .c. -/rwxrwxrwx 0 0 7 -r/WinPcap_3_1_beta_3.exe (_INPCA~1.EXE) (deleted)
0 .c. -rwxrwxrwx 0 0 7 <usb_FAT16-_INPCA~1.EXE-dead-7>
485810 .c. -/rwxrwxrwx 0 0 10 -r/WinPcap_3_1_beta_3.exe (_INPCA~1.EXE) (deleted)
485810 .c. -rwxrwxrwx 0 0 10 <usb_FAT16-_INPCA~1.EXE-dead-10>
Wed Oct 27 2004 16:23:56 485810 m.. -/rwxrwxrwx 0 0 7 -r/WinPcap_3_1_beta_3.exe (_INPCA~1.EXE) (deleted)
0 m.. -rwxrwxrwx 0 0 7 <usb_FAT16-_INPCA~1.EXE-dead-7>
Wed Oct 27 2004 16:24:02 450560 m.. -rwxrwxrwx 0 0 14 <usb_FAT16-_INDUMP.EXE-dead-14>
450560 m.. -/rwxrwxrwx 0 0 14 -r/WinDump.exe (_INDUMP.EXE) (deleted)
Wed Oct 27 2004 16:24:04 450560 .c. -rwxrwxrwx 0 0 14 <usb_FAT16-_INDUMP.EXE-dead-14>
450560 .c. -/rwxrwxrwx 0 0 14 -r/WinDump.exe (_INDUMP.EXE) (deleted)
450560 .c. -/rwxrwxrwx 0 0 12 -r/WinDump.exe (_INDUMP.EXE) (deleted)
0 .c. -rwxrwxrwx 0 0 12 <usb_FAT16-_INDUMP.EXE-dead-12>
Wed Oct 27 2004 16:24:06 450560 m.. -/rwxrwxrwx 0 0 12 -r/WinDump.exe (_INDUMP.EXE) (deleted)
0 m.. -rwxrwxrwx 0 0 12 <usb_FAT16-_INDUMP.EXE-dead-12>
Thu Oct 28 2004 00:00:00 450560 .a. -rwxrwxrwx 0 0 14 <usb_FAT16-_INDUMP.EXE-dead-14>
53056 .a. -/rwxrwxrwx 0 0 15 -r/_apture (deleted)
485810 .a. -/rwxrwxrwx 0 0 10 -r/WinPcap_3_1_beta_3.exe (_INPCA~1.EXE) (deleted)
53056 .a. -rwxrwxrwx 0 0 15 <usb_FAT16-_apture-dead-15>
8814 .a. -/rwxrwxrwx 0 0 17 -r/_ap.gif (deleted)
485810 .a. -rwxrwxrwx 0 0 10 <usb_FAT16-_INPCA~1.EXE-dead-10>
8814 .a. -rwxrwxrwx 0 0 17 <usb_FAT16-_ap.gif-dead-17>
19968 .a. -/rwxrwxrwx 0 0 18 -r/coffee.doc
0 .a. -rwxrwxrwx 0 0 16 <usb_FAT16-_ap.gif-dead-16>
8814 .a. -/rwxrwxrwx 0 0 16 -r/_ap.gif (deleted)
450560 .a. -/rwxrwxrwx 0 0 14 -r/WinDump.exe (_INDUMP.EXE) (deleted)
Thu Oct 28 2004 11:08:24 53056 .c. -/rwxrwxrwx 0 0 15 -r/_apture (deleted)
53056 .c. -rwxrwxrwx 0 0 15 <usb_FAT16-_apture-dead-15>
Thu Oct 28 2004 11:11:00 53056 m.. -rwxrwxrwx 0 0 15 <usb_FAT16-_apture-dead-15>
53056 m.. -/rwxrwxrwx 0 0 15 -r/_apture (deleted)
Thu Oct 28 2004 11:17:44 8814 .c. -rwxrwxrwx 0 0 17 <usb_FAT16-_ap.gif-dead-17>
0 .c. -rwxrwxrwx 0 0 16 <usb_FAT16-_ap.gif-dead-16>
8814 .c. -/rwxrwxrwx 0 0 17 -r/_ap.gif (deleted)
8814 .c. -/rwxrwxrwx 0 0 16 -r/_ap.gif (deleted)
Thu Oct 28 2004 11:17:46 8814 m.. -/rwxrwxrwx 0 0 16 -r/_ap.gif (deleted)
8814 m.. -/rwxrwxrwx 0 0 17 -r/_ap.gif (deleted)
0 m.. -rwxrwxrwx 0 0 16 <usb_FAT16-_ap.gif-dead-16>
8814 m.. -rwxrwxrwx 0 0 17 <usb_FAT16-_ap.gif-dead-17>
Thu Oct 28 2004 19:24:46 19968 .c. -/rwxrwxrwx 0 0 18 -r/coffee.doc
Thu Oct 28 2004 19:24:48 19968 m.. -/rwxrwxrwx 0 0 18 -r/coffee.doc
```

Entries like the one below denote that the file was deleted but its file name entry structure (the corresponding Directory Entry (DE), number 12) still exists:

```
Wed Oct 27 2004 00:00:00 450560 .a. -/rwxrwxrwx 0 0 12 -r/WinDump.exe (_INDUMP.EXE) (deleted)
```

Entries like the one below denote that the DE number 12 is unallocated on the image:

```
Wed Oct 27 2004 16:24:06 0 m.. -rwxrwxrwx 0 0 12 <usb_FAT16-_INDUMP.EXE-dead-12>
```

The following facts related to the FAT filesystem must be considered in the analysis:

[TIME1] *"FAT stores the Written, Accessed, and Created time, although by spec the Created and Access times are optional and the Access time is only accurate to the day."*

[FAT1] *"Each file in FAT can store up to three times (last accessed, written, and created). The last written time is the only 'required' time and is accurate to a second. The create time is optional and is accurate to the tenth of a second (Note that I have seen several system directories in Windows that have a create time of 0). The last access time is also optional and is only accurate to the day (so the times are 00:00:00 in The Sleuth Kit)."*

The Windows FAT timestamp behaviour can be briefly summarized as follows [TIME2]: The creation time specifies when the file was created or copied to the media. The modification time shows the last write operation <sup>1</sup>. However, in a copy it preserves the original (system) timestamp, so a file copied can have a modification time older than its creation time; it looks like it was created after it was modified. The way to determine this can be based on the access time that is displayed when the file was opened, copied, printed. . .

---

<sup>1</sup>This time (m) is the one showed by the `dir` command or the Windows File Manager.

# D USB FLASHDRIVE TIMELINE SIMULATION SESSION

---

**From:** 25 Nov, 2004 (16:00) **To:** 25 Nov, 2004 (17:30)

---

This exercise was performed in order to completely understand the timeline events found on the evidence (see appendix C).

A 64 MB USB flashdrive was formatted from scratch (containing only one partition) in a Windows 2000 SP4 system using the FAT filesystem type. Once formatted, Internet Explorer (v5.00.3700.1000) was used to navigate to <http://www.google.com> and the default Google homepage image (GIF file) was saved directly from the Internet Explorer program to the USB disk, as "google.gif". Then, the Web browser was used to access [http://winpcap.polito.it/install/bin/WinPcap\\_3\\_1\\_beta\\_3.exe](http://winpcap.polito.it/install/bin/WinPcap_3_1_beta_3.exe) and <http://windump.polito.it/install/default.htm> to download the WinPcap library and the WinDump tool respectively from IE, again saved directly to the USB media.

This simulation session tried to reproduce the exact steps that were supposedly performed by Robert Lawrence based on the hypothesis obtained after performing the evidence timeline analysis (see section 2.3). The complete analysis of the simulated session and the obtained "mactime\_fls" file contents are:

```
# dd if=/dev/sda1 of=/tmp/USB_Monica/sda1.img bs=512
127712+0 records in
127712+0 records out
#
# fls -f fat16 -a -m -r /tmp/USB_Monica/sda1.img > /tmp/USB_Monica/sda_body_fls
# ils -f fat16 -m /tmp/USB_Monica/sda1.img > /tmp/USB_Monica/sda_body_ils
# fls -f fat16 -a -m -r /tmp/USB_Monica/sda1.img > /tmp/USB_Monica/sda_body
# ils -f fat16 -m /tmp/USB_Monica/sda1.img >> /tmp/USB_Monica/sda_body
# ll
total 64000
-rw-r--r-- 1 root root 65388544 Nov 25 16:21 sda1.img
-rw-r--r-- 1 root root 34877 Nov 25 16:23 sda_body
-rw-r--r-- 1 root root 658 Nov 25 16:22 sda_body_fls
-rw-r--r-- 1 root root 34219 Nov 25 16:23 sda_body_ils
# mactime -b /tmp/USB_Monica/sda_body_fls > /tmp/USB_Monica/mactime_fls
# ll
...
-rw-r--r-- 1 root root 1998 Nov 25 16:23 mactime_fls
#
# cat mactime_fls
Sat Nov 06 2004 19:29:48 4277 m.. -/rwxrwxrwx 0 0 4 -r/google.gif
Sat Nov 25 2004 00:00:00 0 .a. -/rwxrwxrwx 0 0 3 -r/_oogle.gif (deleted)
485810 .a. -/rwxrwxrwx 0 0 10 -r/WinPcap_3_1_beta_3.exe (WINPCA~1.EXE)
450560 .a. -/rwxrwxrwx 0 0 14 -r/WinDump.exe (WINDUMP.EXE)
0 .a. -/rwxrwxrwx 0 0 7 -r/WinPcap_3_1_beta_3.exe (_INPCA~1.EXE) (deleted)
0 .a. -/rwxrwxrwx 0 0 12 -r/WinDump.exe (_INDUMP.EXE) (deleted)
4277 .a. -/rwxrwxrwx 0 0 4 -r/google.gif
Sat Nov 25 2004 16:16:24 4277 ..c -/rwxrwxrwx 0 0 4 -r/google.gif
0 ..c -/rwxrwxrwx 0 0 3 -r/_oogle.gif (deleted)
Sat Nov 25 2004 16:16:26 0 m.. -/rwxrwxrwx 0 0 3 -r/_oogle.gif (deleted)
Sat Nov 25 2004 16:17:32 0 ..c -/rwxrwxrwx 0 0 7 -r/WinPcap_3_1_beta_3.exe (_INPCA~1.EXE) (deleted)
485810 ..c -/rwxrwxrwx 0 0 10 -r/WinPcap_3_1_beta_3.exe (WINPCA~1.EXE)
Sat Nov 25 2004 16:17:34 0 m.. -/rwxrwxrwx 0 0 7 -r/WinPcap_3_1_beta_3.exe (_INPCA~1.EXE) (deleted)
Sat Nov 25 2004 16:17:42 485810 m.. -/rwxrwxrwx 0 0 10 -r/WinPcap_3_1_beta_3.exe (WINPCA~1.EXE)
Sat Nov 25 2004 16:18:22 450560 ..c -/rwxrwxrwx 0 0 14 -r/WinDump.exe (WINDUMP.EXE)
0 ..c -/rwxrwxrwx 0 0 12 -r/WinDump.exe (_INDUMP.EXE) (deleted)
```



```

Sat Nov 25 2004 16:18:24      0 m.. -/-rwxrwxrwx 0 0 12 -r/WinDump.exe (_INDUMP.EXE) (deleted)
Sat Nov 25 2004 16:18:32 450560 m.. -/-rwxrwxrwx 0 0 14 -r/WinDump.exe (WINDUMP.EXE)
#

```

Fortunately, even the Directory Entries numbers for the two .exe files match between the real incident case disk image and the simulation session, making easier the comparison between the two. The real case timeline records equivalent to the ones showed above are:

```

Wed Oct 27 2004 00:00:00 485810 .a. -/-rwxrwxrwx 0 0 7 -r/WinPcap_3_1_beta_3.exe (_INPCA~1.EXE) (deleted)
450560 .a. -/-rwxrwxrwx 0 0 12 -r/WinDump.exe (_INDUMP.EXE) (deleted)
Wed Oct 27 2004 16:23:50 485810 m.. -/-rwxrwxrwx 0 0 10 -r/WinPcap_3_1_beta_3.exe (_INPCA~1.EXE) (deleted)
Wed Oct 27 2004 16:23:54 485810 .c. -/-rwxrwxrwx 0 0 7 -r/WinPcap_3_1_beta_3.exe (_INPCA~1.EXE) (deleted)
485810 .c. -/-rwxrwxrwx 0 0 10 -r/WinPcap_3_1_beta_3.exe (_INPCA~1.EXE) (deleted)
Wed Oct 27 2004 16:23:56 485810 m.. -/-rwxrwxrwx 0 0 7 -r/WinPcap_3_1_beta_3.exe (_INPCA~1.EXE) (deleted)
Wed Oct 27 2004 16:24:02 450560 m.. -/-rwxrwxrwx 0 0 14 -r/WinDump.exe (_INDUMP.EXE) (deleted)
Wed Oct 27 2004 16:24:04 450560 .c. -/-rwxrwxrwx 0 0 12 -r/WinDump.exe (_INDUMP.EXE) (deleted)
450560 .c. -/-rwxrwxrwx 0 0 14 -r/WinDump.exe (_INDUMP.EXE) (deleted)
Wed Oct 27 2004 16:24:06 450560 m.. -/-rwxrwxrwx 0 0 12 -r/WinDump.exe (_INDUMP.EXE) (deleted)
Thu Oct 28 2004 00:00:00 485810 .a. -/-rwxrwxrwx 0 0 10 -r/WinPcap_3_1_beta_3.exe (_INPCA~1.EXE) (deleted)
8814 .a. -/-rwxrwxrwx 0 0 17 -r/_ap.gif (deleted)
8814 .a. -/-rwxrwxrwx 0 0 16 -r/_ap.gif (deleted)
450560 .a. -/-rwxrwxrwx 0 0 14 -r/WinDump.exe (_INDUMP.EXE) (deleted)
Thu Oct 28 2004 11:17:44 8814 .c. -/-rwxrwxrwx 0 0 17 -r/_ap.gif (deleted)
8814 .c. -/-rwxrwxrwx 0 0 16 -r/_ap.gif (deleted)
Thu Oct 28 2004 11:17:46 8814 m.. -/-rwxrwxrwx 0 0 16 -r/_ap.gif (deleted)
8814 m.. -/-rwxrwxrwx 0 0 17 -r/_ap.gif (deleted)

```

Therefore it seems that the reason why a given file is duplicated is due to the way some applications, like Internet Explorer, save files directly to disk, creating a temporary file copy previously.

# E ALLOCATED PARTITION ENTRY AND BOOT SECTOR ANALYSIS

---

**From:** 24 Nov, 2004 (18:00) **To:** 24 Nov, 2004 (20:00)

---

To deal with the `fdisk` and `sfdisk` error messages found, the initial sector containing the Partition Table (also known as sector 0 or MBR, Master Boot Record) was analyzed in depth using an hexadecimal editor, the Linux `hexedit` tool (v1.2.7-2). The table can describe up to 4 partitions and includes the starting sector, the number of sectors and the type (which typically corresponds to the file system in it). Additionally a starting and ending CHS values (Cylinder/Head/Sector) also exist for each partition.

The first 446 bytes of the first sector contain the bootloader [PARTA1] (analyzed in section E.2), and the partition table itself is located at offset 0x1BE (64 bytes) and there is a 2 byte signature at the end, 0xAA55 (previously explained), analyzed in the next section, E.1. It was confirmed that the disk doesn't have other primary or extended partitions.

Additionally, an analysis of the effects of changing the FAT16 partition type was performed (see section E.3).

## E.1 Disk allocated partition entry and CHS analysis

Both, the standard `fdisk` and `sfdisk` Linux commands detected that there was a discrepancy in the C/H/S values of the disk FAT partition:

```
# fdisk
...
    phys=(249, 16, 32) logical=(224, 2, 31)
...
# sfdisk -luS /T8c/usbfd-64531026-rl-001.img
Disk /T8c/usbfd-64531026-rl-001.img: cannot get size
Disk /T8c/usbfd-64531026-rl-001.img: cannot get geometry

Disk /T8c/usbfd-64531026-rl-001.img: 0 cylinders, 0 heads, 0 sectors/track
Warning: The partition table looks like it was made
    for C/H/S=*/17/32 (instead of 0/0/0).
For this listing I'll assume that geometry.
Units = sectors of 512 bytes, counting from 0
```

Device	Boot	Start	End	#sectors	Id	System
/T8c/usbfd-64531026-rl-001.img1	*	32	121950	121919	4	FAT16 <32M
end: (c,h,s) expected (224,2,31) found (249,16,32)						
/T8c/usbfd-64531026-rl-001.img2		0	-	0	0	Empty
/T8c/usbfd-64531026-rl-001.img3		0	-	0	0	Empty
/T8c/usbfd-64531026-rl-001.img4		0	-	0	0	Empty

The unique allocated partition entry of 16 bytes (see figure 2.2, red square) was analyzed in detail:

- The 0x80 value denotes the partition is bootable (it would be 0x00 otherwise).

- The 0x04 value shows the partition type (FAT16 < 32MB).
- The 0x20000000 value is the partition starting sector number, sector 32 (0x20) <sup>1</sup>.
- The 0x3FDC0100 value is the partition length in sectors, that is, 121919.
- The 0x010100 value is the partition starting sector in C/H/S, that is, C=0, H=1, S=1 ([DISK1] - figure 17.11). First track is not used, 32 first sectors <sup>2</sup>.
- The 0x1020F9 value is the partition final sector in C/H/S, that is, C=249, H=16, S=32 ([DISK1] - figure 17.11).

The C/H/S values <sup>3</sup> fits in 10/8/6 bits respectively [DISK1]. The number of logical cylinders expected is 224 because the tools supposed a geometry of \*/17/32 (C/H/S), and dividing the total number of sectors, 121919, by 17 and by 32 you obtain 224, the tool's expected value <sup>4</sup> different from the real physical one, 249.

## E.2 Disk boot sector analysis

---

**From:** 27 Nov, 2004 (11:00) **To:** 27 Nov, 2004 (12:30)

---

As mentioned before, in a DOS partitioned disk, the boot sector is located in the first 446 bytes of sector 0 [DISK1]. This boot data was extracted (using dd) and its contents examined using the strings and sstrings commands (see section 2.6) and an hexadecimal editor, confirming it is similar to a regular DOS boot sector:

```
# dd if=usb_sector0 of=usb_bootloader bs=446 count=1
1+0 records in
1+0 records out
# ll usb_bootloader
-r----- 1 root root 446 Nov 27 11:53 usb_bootloader
# md5sum usb_bootloader
33a07a59d299ab4ea9f4ab0156f9d86f  usb_bootloader
# strings usb_bootloader
sQ0tN2
t+a'j
Invalid partition table
Error loading operating system
Missing operating system
# sstrings -e l usb_bootloader
#
```

Some tests were made but the same boot sector couldn't be found (WinXP SP1 and W2K SP4). A Google search was performed to look for the MD5 value but no result was obtained either.

This analysis raised up the idea of a new future forensic project, based on having a database of well-known boot sector hashes to verify the boot loader portion available in all

---

<sup>1</sup>All these values must be read backwards because they are represented in little-endian byte order (<http://www.netrino.com/Publications/Glossary/Endianness.html>). For example, from right to left, 0x00000020.

<sup>2</sup><http://www.ata-atapi.com/hiwtabs.htm>

<sup>3</sup>From the Linux fdisk man page.

<sup>4</sup><http://www.ata-atapi.com/hiwchs.htm>

bootable disks (and partitions); given the fact that the boot sector doesn't seem to change dynamically once the OS version and SP are known (and potentially, the program used to partition the disk). This would help to confirm in a reliable way that the first 446 bytes sector portion (the boot loader not including the partition table) is a well-known boot loader obtaining its MD5 value and looking if it matches the boot sector hash of other well known OS and disks boot loaders stored in the database.

The project should differentiate between disk and partition bootloaders. For example, the comparison between the USB flashdrive bootloader and the FAT16 partition bootloader was negative, they didn't match; even the strings they contain are different:

```
# dd if=usb_FAT16 of=usb_FAT16_sector0 bs=512 count=1
1+0 records in
1+0 records out
# dd if=usb_FAT16_sector0 of=usb_FAT16_bootloader bs=446 count=1
1+0 records in
1+0 records out
# md5sum usb_bootloader
33a07a59d299ab4ea9f4ab0156f9d86f  usb_bootloader
# md5sum usb_FAT16_bootloader
3ad1dffc6b85cba35be436cfa3aca7b3  usb_FAT16_bootloader
#
# strings usb_bootloader
sQ0tN2
t+a'j
Invalid partition table
Error loading operating system
Missing operating system
# strings usb_FAT16_bootloader
MSWIN4.1
NO NAME    FAT16    3
8N\${}
r>8-t
at=Nt
Invalid system disk
Disk I/O error
Replace the disk,
```

### E.3 Partition type change analysis

Due to the partition type inconsistencies found during the analysis (see chapter 2), a detailed analysis of the effects associated to the type of FAT16 partitions was performed (see figure E.1) over the simulation flashdrive of appendix D. A FAT partition of type 6 (FAT16) was change to 4 (FAT16 less than 32MB). As a conclusion, the partition type has no effects on the `fdisk` errors observed during the analysis, in the way the partition is mounted or accessed.

```
# fdisk -lu /dev/sda
...
  Device Boot      Start         End      Blocks   Id  System
/dev/sda1  *           32         127743       63856    6   FAT16
#
# sfdisk -l -V /dev/sda
...
  Device Boot  Start      End  #cyls   #blocks   Id  System
/dev/sda1  *        0+      498     499-     63856    6   FAT16
/dev/sda2            0        -         0         0    0   Empty
/dev/sda3            0        -         0         0    0   Empty
/dev/sda4            0        -         0         0    0   Empty
/dev/sda: OK
#
# fdisk /dev/sda

Command (m for help): p
...
  Device Boot      Start         End      Blocks   Id  System
/dev/sda1  *           1         499       63856    6   FAT16

Command (m for help): t
Selected partition 1
Hex code (type L to list codes): 4
Changed system type of partition 1 to 4 (FAT16 <32M)

Command (m for help): p
...
  Device Boot      Start         End      Blocks   Id  System
/dev/sda1  *           1         499       63856    4   FAT16 <32M

Command (m for help): w
The partition table has been altered!
Calling ioctl() to re-read partition table.
Syncing disks.
#
# fdisk -lu /dev/sda
...
  Device Boot      Start         End      Blocks   Id  System
/dev/sda1  *           32         127743       63856    4   FAT16 <32M
#
# sfdisk -l -V /dev/sda
...
  Device Boot  Start      End  #cyls   #blocks   Id  System
/dev/sda1  *        0+      498     499-     63856    4   FAT16 <32M
/dev/sda2            0        -         0         0    0   Empty
/dev/sda3            0        -         0         0    0   Empty
/dev/sda4            0        -         0         0    0   Empty
/dev/sda: OK
#
```

Figure E.1: Partition type change, from 4 to 6

# F USB FLASHDRIVE IMAGE EXTRACTION & METHODOLOGY

Due to the fact this is not a real case but the assignment of the GCFA certification, the initial evidence to be investigated is the image file obtained from the original USB flashdrive. The image was available from an specific Web page associated to this certification <sup>1</sup>.

---

**From:** 22 Nov, 2004 (10:30) **To:** 22 Nov, 2004 (15:30)

---

During the incident response phase of this case evidence seizure occurred. This time it was an easy task and consisted on the acquisition of non-volatile data in the form of a USB flashdrive, found in the suspect cubicle.

This incident was managed following the CC Terminals internal incident response policy, that specifies that once an incident response member collects an evidence it is transferred to a forensic investigator keeping the chain of custody in a proper way.

An image, bit-for-bit copy, of the original evidence was obtained following this steps <sup>2</sup>:

1. In order to avoid an accidental evidence modification/deletion, the write-protect switch available on this USB flashdrive model was turned on [LEXAR1].
2. The USB flashdrive was plugged into one of the USB ports of the forensic workstation (see appendix B), so it became the “/dev/sda” device:

```
# cat /proc/scsi/scsi
Attached devices:
Host: scsi0 Channel: 00 Id: 00 Lun: 00
  Vendor:          Model: Floppy-on-stick  Rev: 1.89
  Type:   Direct-Access                      ANSI SCSI revision: 02
```

3. A cryptographic hash (MD5 value) of the contents of the original USB flashdrive was obtained using the standard Linux md5sum command (v5.2.1) <sup>3</sup>. This value was verified against the one provided by Mark Mawer (see appendix A, figure A.1):

```
# md5sum /dev/sda
338ecf17b7fc85bbb2d5ae2bbc729dd5  /dev/sda
```

4. A raw image of the USB flashdrive contents was obtained using the standard Linux dd tool (v5.2.1) <sup>4</sup>:

```
# dd if=/dev/sda of=/T8c/usbfd-64531026-r1-001.img bs=512
121952+0 records in
121952+0 records out
```

---

<sup>1</sup><https://www.giac.org/GCFAPractical2.0-USBImageAndInfo.zip>

<sup>2</sup>All the steps followed during the forensic analysis try to ensure positive control over the evidence and the image, that is, that they are properly handled and their integrity is not altered.

<sup>3</sup>For all commands used in the paper, their software version has been identified, using the appropriate option, typically --version, -V or -v.

<sup>4</sup>Some tools, such as md5sum or dd, belong to the Linux coreutils package, having the same version.



5. A cryptographic hash (MD5 value) of the raw image was obtained using again the `md5sum` command, confirming that it was an exact bit-for-bit copy of the original USB flashdrive (both MD5 values match):

```
# md5sum /T8c/usbfd-64531026-rl-001.img
338ecf17b7fc85bbb2d5ae2bbc729dd5 /T8c/usbfd-64531026-rl-001.img
```

6. The image file permissions were modified to avoid an accidental modification and some other information related with this evidence was obtained using standard Linux tools, such as the image size or the disk serial number (see appendix A) <sup>5</sup>:

```
# chmod 400 /T8c/usbfd-64531026-rl-001.img
# ll /T8c/usbfd-64531026-rl-001.img
-r----- 1 root root 62439424 Nov 22 10:55 usbfd-64531026-rl-001.img
# cat /proc/scsi/usb-storage/0
Host scsi0: usb-storage
Vendor: USB
Product: Flash Disk
Serial Number: JDSP064-04-5000C
Protocol: Transparent SCSI
Transport: Bulk
Quirks:
```

After the image was obtained, a copy was made in a non-erasable media (CD-R) in case it must be reused, trying to avoid a damage in the original evidence disk due to the usage, maintaining its integrity <sup>6</sup>.

The original evidence (USB disk) was safely stored in a locked cabinet, together with CD-R disk and the chain of custody form, only accessible by the forensic investigator (see appendix A). The rest of the analysis will be based on the extracted image file.

## F.1 Methodology considerations & Reporting

The small size, 64 MB, and features of this case main evidence, the USB flashdrive, have allowed to follow the methodology sequentially, studying the image contents in each phase. When managing forensic cases involving huge amounts of information, such as multiples GB disks, heuristic methods (such as extensive strings searches in large allocated, slack or unallocated space) must be used in order to prioritize the key evidence pieces and their analysis. In this cases, the methodology showed in chapter 2 must be followed using a (cyclic) reiterative method based on the feedback obtained from the previous results. Besides, when working with large amounts of information, other file searching and automatic recovery methods must be used with tools like Foremost <sup>7</sup> or Lazarus <sup>8</sup>, not required in this case.

The in-depth technical report and findings were obtained and documented during the whole forensic analysis process, part of the methodology “Reporting” phase.

<sup>5</sup>The disk serial number for ATA/IDE devices in Linux can also be obtained using the `hdparm -i /dev/DISK` command. It is not the case for USB flashdrives because they are mapped to SCSI devices.

<sup>6</sup>The CD-R containing the image copy was not considered another evidence but an item derived from the original one, therefore, its details and existence have been included in a unique chain of custody form (see appendix A).

<sup>7</sup><http://foremost.sourceforge.net/>

<sup>8</sup><http://www.porcupine.org/forensics/tct.html>

---

**From:** 22 Nov, 2004 (10:30) **To:** 19 Dec, 2004 (23:30)

---

Based on the analysis results and all the evidences obtained, this whole report was obtained, including the findings and conclusions extracted from the information gathered, all them related with the suspect's actions and the relationships between all the different pieces of evidence found.

All the pieces found support the victim claim, and no single piece of evidence has been found that rejects it. Additionally, the recommended actions of chapter 7 would help to corroborate these facts and to extract complementary details, such as the e-mail address used by the suspect.

© SANS Institute 2005, Author retains full rights

# G COMPILATION OF WINPCAP AND WINDUMP

The source code of the WinPcap version used (3.1beta3) <sup>1</sup> and of the WinDump one (3.8beta1) <sup>2</sup> were downloaded and uncompressed to “C:\GCFA\_compile” in the compilation system (see below).

The free latest Cygwin version <sup>3</sup>, v1.5.12-1, was downloaded and installed in a fresh image of the VMWare binary analysis system (see appendix B). To the default Cygwin packages selection, the following ones were added: gcc (v3.3.1-3), flex (v2.5.4a-3), bison (v20030307-1) and make (v3.80-1). The Cygwin “bin” directory was configured in the system PATH.

WinPcap can be compiled for two different Windows platforms, Windows 9x or Windows NT (or greater). The later option was selected based on the results of the analysis of the OSes involved in this case (see chapter 2). Figure G.2 shows the steps followed to compile both, the library (two dynamic libraries files indeed) and the tool. In order to compile “wpcap.dll” a change, removing the “--nounistd” Flex option, was required in “C:\GCFA\_compile\winpcap\wpcap\PRJ\GNUmakefile” <sup>4</sup> (see figure G.1).

```
From:
.l.c:
@rm -f $@
$(LEX) --nounistd -t $< >$*.c
To:
.l.c:
@rm -f $@
$(LEX) -t $< >$*.c
```

Figure G.1: Flex option remove to compile WinPcap

The pro of using Cygwin is that it is freely available, however, the con is reflected in the WinPcap documentation itself:

*“Only the Packet and WPCAP DLLs can be built under gcc. The drivers themselves (packet.sys and packet.vxd) are too dependent on MSVC for their tools and assembler.”*

The WinDump compilation process started once the library was compiled (see figure G.3). Comparing the libraries compiled with the ones installed by the official WinPcap release during chapter 4, and the executable compiled with the one recovered from the USB flashdrive, it is confirmed that they are different (see figure G.4). The manually compiled libraries are much smaller than the official ones, while the executable is bigger.

Some of the reasons why the files didn’t match are explained in chapter 5. Additionally, inspecting the “GNUmakefile” of Windump it was confirmed that it is compiled using the “wpcap.dll” and “wsock32.dll” libraries, “LIBS = -L \$PCAP\_DIR/WPCAP/LIB -lwpcap -lwsock32”, so this second library, although the binary is dynamically compiled, could affect the end result. The same applies to the “wpcap.dll” library, it is based on the “packet.dll” and “wsock32.dll” libraries, “LIBS = -L ../.\$PACKET\_DIR/DLL/Project

<sup>1</sup>The official “wpcapsrc\_3\_1\_beta\_3.zip” URL was referenced in chapter 5.

<sup>2</sup>[http://windump.polito.it/install/bin/windump\\_3\\_8\\_3\\_beta/WDumpSrc.zip](http://windump.polito.it/install/bin/windump_3_8_3_beta/WDumpSrc.zip)

<sup>3</sup>A Linux-like environment for Windows.

<sup>4</sup>This option is not recognized by the Windows flex command.

```

C:\>cd GCFA_compile\winpcap\packetNtx\Dll\Project
C:\GCFA_compile\winpcap\packetNtx\Dll\Project>make
gcc -I ../../../../common -shared -mno-cygwin -O -D_WINNT4 -o ../Packet32.o -c ../
Packet32.c
...
C:\GCFA_compile\winpcap\packetNtx\Dll\Project>dir Packet.dll
...
12/15/2004 05:34p          42,283 Packet.dll
...
C:\GCFA_compile\winpcap\packetNtx\Dll\Project>md5deep Packet.dll
282e6875567d4278815a2517ae71db32 C:\GCFA_compile\winpcap\packetNtx\Dll\Project\Packet.dll

C:\>cd GCFA_compile\winpcap\wpcap\PRJ
C:\GCFA_compile\winpcap\wpcap\PRJ>make
gcc -I ../libpcap -I ../libpcap/bpf -I ../libpcap/lbl -I ../libpcap/Win32/Includ
e -I ../libpcap/Win32/Include/ipv6kit -I ../../common -I ../Win32-Extensions -DLI
BPCAP_EXPORTS -DYY_NEVER_INTERACTIVE -Dyylval=pcap_lval -DHAVE_STRERROR -DNEED_A
DDRINFO_H -DINET6 -DWIN32 -DSIZEOF_CHAR=1 -DSIZEOF_SHORT=2 -DSIZEOF_INT=4 -DWPCA
P -D'_U_=' -DHAVE_SNPRINTF -DHAVE_VSNPRINTF -mno-cygwin -shared -O -o ../libpcap
/bpf/net/bpf_filter.o -c ../libpcap/bpf/net/bpf_filter.c
...
C:\GCFA_compile\winpcap\wpcap\PRJ>dir wpcap.dll
...
12/15/2004 06:30p          181,507 wpcap.dll
...
C:\GCFA_compile\winpcap\wpcap\PRJ>md5deep wpcap.dll
9aaa73ad30118732ae8be91330d3b367 C:\GCFA_compile\winpcap\wpcap\PRJ\wpcap.dll

```

Figure G.2: Compilation of WinPcap in Windows 2000 SP4 using Cygwin

```

C:\>cd GCFA_compile\tcpdump\win32\prj
C:\GCFA_compile\tcpdump\win32\prj>make
gcc -I ../../../../winpcap/wpcap/libpcap/bpf -I ../../../../winpcap/wpcap/libpcap -I .
../../../../winpcap/wpcap/libpcap/Win32/Include -I ../../../../winpcap/wpcap/libpcap/Wi
n32/Include/net -I ../../Win32/Include -I ../../linux-Include -I ../../lbl -I..
.. -DWIN32 -DHAVE_ADDRINFO_H -DHAVE_PCAP_FINDALLDEVS -DINET6 -DSIZEOF_CHAR=1 -DH
AVE_SOCKETADDR_STORAGE -DHAVE_PCAP_DUMP_FLUSH -DHAVE_REMOTE -DHAVE_PCAP_LIB_VERSIO
N -DSIZEOF_SHORT=2 -DSIZEOF_INT=4 -D_U_="__attribute__((unused))" -mno-cygwin -O
-o ../../addrtoname.o -c ../../addrtoname.c
...
C:\GCFA_compile\tcpdump\win32\prj>dir windump.exe
...
12/15/2004 06:40p          519,152 windump.exe
...
C:\GCFA_compile\tcpdump\win32\prj>md5deep windump.exe
b123d41be335e9d902e86a752eb39c2e C:\GCFA_compile\tcpdump\win32\prj\windump.exe

```

Figure G.3: Compilation of WinDump in Windows 2000 SP4 using Cygwin

-lPacket -lwsck32". Therefore, the Windows OS version used to compile the program could also affect the result. In this case, the "wsck32.dll" file used was version 5.0.2195.6603 (W2K SP4)<sup>5</sup>.

Another behaviour that was identified during the verification tests was that the official WinPcap library is capable of linking the sniffing tools, like Windump, against the "GenericNdisWanAdapter" interface, while the default compiled version doesn't recognize it<sup>6</sup>. This is another reason why obtaining the same MD5 value without knowing the exact compilation options and environment used in the official release is very difficult:

- Using the compiled WinPcap version (packet.dll and wpcap.dll):

<sup>5</sup>Version was obtained checking the "Properties" of the file, located in "C:\Winnt\system32".

<sup>6</sup>This can be tested placing the compiled WinPcap files (.dll) in the tool current directory.

```
C:\GCFA>dir WinDump.exe
...
11/27/2004  01:20p                450,560 WinDump.exe
C:\GCFA>md5deep WinDump.exe
79375b77975aa53a1b0507496107bff7  C:\GCFA\WinDump.exe

C:\WINNT\system32>dir wpcap.dll packet.dll
...
05/14/2004  01:02p                225,280 wpcap.dll
...
05/14/2004  11:30a                81,920 packet.dll
C:\WINNT\system32>md5deep wpcap.dll packet.dll
5c5561185a8751711156934585f002e8  C:\WINNT\system32\wpcap.dll
28f57308b067836d715aa070d29c3232  C:\WINNT\system32\packet.dll
```

Figure G.4: Official compiled versions of WinPcap and WinDump

```
C:\GCFA_compile\>windump -D
1.\Device\NPF_{5A2B252B-0A24-4014-A92A-1896B52A418D} (AMD PCNET Family Ethernet Adapter)

- Using the official WinPcap version:
C:\GCFA_compile\tcpdump\win32\prj>windump -D
1.\Device\NPF_GenericNdisWanAdapter (Generic NdisWan adapter)
2.\Device\NPF_{5A2B252B-0A24-4014-A92A-1896B52A418D} (AMD PCNET Family Ethernet Adapter)
```

# H MD5 VALUE OF ALL ANALYSIS FILES

This appendix contains the MD5 values of all the intermediate files used and generated during the whole forensic analysis process. This list can be used as a validation tool to check the integrity of all the analysis intermediate steps <sup>1</sup>. See appendix B for the directories used during the analysis and its purpose.

## T8c\_image:

```
# md5sum *
33a07a59d299ab4ea9f4ab0156f9d86f  usb_bootloader
5f830a763e2144483f78113a8844ad52  usb_FAT16
ec2ca341a1e4687e63989e6d994419fe  usb_FAT16.slack
338ecf17b7fc85bbb2d5ae2bbc729dd5  usbfd-64531026-rl-001.img
bf619eac0cdf3f68d496ea9344137e8b  usb_missing
5bf1cea807dec8655ed18b9bbf2ee918  usb_sector0
51596dda30fc38f0df3556d6f115256d  usb_unallocated
```

## T8c\_image/files:

```
# md5sum *
2097b7b0a9fedb4238b67e976c4ae1cb  capture
a833c58689596eda15a27c931e0c76d1  coffee.doc
9785a777c5286738f9deb73d8bc57978  her.doc
ca601d4f8138717dca4de07a8ec19ed1  hey.doc
9bc3923cf8e72fd405d7cea8c8781011  map.gif
79375b77975aa53a1b0507496107bff7  WinDump.exe
cdfcf8565e622daf838b8f5c692eb11b  WinPcap_3_1_beta3_fragment.exe
```

## T8c\_image/timeline:

```
# md5sum *
ec37d516b8b13748b4210ff6bc668352  body
2d2f586b898d914ee354792931e791e8  body_fls
89b7d2eb551c5c8c5a32befe0092c2ce  body_ils
1d5473df1bc6153c77cbc585d8ec64d5  mactime_fls.txt
98b8769fd04aa471b548612929f7c194  mactime_ils.txt
22193dff6c66c4d62e135c30b97cb2682  mactime.txt
```

## T8c\_image/slack:

```
# md5sum *
aae15c5605d8708dbd7e3fef53b50554  images-usb_FAT16-Sector550.raw
bf619eac0cdf3f68d496ea9344137e8b  images-usb_FAT16-Sector590.raw
fc930a486a89ad02091e64870435e825  images-usb_FAT16-Sector630.raw
```

## T8c\_binaries:

```
# md5sum *
0c805905fb1faa2b5c553d0379797cda  WDumpSrc_3.8.3beta.zip
79375b77975aa53a1b0507496107bff7  WinDump_3.8.3beta.exe
11e090da8cd414bd0267e40beae86f5b  WinPcap_3_0.exe
4511ee3b4e5d8150c035a140dfba72c0  WinPcap_3_1_beta_3.exe
cdfcf8565e622daf838b8f5c692eb11b  WinPcap_3_1_beta_3_fragment_official.exe
```

<sup>1</sup>For the most important files, a screen shot was captured during the analysis.



```
70553e1c186284f46b46c9521bba629b WinPcap_3_1_beta_3_fragment_official_truncated.exe
fc930a486a89ad02091e64870435e825 WinPcap_3_1_beta_3_sector40.exe
5de823defbc70e2b1f1989f96bacb583 WinPcap_3_1_beta4.exe
01b5033a0fb830e44e44938037cd48e2 wpcapsrc_3_0.zip
d209e1a6a069c53460fbf57f1e9e78d8 wpcapsrc_3_1_beta_3.zip
9e3c3c46fbb1a048c09e4f0fd77e975a wpcapsrc_3_1_beta4.zip
```

/opt/EvidenceLocker/GCFA-v2.0-Option1-11-2004/LexarUSB/images:

# md5sum \*

```
4ba0abdb836f48f340447a9d8a348155 md5.txt
5f830a763e2144483f78113a8844ad52 usb_FAT16
```

/opt/EvidenceLocker/GCFA-v2.0-Option1-11-2004/LexarUSB/output:

# md5sum \*

```
472fd6cb5c66819111b9c310107b0b95 body
c8ca2c268131c13fdc135c651054238a md5.txt
1367339ff3d07b593f886c9f9caff99f timeline.txt
f0e36e7b63a653819232cb03e56d8be9 timeline.txt.sum
a53b5443fdce5bb5d0f22f9f373cde2c usb_FAT16.asc
879c61833ac255b6396d737ea73d96d7 usb_FAT16.unalloc
fffaf1623fc23cd61939d7144afe4d74 usb_FAT16.unalloc.asc
32335508658e34ec8ee98ae0efe53d97 usb_FAT16.unalloc.rest
b11ef8c5c21ee813a5bc9d91ef3cc3a2 usb_FAT16.unalloc.uni
f61b6c2ca915205396a2d184b873c381 usb_FAT16.uni
```

# Bibliography

- [ADDI1] "The Role of Computer Forensics in Stopping Executive Fraud". Scott Laliberte, Ajay Gupta. Addison-Wesley. <http://www.informit.com/articles/printerfriendly.asp?p=336258> (October 2004)
- [ALEJ1] "El uso legítimo del correo electrónico". Alejandra Castro Bonilla. <http://delitosinformaticos.com/delitos/correo.shtml> (December 2002)
- [ANTI1] "AntiSniff". v.1.0.21. L0pht. <http://www.securityfocus.com/tools/1004> (December 2004)
- [ARPS1] "Real World ARP Spoofing". Raúl Siles Peláez. August 2003. [http://www.giac.org/practical/GCIH/Raul\\_Siles\\_GCIH.pdf](http://www.giac.org/practical/GCIH/Raul_Siles_GCIH.pdf) (November 2003)
- [ART50] "Artículo 50.1 - Código Penal". C.P. 1995. [http://www.igsap.map.es/cia/dispo/7734\\_2.htm#sec4cap1tit3](http://www.igsap.map.es/cia/dispo/7734_2.htm#sec4cap1tit3) (November 2004)
- [AUTO1] "Autopsy (The Sleuth Kit)". Brian Carrier. <http://www.sleuthkit.org/autopsy/index.php> (November 2004)
- [BIN1] "BinText". v3.00. Foundstone. <http://www.foundstone.com/resources/termsofuse.htm?file=bintext.zip> (November 2004)
- [CP1] "Código Penal (C.P) Español". 1995. <http://www.igsap.map.es/cia/dispo/7734.htm>, <http://www.delitosinformaticos.com/delitos/codigopenal.shtml> (December 2004)
- [CUER1] "Delitos informáticos: Protección Penal de la Intimidad". Jose Cuervo. <http://www.informatica-juridica.com/trabajos/delitos.asp> (December 2004)
- [DISK1] "Chapter 17 - Disk and File System Basics". <http://www.microsoft.com/resources/documentation/windowsnt/4/workstation/reskit/en-us/diskdesc.msp> (November 2004)
- [ESTA1] "Estatuto de los Trabajadores". Real Decreto Legislativo 1/1995, de 24 de marzo. <http://www.unex.es/gerencia/et.htm> (November 2004)
- [ETHE1] "Ethereal". <http://www.ethereal.com/> (November 2004)
- [EU1] "Convention on Cybercrime". Budapest, 23.XI.2001. Council of Europe (COE). [http://www.coe.int/T/E/Legal\\_affairs/Legal\\_co-operation/Combating\\_economic\\_crime/Cybercrime/](http://www.coe.int/T/E/Legal_affairs/Legal_co-operation/Combating_economic_crime/Cybercrime/), <http://www.coe.int/T/E/Com/Files/Themes/Cybercrime/default.asp> (December 2004)

- [FAT1] "The FAT File System. Sleuth Kit Implementation Notes (SKINs)". Brian Carrier. [http://www.sleuthkit.org/sleuthkit/docs/skins\\_fat.html](http://www.sleuthkit.org/sleuthkit/docs/skins_fat.html) (November 2004)
- [FILEM1] "FileMon". v4.34. SysInternals. <http://www.sysinternals.com/ntw2k/source/filemon.shtml> (September 2004)
- [HONEY1] "Honeyd". Neil Provos. <http://www.honeyd.org>, <http://www.citi.umich.edu/u/provos/honeyd/> (August 2004)
- [HONEY2] "The Honeyney Project". <http://www.honeynet.org> (August 2004)
- [INFOR16] "The Sleuth Kit Informer. Issue 16. sstrings and Unicode searching". Brian Carrier. <http://www.sleuthkit.org/informer/sleuthkit-informer-16.html> (September 2004)
- [KAMP1] "An Introduction to DOS FAT Volume and File Structure". Mark Kampe. <http://www.seas.ucla.edu/classes/mkampe/cs111.fq04/docs/dos.html> (November 2004)
- [LEXAR1] "Lexar Media 64 MB JumpDrive Portable USB Flash Drive". <http://www.lexar.com/jumpdrive/jd.html> (November 2004)
- [LIST1] "ListDLLs". v2.23. Sysinternals. <http://www.sysinternals.com/ntw2k/freeware/listdlls.shtml> (November 2004)
- [MD5D1] "md5deep". v1.5. Kornblum, J. <http://md5deep.sourceforge.net/> (October 2004)
- [MSDOS1] "MS-DOS Partition Summary". Microsoft. <http://support.microsoft.com/kb/69912/EN-US/> (November 2004)
- [MSFAT1] "MS Extensible Firmware Initiative FAT32 File System Specification". rev. 1.03, December 6, 2000. Microsoft. <http://www.microsoft.com/whdc/system/platform/firmware/fatgen.msp> (November 2004)
- [MURR1] "Digging Covert Channels". Michael Murr. GCFA v1.2. [http://www.giac.org/practical/GCFA/Michael\\_Murr\\_GCFA.pdf](http://www.giac.org/practical/GCFA/Michael_Murr_GCFA.pdf) (June 2003)
- [NETC1] "Netcat (nc)". [http://www.atstake.com/research/tools/network\\_utilities/](http://www.atstake.com/research/tools/network_utilities/), <http://netcat.sourceforge.net> (September 2003)
- [P0F1] "The new P0f v2". Michal Zelwski. <http://lcamtuf.coredump.cx/p0f.shtml> (December 2004)
- [PARTA1] "Minimal partition table specification". Andries Brouwer. [http://www.win.tue.nl/~aeb/partitions/partition\\_tables.html](http://www.win.tue.nl/~aeb/partitions/partition_tables.html) (October 2004)
- [PARTY1] "Partition Types". Andries Brouwer. [http://www.win.tue.nl/~aeb/partitions/partition\\_types.html](http://www.win.tue.nl/~aeb/partitions/partition_types.html) (October 2004)
- [PORTS1] "IANA Assigned port numbers". <http://www.iana.org/assignments/port-numbers> (August 2004)
- [PROCE1] "Process Explorer". v.5.23. SysInternals. <http://www.sysinternals.com/ntw2k/freeware/procexp.shtml> (September 2004)

- [PROD1] "*proDETECT*". Egemen Tas. <http://sourceforge.net/projects/prodetect/> (December 2004)
- [PROM1] "*PromiScan*". Security Friday. <http://www.securityfriday.com/products/promiscan.html> (December 2004)
- [REGM1] "*RegMon*". v4.34. SysInternals. <http://www.sysinternals.com/ntw2k/source/regmon.shtml> (September 2004)
- [REGS1] "*RegShot*". v1.61e1. <http://regshot.yeah.net> (September 2004)
- [SANA1] "*Detection of Promiscuous Nodes using ARP packets*". Daiji Sanai. [http://www.securityfriday.com/promiscuous\\_detection\\_01.pdf](http://www.securityfriday.com/promiscuous_detection_01.pdf) (August 2001)
- [SLEU1] "*The Sleuth Kit*". Brian Carrier. <http://www.sleuthkit.org/sleuthkit/index.php> (November 2004)
- [SNORT1] "*Snort Network IDS*". <http://www.snort.org> (May 2004)
- [SPY1] "*The espionage crime using computer devices*". Portalley.com. <http://www.delitosinformaticos.com/delitos/espionaje.shtml> (October 2003)
- [TCPD1] "*Tcpdump*". <http://www.tcpdump.org> (May 2004)
- [TDIM1] "*TDIMon*". v1.0. SysInternals. <http://www.sysinternals.com/ntw2k/freeware/tdimon.shtml> (September 2004)
- [TIME1] "*File Activity Timelines*". Brian Carrier. [http://www.sleuthkit.org/sleuthkit/docs/ref\\_timeline.html](http://www.sleuthkit.org/sleuthkit/docs/ref_timeline.html) (November 2004)
- [TIME2] "*What time is it?*". Mares and Company. <http://www.dmares.com/maresware/articles/filetimes.htm> (November 2004)
- [TIT10] "*Título X. Capítulo I. Del descubrimiento y revelación de secretos*". Código Penal Español. [http://www.igsap.map.es/cia/dispo/7734\\_3.htm#cap1tit10](http://www.igsap.map.es/cia/dispo/7734_3.htm#cap1tit10) (October 1995)
- [URL1] "*URL encoding tables*". <http://www.blooberry.com/indexdot/html/topics/urlencoding.htm>, [http://www.w3schools.com/html/html\\_ref\\_urlencode.asp](http://www.w3schools.com/html/html_ref_urlencode.asp) (November 2004)