CC Terminals –Harassment Case
A GIAC GCFA Practical Submission

Dean Farrington
GCFA Practical Version 2.0
Option 1
3/5/2005

Summary

This paper details the forensic analysis of a 64MB USB drive conducted on behalf of the company CC Terminals. In the paper we will cover the steps taken to preserve the integrity of the evidence, the duplication of the original drive, and the detailed analysis steps. It will conclude with a presentation of the evidence, a brief description of the legal implications and some recommendations for follow up.

Table of Contents

Executive Summary

Leila Conlay contacted corporate security for CC Terminals reporting that she was being harassed by a fellow employee named Robert Lawrence. Corporate security launched an investigation, during the process an after hours search of Mr. Lawrence's cubicle was conducted and a USB Flash drive, which is a small storage device that can fit in your pocket, was seized. This drive was subjected to computer forensic analysis to determine the contents and how and when they where used.

The drive contained 3 Microsoft Word Document files that held messages of increasingly strong tone and culminating with an indirect threat. The drive also contained several files that had been deleted. These where recovered and examined as well. Files on a drive are like books in a library, and the File Allocation Table is like the card catalog. When files are deleted from a drive the process is similar to removing the card from the catalog, but not removing the book from the shelf. It is still there, only the pointer to it has been removed. To get the deleted files back we looked at the location where the file had been, which was still recorded in the File Allocation Table and copied them to a new place for examination.

Among the recovered files was a tool for capturing raw traffic from a computer network and an output file from that same tool. The output file contained a full e-mail message being set through the popular web based e-mail provider Hotmail from Miss Conlay to another individual. This message gave a location and time where they where to meet. The final file that was recovered was an image file of a Microsoft MapPoint map. MapPoint is a tool for generating street maps. The image file recovered specifically identified the street corner where Miss Conlay had arranged to meet the person in the e-mail. The captured e-mail message not only gave the time and place for the meeting but also disclosed Miss Conlay's personal e-mail address for the suspect's use.

The presence of the captured e-mail session is a violation of the Federal Wiretap statute, 18 U.S.C §§ 2510-22 which is also referred to as Title III of the Omnibus Crime Control and Safe Streets Act of 1968. This law makes it a crime to capture network traffic without authorization unless you fall into one of a handful of exceptions. Mr. Lawrence's job does not encompass any of those exceptions.

Examination Details

I received custody of a 64MB Lexar Media USB jump drive enclosed in an evidence bag from Mark Mawer who is the Security Administrator for CC Terminals. I signed for custody of this drive on the chain of custody form to maintain the proper ability to audit who has had access to an item which is potentially evidence.

Evidence Tag Front

Tag: USBFD-64531026-RL-001	Date: 10/29/2004
Description: 64M Lexar Media Jump Drive	Collected by: Mark Mawer
Serial: JDSP064-04-5000C	
Image: USBFD-64531026-RL-001.img	Case #: **CCT10292004**
MD5: 338ECF17b7fc85bbb2d5ae2bbc729dd5	

Evidence Tag Rear

Chain of Custory			
From: Mark Mawer Location: Evidence Locker	Date: 11/1/04	Reason: Forensic Duplication of potential evidence	To: Dean Farrington Location: Forensic Examination Room
From: Location:	Date:	Reason:	To: Location
From: Location:	Date:	Reason:	To: Location
From: Location:	Date: 11/1/04	Reason:	To: Location

I take the drive to the Forensic Examination room, which is a secure access location. No one except the forensic examiners can enter this room to prevent contamination or loss of control of evidence. To prevent alteration of the original evidence during the investigation the drive will be imaged on a Forensics Workstation running Fedora Core 2. This will allow the examination to be conducted on an exact duplicate of the evidence while preserving the integrity of the original evidence.

Before I can perform the actual imaging, a change needs to be made to the

Forensics Workstation. USB technology was designed to be extremely “User Friendly”. When a USB drive of this type is connected to the system, Fedora Core 2 will attempt to mount it automatically. This will result in it being mounted in a forensically unsound manner, in other words in a state where it can be written to. I will need to modify the /etc/fstab file on the Forensics Workstation to mount the drive in read only mode. By default on Fedora Core 2 there is no entry for the USB drive in the /etc/fstab file. When you insert a USB drive the system detects the hardware change and an entry is dynamically created. To prevent this from occurring we will edit /etc/fstab and add our own entry.

USB drives are commonly recognized by Linux as SCSI Devices, since there are no SCSI devices in the forensic workstation it should be mounted as /dev/sda1. The “sd” in /dev/sda stands for SCSI Device. Just to be certain this was how the device would be mounted I inserted another USB drive (that was not evidence in any investigation) and checked the contents of /etc/fstab after the system automatically mounted it. It showed the drive mounted in this manner:

```
/dev/sda1    /mnt/flash   auto   noauto,owner,kudzu 0 0
```

I unmounted the drive and used the text editor vi to edit the /etc/fstab file by inserting the following at the end of the existing entries:

```
/dev/sda1  /mnt/flash  auto ro,noauto,users 0 0
```

This string causes the drive to be mounted read only, so that no unintentional changes can be made to the file system. It also specifies that any user can mount the removable device. Since this is a controlled workstation it makes sense that any investigator should be able to mount USB Drives on this system. I saved my change and created the directory /mnt/flash to be the mounting point for the USB drive.

I reinserted my test USB drive and issued the command

```
[deanf@forensicsworkstation]$mount /mnt/flash
```

The drive mounted and appeared in the file system. I then attempted to write a file to it in order to test the state of the drive. I received an error indicating that the system was unable to mount the drive as it was mounted “Read Only”. This was the result I wanted. I unmounted the test drive and prepared to image the evidence.

I took the original evidence, specifically the USB drive, and a log book for making my case notes. Making all entries in ink to ensure my notes can be properly used as evidence if required in the future, I noted my internal case number (CCT10292004), the date, and the serial number of the evidence drive a the top of the sheet. I then wrote the time I began the imaging process in the left hand margin and mounted the drive. All following steps are noted in the log book with their corresponding timestamps, I will not make further reference to the recording of these notes unless there is a specific point to make regarding the recordkeeping.

To mount the drive I used the command:

```
[deanf@forensicsworkstation]$mount /mnt/flash
```

I will first capture an MD5 hash of the evidence drive to ensure it was not modified prior to my imaging it. A hash is also sometimes known as a checksum or a fingerprint, it is basically a digital fingerprint of a file or drive. To obtain a hash an algorithm is applied to a file or drive, the resulting value is unique to that file or drive. MD5 is the commonly accepted hashing function for forensic work and is widely used today. MD5 is a 128 bit hash function create by Ron Rivest of MIT (and the "R" in RSA) in 1992, it was published as RFC 1321[1].

I am using the tool md5sum to generate my hashes. This tool is commonly available on Linux and MD5 implementations are readily downloadable for Windows[2].

I start by taking a hash of the evidence that I just mounted:

```
deanf@forensicsworkstation $md5sum /dev/sda1
338ECF17b7fc85bbb2d5ae2bbc729dd5   /dev/sda1
```

I compared the hash I obtained with the value on the evidence tag attached to the drive I received from Mark Mawer. Since the hashes match this allows me to conclude that the evidence I received has not been altered since Mark Mawer recorded the hash initially. Since I did not personally collect this evidence, the weakest link will be the evidence collection, handling, and documentation process used by Mr. Mawer. Any attack on the soundness of the evidence would have to be directed at the time between the initial collection and the establishment of the hash that is recorded on the evidence tag.

Now I need to create a Bit image copy of the USB drive which will give me a forensic duplicate. The forensic duplicate contains every bit of information contained in the source drive. The physical drive is considered "Best Evidence" according to the Federal Rules of Evidence (FRE) §1002, to preserve the integrity of the evidence, only the bare minimum amount of work is ever done on the original (evidence) drive. We will use the image we produce as our working copy of the drive for our analysis. FRE §1003, Admissibility of Duplicates[3] states that "A duplicate is admissible to the same extent as an original unless (1) a genuine question is raised as to the authenticity of the original or (2) in the

[1] http://www.faqs.org/rfcs/rfc1321.html
[2] http://users.erols.com/gmgarner/forensics/ or http://www.fourmilab.ch/md5/
[3] http://www.law.cornell.edu/rules/fre/rules.htm

circumstances it would be unfair to admit the duplicate in lieu of the original."

To capture the image I will use the *nix backup tool "dd"[4], or data dumper, which produces a bit-stream backup image. This will minimize the chance of a challenge based on FRE §1003, such a challenge would likely be based on a claim that the image was not exact or complete. We will actually use dd twice, this first time we will be making the Bit-Stream Image and later we will also be using dd to break out the individual partition(s) we find on the drive.

First we image the entire drive:

```
dd if=/dev/sda1 of=/forensics/USBFD-64531026-RL-001.img
```

This causes dd to use /dev/sda1 as the input file(if=) and /forensics/USBFD-64531026-RL-001.img as the output file(of=) for the image. As soon as the image is created we make a MD5 hash of the image file (copy) to see that it matches the MD5 of the actual evidence drive (original). If the hash matches the copy is identical.

```
deanf@forensicsworkstation $md5sum USBFD-64531026-RL-001.img
338ECF17b7fc85bbb2d5ae2bbc729dd5  USBFD-64531026-RL-001.img
```

I could also have created the image using dcfldd which would allow me to make the image, display a running status bar and create the MD5 hash automatically upon completion of the dd image. Dcfldd was originally created by the Department of Defense Computer Forensic Laboratory[5] Its source code is now available

Since the hash matches, we have good copy and can proceed. I copy the MD5 hash into a text file and then burn the image file and the file containing the hash onto 3 CD-R disks. CD-R is chosen over CD-RW since it is write once media. It cannot hold artifacts from files previously stored there. I label each disk with the case number and the date. 2 copies of the CD-R are placed into individual evidence bags with a chain of custody card attached and filled out. These copies of the disk are for later needs and are returned to the evidence safe along with the actual USB drive that is our original evidence. From this point forward we are going to operate only on the copy of the drive we created.

First we need to determine where the partition is located in the image we have crated. We will do this using the tool mmls which is a part of the sleuthkit[6]. The mmls tool will read the partition table in the image file and display the partitions for us. Based on the suspects position in the company and the common uses of USB drives, I suspect this drive will contain a windows file system. Since we are

[4] http://www.crazytrain.com/dd.html
[5] http://www.dcfl.gov/dcfl/dcfl.htm
[6] http://www.sleuthkit.org/

expecting to be dealing with a Windows partition we will begin by asking mmls to identify a dos partition table using the –t flag.

```
[deanf@LinuxForensics CCTerminals]$ mmls -t dos 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)
[deanf@LinuxForensics CCTerminals]$
```

Mmls has told us several interesting things in this output, we now know the default block size is 512 bytes, there is a Dos FAT16 partition, and a small amount of unallocated space on this drive.

Now we will use dd again to separate out the partitions for examination. This time we will be using some additional options of the dd command in order to specify exactly which part of the complete image we wish to carve off. In addition to the Input and output files we will be specifying the block size (bs=) which mmls gave us, using the values called skip and count. Skip tells dd where to begin copying and count tells dd where to stop copying. We will be using the values from mmls to carve off the DOS Fat16 partition first Mmls told us that partition began at 0000000032 and ended at 0000121950. Knowing this we can determine our skip and count values. Our skip value is 32 since this is the starting offset for the DOS partition, to get our count value we take the end value which 121950 and subtract 32 from it to arrive at a count value of 121918.

```
[deanf@LinuxForensics]$ dd if=USBFD-64531026-RL-001.img bs=512 skip=32
count=121918 of=part1.dd
121918+0 records in
121918+0 records out
[deanf@LinuxForensics]$md5sum part1.dd
95680b1e67eaa94bcc023a105aa1da15  part1.dd
```

Once the dd was complete I created an MD5 of the new image as well. This new hash is different than the hash of the full drive, it will be important to demonstrate I make no changes to the newly extracted partition.

Next I also carved out the unallocated space for examination as well.

```
[deanf@LinuxForensics]$ dd if=USBFD-64531026-RL-001.img bs=512 skip=1 count=31
of=part2.dd
31+0 records in
31+0 records out
[deanf@LinuxForensics]$ md5sum part2.dd
51596dda30fc38f0df3556d6f115256d  part2.dd
```

Once I had the partitions, the first thing I needed to do was establish my timeline of file accesses. If we use the analogy of a library, every file on a drive would be a book. In a library there is a card catalog that contains information about every book in the building along with a pointer to where that book is located. On a device with a FAT file system the equivalent of that card catalog is the File Allocation Table (FAT). The FAT contains information about every file in the file system as well as the pointer to where that file is located. Some of the information about each file contained in the FAT is the MAC time. MAC stands for Modified, Accessed, and Created times. These three attributes tell you when a file was last read, was last written, or was last accessed. Creating a timeline can be very important in conducting a forensic examination as it can help establish the order in which events occurred (such as during a hacking incident), the order files have been accessed, or provide a pointer to files that have been changed without the knowledge of the system owner or Admin. Since the FAT files system does not support file level access controls like NTFS we will not be able to gather information on permissions, users, and groups.

I used the tool fls from the sleuthkit to write the file and directory entries to a file. The fls tool works at the file name layer, it reads the inode value of a directory and processes the contents in order to display the files including any deleted files that were in the directory. The following command tells fls to recursively examine the image (-r), output the results in mactime format (-m), and append the directory e: to each file name.

```
[deanf@LinuxForensics CCTerminals]$ fls -f fat16 -m e: -r part1.dd >
part1.fls
```

This data will be augmented with the output of the command ils (Inode lister) which will extract additional data on deleted files. The ils tool works at the Meta Data layer.

```
[deanf@LinuxForensics CCTerminals]$ ils -f fat16 -m part1.dd >
part1.ils
```

Once I have obtained the output of both fls and ils I need to combined these into a single file. If I had been investigating a complete operating system I might have to do this for multiple partitions, but in this case I have only 2 files. I combined these files using the Linux cat (concatenate) command to create a single file.

```
[deanf@LinuxForensics CCTerminals]$cat part1.?ls >part1.mac
```

Once I have my single file I use the mactime tool to sort the entries in the file based on MAC timestamp

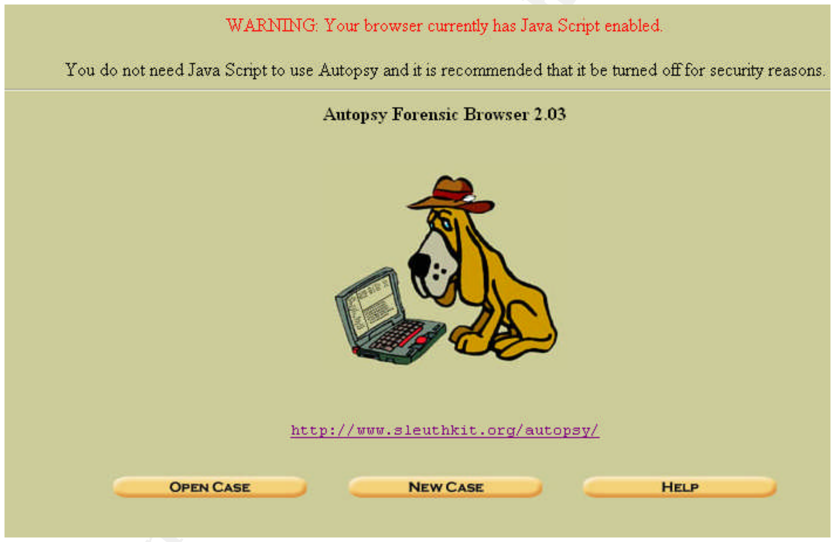
```
[deanf@LinuxForensics CCTerminals]$ mactime -b part1.mac > part1.all
```

My resulting timeline of activity on this USB drive is included as Appendix A.
A short review of the timeline reveals only a few files are present on the drive,

several files that where stored on the drive have been deleted.

I decide to mount the drive image using Autopsy to speed my investigation. Autopsy[7] is a graphical interface for the tools in the SleuthKit[8] both of which were created by Brian Carrier. Some might argue against the use of "point and click" forensics, but in this instance it will speed my initial investigation and I can return and redo any results that warrant it by hand. Autopsy is automating the use of the tools contained in the Sleuth Kit so I can validate Autopsy results by executing the Sleuth Kit tools manually.

I launch Autopsy on my forensics workstation; this opens the Autopsy interface inside a browser window. The first step I need to perform in order to configure the environment for my investigation is to create a new case within Autopsy. I select the "new case" button and provide a Case Name, a Case Description, and list myself as the investigator.


WARNING: Your browser currently has Java Script enabled.

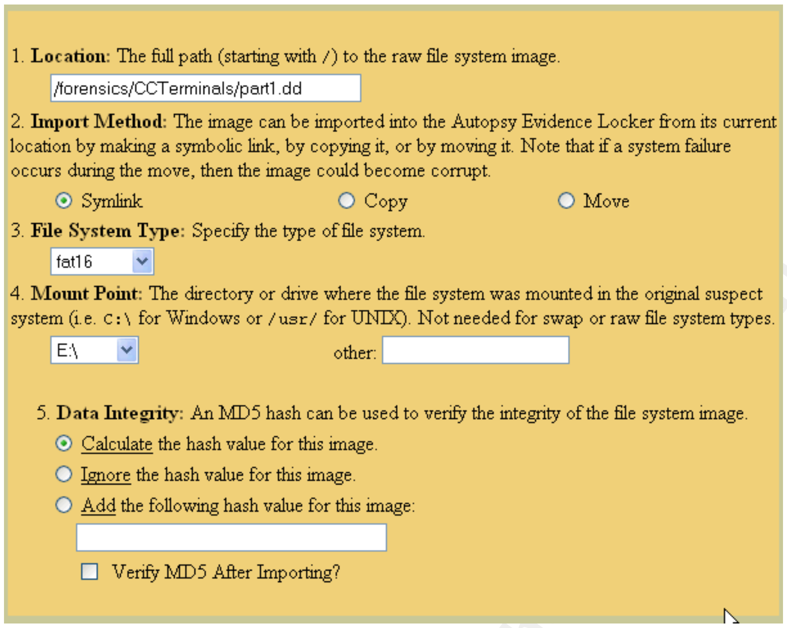
You do not need Java Script to use Autopsy and it is recommended that it be turned off for security reasons.

Autopsy Forensic Browser 2.03

http://www.sleuthkit.org/autopsy/

Once my case is created I need to configure a host. This is normally the Computer being examined, but in this case it is simply the description of the USB drive. I provide the name, description (I use the drive's serial number), and the time zone. Finally I need to add the partitions to the case so we can begin the examination. To do this I provide the path to the partition image. I select the option to make a symbolic link to the location of the existing part1.dd and part2.dd files. I specify the file system type (fat16 for the pat1.dd file and raw for the part2.dd file which was the unallocated space). I choose a mount point of E:\ for the part1.dd image since that drive letter was not in use on the VMware image I use for forensics. I select to have the MD5 sums calculated for the image for integrity as well.

[7] http://www.sleuthkit.org/autopsy/index.php
[8] http://www.sleuthkit.org/sleuthkit/index.php


1. **Location:** The full path (starting with /) to the raw file system image.

/forensics/CCTerminals/part1.dd

2. **Import Method:** The image can be imported into the Autopsy Evidence Locker from its current location by making a symbolic link, by copying it, or by moving it. Note that if a system failure occurs during the move, then the image could become corrupt.

[✓ Symlink] Copy Move

3. **File System Type:** Specify the type of file system.

fat16

4. **Mount Point:** The directory or drive where the file system was mounted in the original suspect system (i.e. `C:\` for Windows or `/usr/` for UNIX). Not needed for swap or raw file system types.

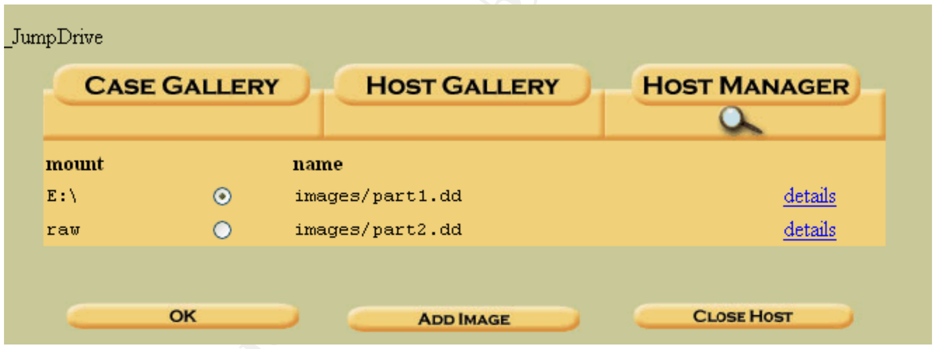
E:\ other:

5. **Data Integrity:** An MD5 hash can be used to verify the integrity of the file system image.

[✓ Calculate the hash value for this image.]
Ignore the hash value for this image.
Add the following hash value for this image:

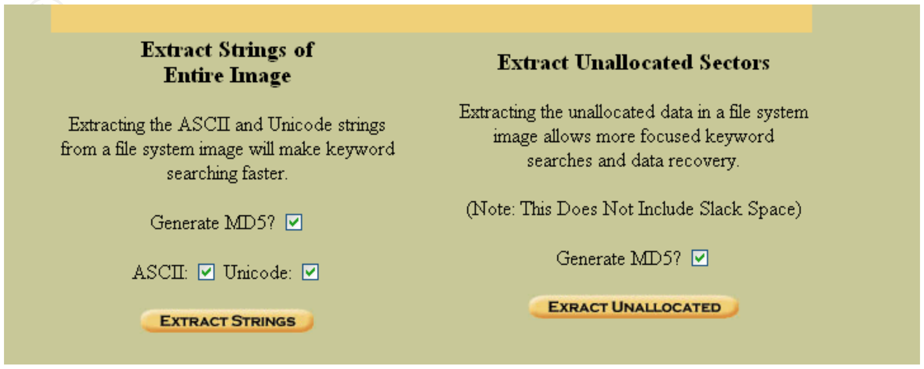
☐ Verify MD5 After Importing?

Once the image files have been imported I select the details link for the E:\ drive.


_JumpDrive

mount		name	
E:\	●	images/part1.dd	details
raw	○	images/part2.dd	details

This presents me with options to extract the strings from the entire image to allow for faster searching and to extract the unallocated sectors. I do both of these things. On a 64 Meg drive these are all fairly quick operations. On an image of a larger system these operations could take a considerable amount of time to complete.


Extract Strings of Entire Image

Extracting the ASCII and Unicode strings from a file system image will make keyword searching faster.

Generate MD5? ✓

ASCII: ✓ Unicode: ✓

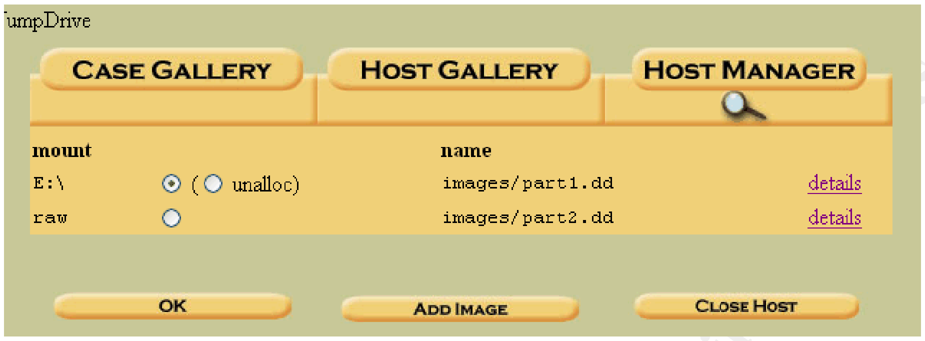
Extract Unallocated Sectors

Extracting the unallocated data in a file system image allows more focused keyword searches and data recovery.

(Note: This Does Not Include Slack Space)

Generate MD5? ✓

To begin examining the drive contents I select the E:\ drive and click Ok.



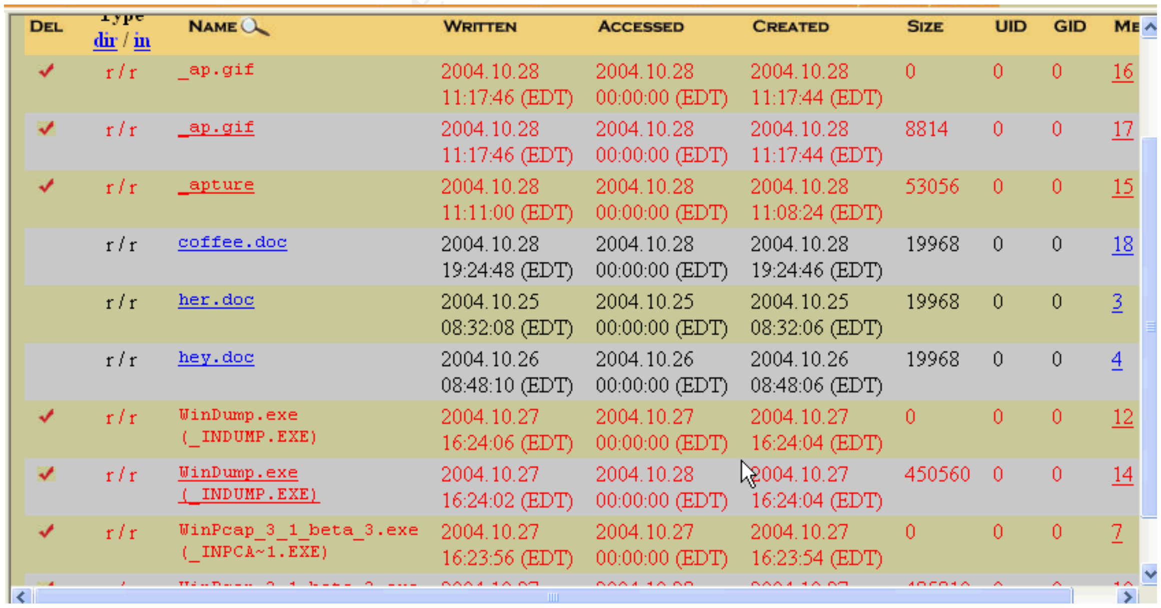
mount		name	
E:\	● (○ unalloc)	images/part1.dd	details
raw	○	images/part2.dd	details

From there I selected the Image Details option from the menu at the top of the interface. I looked over the information about the image and recorded it in my case notes. The Contents showed 3 files, which should correspond to the 3 word documents we saw on the timeline. The numbers below are the clusters of sectors that make up each file.

FAT CONTENTS (in sectors)

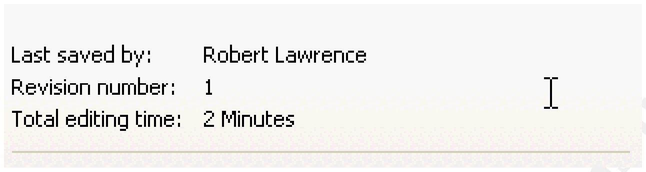
511-550 (40) -> EOF
551-590 (40) -> EOF
591-630 (40) -> EOF

I switch over to the File Analysis screen, in this view I can see not only the 3 document files but also the deleted files we saw in the timeline.



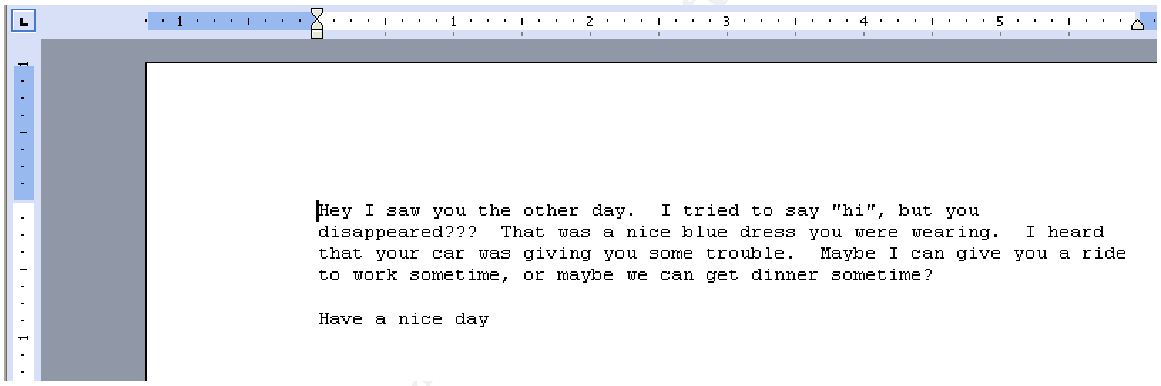
DEL	Type dir / in	NAME	WRITTEN	ACCESSED	CREATED	SIZE	UID	GID	ME
✓	r / r	_ap.gif	2004.10.28 11:17:46 (EDT)	2004.10.28 00:00:00 (EDT)	2004.10.28 11:17:44 (EDT)	0	0	0	16
✓	r / r	_ap.gif	2004.10.28 11:17:46 (EDT)	2004.10.28 00:00:00 (EDT)	2004.10.28 11:17:44 (EDT)	8814	0	0	17
✓	r / r	_apture	2004.10.28 11:11:00 (EDT)	2004.10.28 00:00:00 (EDT)	2004.10.28 11:08:24 (EDT)	53056	0	0	15
	r / r	coffee.doc	2004.10.28 19:24:48 (EDT)	2004.10.28 00:00:00 (EDT)	2004.10.28 19:24:46 (EDT)	19968	0	0	18
	r / r	her.doc	2004.10.25 08:32:08 (EDT)	2004.10.25 00:00:00 (EDT)	2004.10.25 08:32:06 (EDT)	19968	0	0	3
	r / r	hey.doc	2004.10.26 08:48:10 (EDT)	2004.10.26 00:00:00 (EDT)	2004.10.26 08:48:06 (EDT)	19968	0	0	4
✓	r / r	WinDump.exe (_INDUMP.EXE)	2004.10.27 16:24:06 (EDT)	2004.10.27 00:00:00 (EDT)	2004.10.27 16:24:04 (EDT)	0	0	0	12
✓	r / r	WinDump.exe (_INDUMP.EXE)	2004.10.27 16:24:02 (EDT)	2004.10.28 00:00:00 (EDT)	2004.10.27 16:24:04 (EDT)	450560	0	0	14
✓	r / r	WinPcap_3_1_beta_3.exe (_INPCA~1.EXE)	2004.10.27 16:23:56 (EDT)	2004.10.27 00:00:00 (EDT)	2004.10.27 16:23:54 (EDT)	0	0	0	7

I decide to begin by examining the 3 document files. I select each in turn starting with the oldest one named her.doc; I select the file name in the File Analysis screen and then examine both the ASCII Report and the ASCII Strings Report. I note that the name of the user who created the word document is recorded in the metadata of the file; Robert Lawrence's name appears there as the user who created and also last saved the file. Opening the exported file in Word will reveal this information in the properties of the document.



Last saved by: Robert Lawrence
Revision number: 1
Total editing time: 2 Minutes

I record the information contained in the file and make an export of the file using the Autopsy Export function. Once the file is saved locally I immediately make a MD5 hash for it. I copied the file to an XP system running inside a VMware session in order to open the file in Microsoft Word to obtain these screenshots.

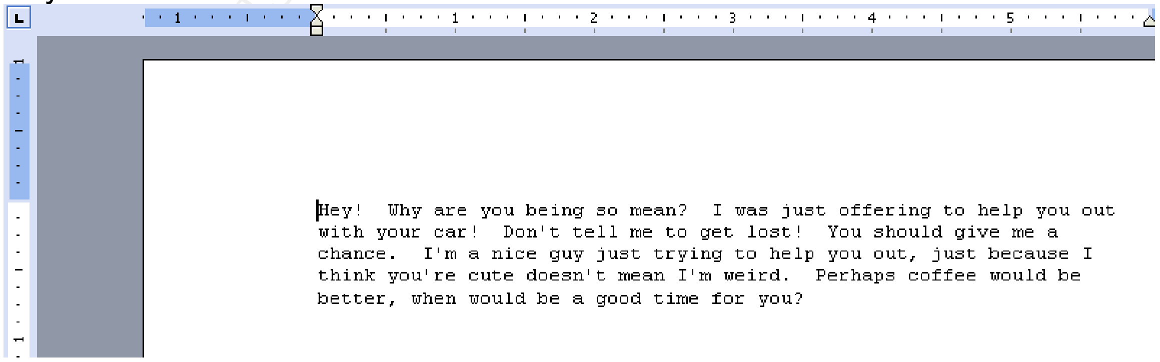


```
Hey I saw you the other day.  I tried to say "hi", but you
disappeared???  That was a nice blue dress you were wearing.  I heard
that your car was giving you some trouble.  Maybe I can give you a ride
to work sometime, or maybe we can get dinner sometime?

Have a nice day
```

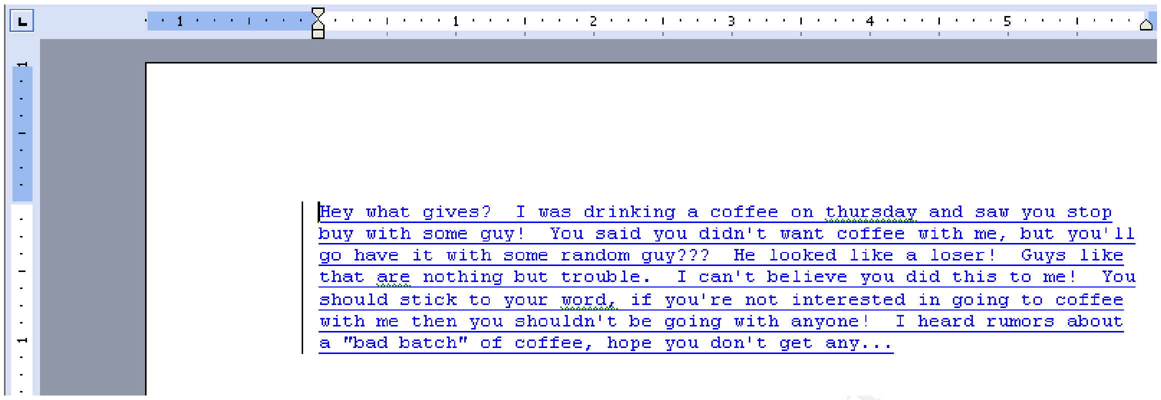
I examine the other 2 files using the same technique. I again note the information contained in the files and make export versions wit the associated MD5 hashes.

Hey.doc



```
Hey!  Why are you being so mean?  I was just offering to help you out
with your car!  Don't tell me to get lost!  You should give me a
chance.  I'm a nice guy just trying to help you out, just because I
think you're cute doesn't mean I'm weird.  Perhaps coffee would be
better, when would be a good time for you?
```

Coffee.doc



Hey what gives? I was drinking a coffee on thursday and saw you stop buy with some guy! You said you didn't want coffee with me, but you'll go have it with some random guy??? He looked like a loser! Guys like that are nothing but trouble. I can't believe you did this to me! You should stick to your word, if you're not interested in going to coffee with me then you shouldn't be going with anyone! I heard rumors about a "bad batch" of coffee, hope you don't get any...

In the contents of the documents I note a disturbing trend, however it only my role to establish evidence and report on it. I make notes to ensure that the HR department reviews the report for a decision on how to proceed.

I next turn my attention to the deleted files on the USB drive. These are files that have been unallocated at the Data Layer. As long as the files have not been overwritten by another file they can be recovered.

Again starting with the oldest first I turn my attention to WinPcap_3_1_beta_3.exe and WinDump.exe. I know from experience that these files make up the program and the drivers for a free windows version of the packet sniffing program Tcpdump. If I did not have experience with these files previously a search using google (http://www.google.com) would quickly identify them as well. I run a ASCII strings report for the file Windump.exe, it shows that the sectors are not overwritten so it should be recoverable. I export a copy of the file as "images-part1.dd-E...WinDump.exe.._INDUMP.EXE" and make an immediate MD5 checksum for it. I copy the file to a Windows XP system running in VMware on my Forensic Workstation and execute it with the –V flag. This causes the executable to return the version information:

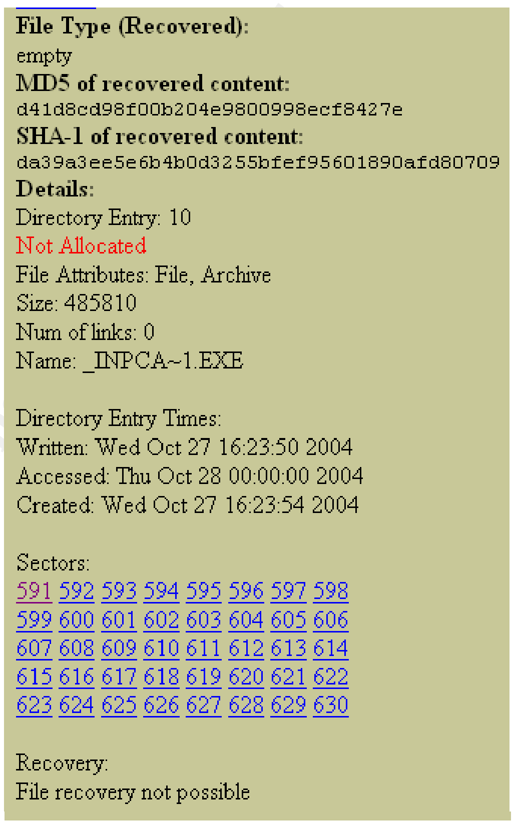
```
C:\>images-part1.dd-E...WinDump.exe.._INDUMP.EXE -V
images-part1.dd-E...WinDump.exe.._INDUMP.EXE version 3.8.3 beta,
based on tcpdum
p version 3.8.3
WinPcap version 3.1 beta4 (packet.dll version 3, 1, 0, 24), based on
libpcap ver
sion 0.8.3
Usage: images-part1.dd-E...WinDump.exe.._INDUMP.EXE [-
aAdDeflLnNOpqRStuUvxX] [ -
B size ] [-c count] [ -C file_size ]
                [ -E algo:secret ] [ -F file ] [ -i interface ] [ -r
file ]
                [ -s snaplen ] [ -T type ] [ -w file ] [ -y
datalinktype ]
                [ expression ]
```

As I suspected from the name, the file is Windump version 3.8.3 beta, the help information provides the version number. To further confirm this, I download Windump 3.8.3 beta from http://windump.polito.it and generate a MD5 hash for the newly downloaded file

```
deanf@forensicsworkstation $md5sum windump
79375b77975aa53a1b0507496107bff7  Windump.exe

deanf@forensicsworkstation $md5sum images-part1.dd-
E...WinDump.exe.._INDUMP.EXE
79375b77975aa53a1b0507496107bff7  images-part1.dd-
E...WinDump.exe.._INDUMP.EXE.
```

The hashes match so I can safely conclude that the deleted file recovered from the USB drive was in fact windump.exe. The file WinPcap_3_1_beta_3.exe is a bit of a problem however. It was not only deleted on 10/28 but part of it was later overwritten by the file coffee.doc. The WinPcap file was located at cluster 10, I went to the Metadata menu in Autopsy and selected the link for cluster 10. It shows me that the file should occupy the sectors from 591 to 630. Coffee.doc at cluster 18 occupies this same range of sectors, effectively overwriting this file. Note that the recovery information for cluster 10 in Autopsy also shows that recovery is not possible.



File Type (Recovered):
empty
MD5 of recovered content:
d41d8cd98f00b204e9800998ecf8427e
SHA-1 of recovered content:
da39a3ee5e6b4b0d3255bfef95601890afd80709
Details:
Directory Entry: 10
Not Allocated
File Attributes: File, Archive
Size: 485810
Num of links: 0
Name: _INPCA~1.EXE

Directory Entry Times:
Written: Wed Oct 27 16:23:50 2004
Accessed: Thu Oct 28 00:00:00 2004
Created: Wed Oct 27 16:23:54 2004

Sectors:
591 592 593 594 595 596 597 598
599 600 601 602 603 604 605 606
607 608 609 610 611 612 613 614
615 616 617 618 619 620 621 622
623 624 625 626 627 628 629 630

Recovery:
File recovery not possible

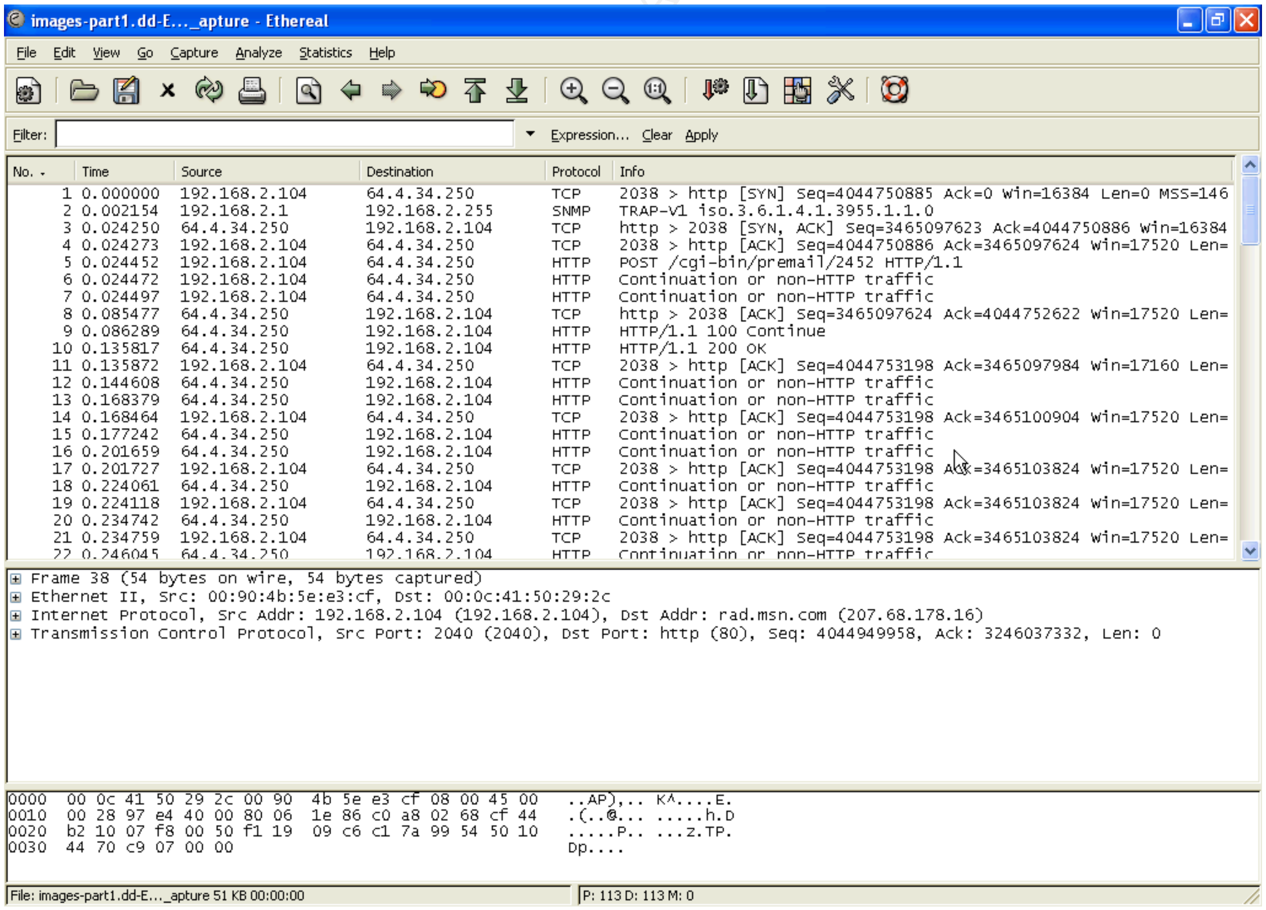
Because the Windump program cannot run without the WinPcap drivers it can be reasonably concluded that the deleted file was in fact the installer for the drivers but we will not be able to prove this from this drive image.

Next I turn my attention to the file “capture”, it was created on 10/28 at 11:08 and last written to at 11:11. An Autopsy Strings report indicates that this is a TCPdump Capture File, the likely output of the Windump program. I export this file as images-part1.dd-E..._apture and create an MD5 sum of it.

As a double-check once the file has been exported and the hash generated I execute the command file on it to confirm what the file contents are

```
[deanf@LinuxForensics CCTerminals]$ file images-part1.dd-E..._apture
images-part1.dd-E..._apture: tcpdump capture file (little-endian) -
version 2.4 (Ethernet, capture length 4096)
[deanf@LinuxForensics CCTerminals]$
```

Satisfied that the file is in fact a capture file, I open the file with the program Ethereal[9] which is a protocol analyzer. This file could be opened with a variety of tools such as tcpdump/windump or various types of protocol analyzers however Ethereal was chosen since it offers some nice decoding and search function as well as being a free tool that is frequently updated.


images-part1.dd-E..._apture - Ethereal
File Edit View Go Capture Analyze Statistics Help
Filter:
Expression... Clear Apply

```
No. -  Time      Source          Destination     Protocol Info
     1 0.000000  192.168.2.104   64.4.34.250     TCP      2038 > http [SYN] Seq=4044750885 Ack=0 Win=16384 Len=0 MSS=146
     2 0.002154  192.168.2.1     192.168.2.255   SNMP     TRAP-V1 iso.3.6.1.4.1.3955.1.1.0
     3 0.024250  64.4.34.250     192.168.2.104   TCP      http > 2038 [SYN, ACK] Seq=3465097623 Ack=4044750886 Win=16384
     4 0.024273  192.168.2.104   64.4.34.250     TCP      2038 > http [ACK] Seq=4044750886 Ack=3465097624 Win=17520 Len=
     5 0.024452  192.168.2.104   64.4.34.250     HTTP     POST /cgi-bin/premail/2452 HTTP/1.1
     6 0.024472  192.168.2.104   64.4.34.250     HTTP     Continuation or non-HTTP traffic
     7 0.024497  192.168.2.104   64.4.34.250     HTTP     Continuation or non-HTTP traffic
     8 0.085477  64.4.34.250     192.168.2.104   TCP      http > 2038 [ACK] Seq=3465097624 Ack=4044752622 Win=17520 Len=
     9 0.086289  64.4.34.250     192.168.2.104   HTTP     HTTP/1.1 100 Continue
    10 0.135817  64.4.34.250     192.168.2.104   HTTP     HTTP/1.1 200 OK
    11 0.135872  192.168.2.104   64.4.34.250     TCP      2038 > http [ACK] Seq=4044753198 Ack=3465097984 Win=17160 Len=
    12 0.144608  64.4.34.250     192.168.2.104   HTTP     Continuation or non-HTTP traffic
    13 0.168379  64.4.34.250     192.168.2.104   HTTP     Continuation or non-HTTP traffic
    14 0.168464  192.168.2.104   64.4.34.250     TCP      2038 > http [ACK] Seq=4044753198 Ack=3465100904 Win=17520 Len=
    15 0.177242  64.4.34.250     192.168.2.104   HTTP     Continuation or non-HTTP traffic
    16 0.201659  64.4.34.250     192.168.2.104   HTTP     Continuation or non-HTTP traffic
    17 0.201727  192.168.2.104   64.4.34.250     TCP      2038 > http [ACK] Seq=4044753198 Ack=3465103824 Win=17520 Len=
    18 0.224061  64.4.34.250     192.168.2.104   HTTP     Continuation or non-HTTP traffic
    19 0.224118  192.168.2.104   64.4.34.250     TCP      2038 > http [ACK] Seq=4044753198 Ack=3465103824 Win=17520 Len=
    20 0.234742  64.4.34.250     192.168.2.104   HTTP     Continuation or non-HTTP traffic
    21 0.234759  192.168.2.104   64.4.34.250     TCP      2038 > http [ACK] Seq=4044753198 Ack=3465103824 Win=17520 Len=
    22 0.246045  64.4.34.250     192.168.2.104   HTTP     Continuation or non-HTTP traffic
```

```
⊞ Frame 38 (54 bytes on wire, 54 bytes captured)
⊞ Ethernet II, Src: 00:90:4b:5e:e3:cf, Dst: 00:0c:41:50:29:2c
⊞ Internet Protocol, Src Addr: 192.168.2.104 (192.168.2.104), Dst Addr: rad.msn.com (207.68.178.16)
⊞ Transmission Control Protocol, Src Port: 2040 (2040), Dst Port: http (80), Seq: 4044949958, Ack: 3246037332, Len: 0
```

```
0000  00 0c 41 50 29 2c 00 90  4b 5e e3 cf 08 00 45 00   ..AP),.. K^....E.
0010  00 28 97 e4 40 00 80 06  1e 86 c0 a8 02 68 cf 44   .(..@... .....h.D
0020  b2 10 07 f8 00 50 f1 19  09 c6 c1 7a 99 54 50 10   .....P.. ...z.TP.
0030  44 70 c9 07 00 00                                  Dp....
```

File: images-part1.dd-E..._apture 51 KB 00:00:00 | P: 113 D: 113 M: 0

[9] http://www.ethereal.com/

The network traffic contained in the capture file is a HTTP session between the host at 192.168.2.104 and 64.4.34.250 (by12fd.bay12.hotmail.msn.com). Hotmail is a free e-mail account provider. Within the e-mail the user flowergirl96 is accepting an invitation to coffee at the coffee shop at the corner of Hollywood and McCadden from SamGuarillo@hotmail.com.

The following is an extraction of the e-mail conversation presumed to be between Miss Conaly and Mr. Guarillo. It was obtained by opening the capture file that was recovered from the USB drive in the program Ethereal. Ethereal is a program for collecting (commonly known as “Sniffing”) network traffic and analyzing the results. This is the content of the HTTP session between Miss Conlay’s browser and the e-mail server at Hotmail.

```
POST /cgi-bin/premail/2452 HTTP/1.1                    ← a post command sends data to the server over HTTP
Accept: image/gif, image/x-xbitmap, image/jpeg, image/pjpeg, application/vnd.ms-excel, application/vnd.ms-powerpoint,
application/msword, */*
Referer: http://by12fd.bay12.hotmail.msn.com/cgi-
bin/compose?&curmbox=F000000001&a=27d6f510deac1bac5415e72029263cd9
Accept-Language: en-us
Content-Type: application/x-www-form-urlencoded
Accept-Encoding: gzip, deflate
User-Agent: Mozilla/4.0 (compatible; MSIE 6.0; Windows NT 5.1; .NET CLR 1.1.4322)
Host: by12fd.bay12.hotmail.msn.com                              ← The machine the data is going
to
Content-Length: 576
Connection: Keep-Alive
Cache-Control: no-cache
Cookie: MC1=V=3&GUID=49A9B22A05294A1A81F11881BF3C264B; y=1;
MSPAuth=5Qr3f0LU3B54zQBmCG3iUtdaiAo608EFiBYmrtzv6oAL1cQ1ayApRce4N7XCEkk%2aa5e9H9cWS5x%21xBTivKy
%2aSEwg%24%24;
MSPProf=5e1XcTCShGOf1gQhcClTXJM67JMAbywIG67BmEwf%2aNbKWq2vOyMjJTO2P1%2aaU%2aviMTcr8nestOX6uJi
5QYv9nb%21V3ReGZPm3yhrewvAYzs3vjyK4rdsGyuC2UGGRIga01ksxgsOTye%2aN6x6RSiEoVSY1B7nwcTwqlcErZoYBZY
ceDYvmlHy2W1RBkki3tMoJtq2IN4ZFwblNM%24; PIM=1%2clang%2cEN%2ctabstyle%2c4%2c
cluster%2cby12fd%252ebay12%252ehotmail%252emsn%252ecom%2ctimestamp%2c1098692237%2csection%2cpersona
l%2csubsection%2cInvalidSubSection; mid=29ede1b79f320aa332327a4460; HMSatchmo=0; HMP1=1;
HMSC0899=224flowergirl96%40hotmail%2ecomrEM%2a5jEHcXVGV4%2aAWzQ6w%2a0KAj39KgAbJwM3dx89O12eFC
P8QpvDRxtOmG0LfDW%2azTT3QAp7%2aslY6H2QtQ5HQXNkLZglQmXIy9iEXRtDjJoz9OYjoxLF3Ma%2axDVQGszV4go%2a
u43pw8jYIglxM0UW%21z0ldqqhUN1TQ4ctSsc5TvwyIbDyDgcRpTSWI4a5eks5
ccQVXfG4uV1JekTVpqRyBUcsm9mPtf5j55s7ZhD82ttArNKHEJD92eufZJ8AVnTljxVkdfoHs%2aAyv%2a4HRUpaX5MT3Rkmxf
vaHdNIXwLGY3eGw2iYFxTBWHxOhAZMfocojMk6YQHaSLzEp4ueB3Cq0fUl29ndIe9jfW71zZRlTOxLaRk0LgudQuu%2aGGwy
JX%21WH%2aUfLO%2aeKlnyxDTIY35xVxy0LwJQ7wGI7fxd%2aTBu%2apX7tNZYmw6n4bzSUMtIXi6f
```

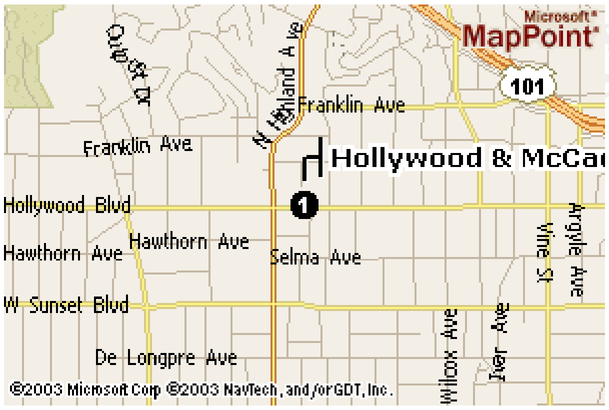
Above in red above you can see an E-mail address (flowergirl96@hotmail.com) that is unlikely to belong to Mr. Lawrence

```
curmbox=F000000001&HrsTest=&_HMaction=Send&FinalDest=&subaction=&plaintext=&login=flowergirl96&msg=&start=&
len=&attfile=&attlistfile=&eurl=&type=&src=&ref=&ru=&msghdrid=b16479b18beec291196189c78555223c_1098692452&RT
Ebgcolor=&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+sound
s+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
HTTP/1.1 100 Continue
```

In the block above we can see the user sending the e-mail (flowergirl96) who it is being sent to (SanGarillo@hotmail.com) and the text of the e-mail which includes the address and time that they will be meeting.

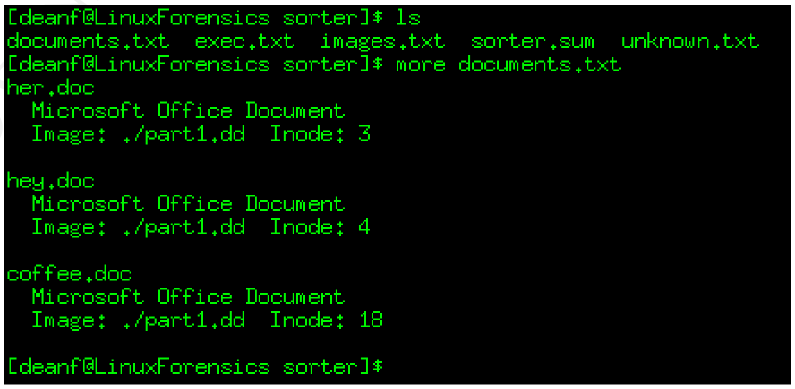
The information contained in this capture file would explain how Mr. Lawrence would have come to know Miss Conlay's personal e-mail address.

The final file to be examined is the deleted file _ap.gif. This file is exported from Autopsy and an MD5 sum created for it. It is opened in a browser and the contents are a map from Microsoft Map Point showing the corner of Hollywood and McCadden. This shows that Mr. Lawrence knew where Miss Conaly was to be at the appropriate time.



To ensure I have not missed possible evidence I also look at the small partition of unallocated space I carved out of the dd image of the drive. I use the ASCII reporting functionality to review the sectors in that partition. No ASCII text is found in that partition.

To double check my file recovery I run the tool sorter. Sorter is another component of the Sleuth Kit which automates and combines the operation of several other commands. What it does is execute the fls command which displays the file entries in a directory inode (or File Allocation Table). It then runs the icat command which displays the contents of the disk blocks allocated to the cluster. Finally the data from icat is passed to the command file which determines what the file type actually is, the data is then noted in the output or saved to file.



```
[deanf@LinuxForensics sorter]$ ls
documents.txt  exec.txt  images.txt  sorter.sum  unknown.txt
[deanf@LinuxForensics sorter]$ more documents.txt
her.doc
  Microsoft Office Document
  Image: ./part1.dd  Inode: 3

hey.doc
  Microsoft Office Document
  Image: ./part1.dd  Inode: 4

coffee.doc
  Microsoft Office Document
  Image: ./part1.dd  Inode: 18

[deanf@LinuxForensics sorter]$
```

The results are several output files, one for each file type identified. The contents of the output file documents.txt as shown above confirm that we have identified all the MS Word documents in the partition. Since Sorter uses the file command it can identify files that have had their file extension changed to attempt to obscure them as well.

Running sorter with the –s flag will cause it to save the files it identifies automatically in directories that are named for the type of file, such as documents, exec for executables, and images. I run sorter in this mode and compare MD5 hashes of the files recovered through Autopsy with the hashes of the files recovered with sorter. Since each set of hashes match I feel I have validated my earlier recovery efforts.

Finally I extract the slack space to ensure no additional evidence is hiding there. Slack space is allocated space that the file does not actually take up. A way to look at it is using the analogy of a videotape. If you have a 2 hour videotape and you record a 30 minute sitcom on it, the remaining 1.5 hours is "slack space". If that 2 hour videotape had previously been used to record a 1 hour program, and subsequently you recorded over it to capture your 30 minute sitcom, the remaining 30 minutes of the 1 hour program would still be viewable. This is what we are looking for when examining slack space, any data that remains in the allocated but unused space.

The tool dls from the SleuthKit can extract slack space. I execute it with the –s flag to do this and place the output in a file. The –s flag will tell dls to look in each file and print any data from the end of the file to the end of the cluster

```
deanf@LinuxForensics$ dls -s -f fat16 part1.dd >part1-slack.txt
```

I review the file with the Linux more command, the strings command and a hex editor but only binary data is revealed.

Image Details

File System Type: FAT
OEM Name: MSWIN4.1
Volume ID: 0x0
Volume Label (Boot Sector): NO NAME
Volume Label (Root Directory):
File System Type Label: FAT16

Sectors before file system: 32

File System Layout (in sectors)
Total Range: 0 - 121918
* Reserved: 0 - 0
** Boot Sector: 0
* FAT 0: 1 - 239
* FAT 1: 240 - 478
* Data Area: 479 - 121918
** Root Directory: 479 - 510
** Cluster Area: 511 - 121918

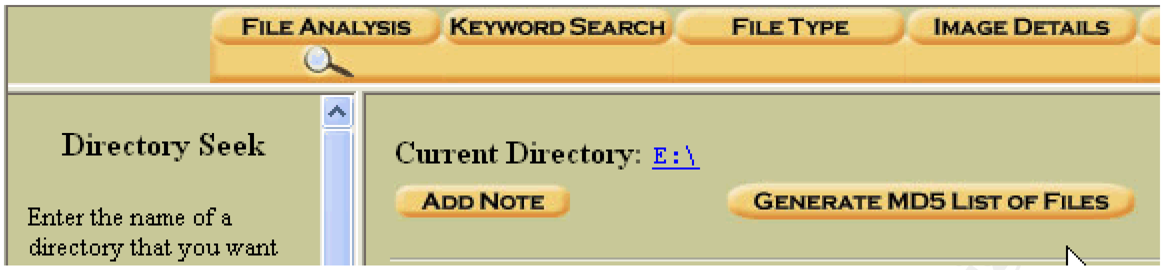
METADATA INFORMATION
Range: 2 - 1942530
Root Directory: 2

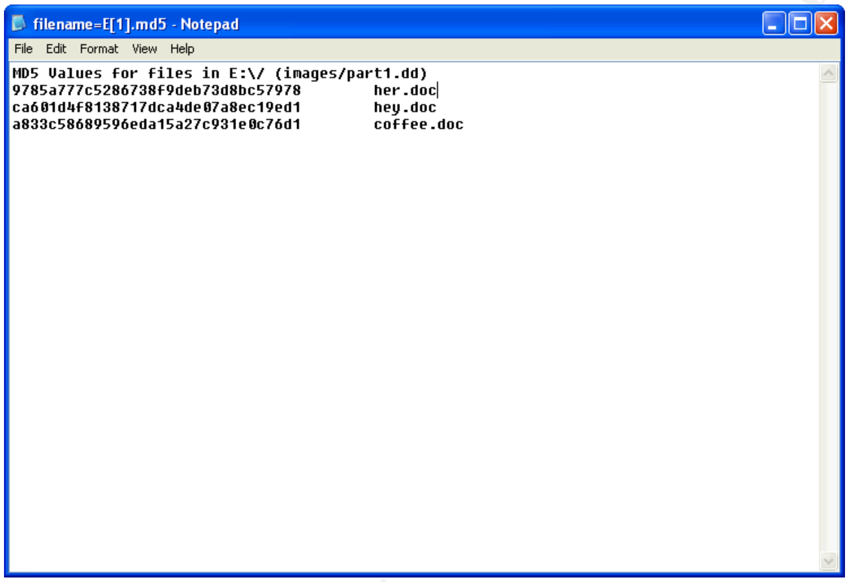
CONTENT INFORMATION
Sector Size: 512
Cluster Size: 1024
Total Cluster Range: 2 - 60705

The following are the files that are present on the USB drive. All files except the word documents have been deleted but recovered. One file WinPcap_3_1_beta_3.exe was deleted, but was then overwritten by the file coffee.doc. Because of this we have the inode data telling us it existed, but it cannot be recovered.

MD5 Hash	File Name	Size (in bytes)	Last Modified	Last Accessed	Last Changed
9785a777c5286738f9deb73d8bc57978	her.doc	19,968	2004.10.25 08:32:08 EDT	2004.10.25 00:00:00 EDT	2004.10.25 08:32:06 EDT
ca601d4f8138717dca4de07a8ec19ed1	hey.doc	19,968	2004.10.26 08:48:10 EDT	2004.10.26 00:00:00 EDT	2004.10.26 08:48:06 EDT
a833c58689596eda15a27c931e0c76d1	coffee.doc	19,968	2004.10.28 19.24:48 EDT	2004.10.28 00:00:00 EDT	2004.10.28 19:24:46 EDT
79375b77975aa53a1b0507496107bff7	Windump.exe	450,560	2004.10.27 16:24:02 EDT	2004.10.28 00:00:00 EDT	2004.10.27 16:24:04 EDT
9bc3923cf8e72fd405d7cea8c8781011	map.gif	8814	2004.10.28 11:17:46 EDT	2004.10.28 00:00:00 EDT	2004.10.28 11:17:44 EDT
2097b7b0a9fedb4238b67e976c4ae1cb	capture	53,056	2004.10.28 11:11:46 EDT	2004.10.28 00:00:00 EDT	2004.10.28 11:08:46 EDT

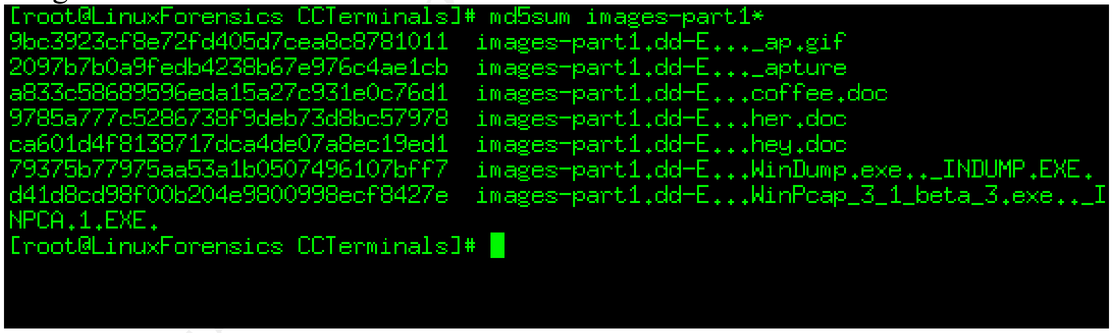
Hashes for the Word Documents can be obtained by using the 'Generate MD5 List of Files" option in Autopsy's File Analysis window.





```
MD5 Values for files in E:\/ (images/part1.dd)
9785a777c5286738f9deb73d8bc57978         her.doc
ca601d4f8138717dca4de07a8ec19ed1         hey.doc
a833c58689596eda15a27c931e0c76d1         coffee.doc
```

I compared this to the MD5 sums I obtained from the files I exported from the drive image. I used the md5sum tool to create these hashes.



```
[root@LinuxForensics CCTerminals]# md5sum images-part1*
9bc3923cf8e72fd405d7cea8c8781011  images-part1.dd-E..._ap.gif
2097b7b0a9fedb4238b67e976c4ae1cb  images-part1.dd-E..._apture
a833c58689596eda15a27c931e0c76d1  images-part1.dd-E...coffee.doc
9785a777c5286738f9deb73d8bc57978  images-part1.dd-E...her.doc
ca601d4f8138717dca4de07a8ec19ed1  images-part1.dd-E...hey.doc
79375b77975aa53a1b0507496107bff7  images-part1.dd-E...WinDump.exe.._INDUMP.EXE.
d41d8cd98f00b204e9800998ecf8427e  images-part1.dd-E...WinPcap_3_1_beta_3.exe.._I
NPCA.1.EXE.
[root@LinuxForensics CCTerminals]#
```

No user and group information exists on this drive as it is formatted with the FAT16 file system. The FAT format does not support file level access controls. If such information had been available we could have attempted to match the user and group information with the information from the system where the files where created.

Forensic Details

Two executables where recovered from the USB drive image.
The first program recovered from the USB drive was WinDump.exe, the other executable is the installer for the windows implementation of the libpcap library WinPcap_3_1_beta_3.exe. These drivers are needed to allow WinDump to operate. The combination of these two executables makes up the main tool used by Mr. Lawrence which is a Network traffic capture tool (packet sniffer).

Windump[10] is a real time packet capture tool that operates from the command prompt window on a Windows system. It is a version of the Unix/Linux based tcpdump[11] tool that was ported to run on Windows hosts. It is capable of recording all the network traffic in the form of packet headers and/or packet data that the system it is running on has access to. It takes packets from the network interface and writes them to the screen or to file. It is also capable of filtering only packets that match certain criteria such as protocol and source or destination address.

This tool is commonly used for network troubleshooting and network traffic analysis.

Program Identification

The file on the USB drive was a binary so I cannot compare uncompiled source to that obtained independently to verify it's identity, so on a WindowsXP system running under VMware I executed it with the –V flag. This causes the executable to return the version information:

```
C:\>images-part1.dd-E...WinDump.exe.._INDUMP.EXE -V
images-part1.dd-E...WinDump.exe.._INDUMP.EXE version 3.8.3 beta,
based on tcpdum
p version 3.8.3
WinPcap version 3.1 beta4 (packet.dll version 3, 1, 0, 24), based on
libpcap ver
sion 0.8.3
Usage: images-part1.dd-E...WinDump.exe.._INDUMP.EXE [-
aAdDeflLnNOpqRStuUvxX] [ -
```

[10] http://windump.polito.it/
[11] http://www.tcpdump.org/

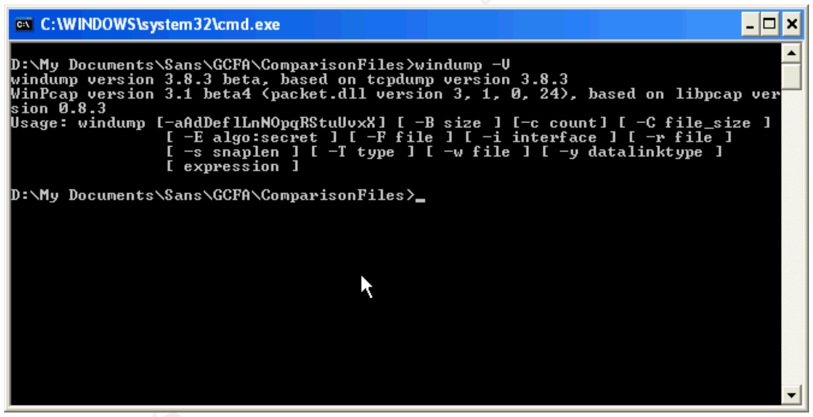
```
B size ] [-c count] [ -C file_size ]
                [ -E algo:secret ] [ -F file ] [ -i interface ] [ -r
file ]
                [ -s snaplen ] [ -T type ] [ -w file ] [ -y
datalinktype ]
                [ expression ]
```

As I suspected from the name, the file is Windump, the help information shows that the version is 3.8.3 beta. To further confirm this, I download Windump 3.8.3 beta from http://windump.polito.it and generate a MD5 hash for the newly downloaded file

```
79375b77975aa53a1b0507496107bff7  Windump.exe

79375b77975aa53a1b0507496107bff7  images-part1.dd-
E...WinDump.exe.._INDUMP.EXE.
```

The hashes match so I can safely conclude that the deleted file recovered from the USB drive was in fact windump.exe. I also ran the comparison file and recorder the output for comparison the output from part1.dd-E...WinDump.exe.._INDUMP.EXE



```
D:\My Documents\Sans\GCFA\ComparisonFiles>windump -V
windump version 3.8.3 beta, based on tcpdump version 3.8.3
WinPcap version 3.1 beta4 (packet.dll version 3, 1, 0, 24), based on libpcap ver
sion 0.8.3
Usage: windump [-aAdDeflLnNOpqRStuUvxX] [ -B size ] [-c count] [ -C file_size ]
                [ -E algo:secret ] [ -F file ] [ -i interface ] [ -r file ]
                [ -s snaplen ] [ -T type ] [ -w file ] [ -y datalinktype ]
                [ expression ]

D:\My Documents\Sans\GCFA\ComparisonFiles>_
```

As final proof I run strings on both the recovered file and the downloaded comparison file and pipe the results to a text file. I then use a windows based file comparison tool called CSDiff[12] to identify any differences in the output. The tool identifies no differences in the strings output.

Legal Implications

There are many items in this investigation that should be reviewed for possible further action. The recovered file "Capture" is a recording of electronic communications between Leila Conaly and a Mr. Sam Guarillo. This communication is in the form of a captured e-mail message. Capturing electronic communications is a violation of one of two federal laws, The Electronic Communications Privacy Act, 18 U.S.C. §§ 2701-12 covers access to

[12] http://www.componentsoftware.com/products/csdiff/download.htm

stored information like E-mail and Voice Mail or the Wiretap statute, 18 U.S.C §§ 2510-22 which is also referred to as Title III of the Omnibus Crime Control and Safe Streets Act of 1968 which covers the real time collection of wire and electronic communications content. In this discussion we will henceforth refer to this as Title III. This specific instance seems to be a violation of Title III as the communication (the e-mail message) was intercepted in transit (in real time) instead of when stored which would have been a violation of the ECPA. Further investigation would be required to determine if consent of either party in the e-mail communication was given to monitor the communication. There are several exceptions that could allow such monitoring to take place, but due to the nature of his job (sales) Mr. Lawrence does not qualify for most of them with the possible exception of having the consent of one or more parties to the communication. The location of the incident (state) would be a know factor in a real investigation, for the purpose of this practical the location was not provided as part of the background. Based on the recovered evidence file Map.gif I did some location research, it appears that the location mentioned in Miss Conlay's e-mail and the scene of one of her encounters with Mr. Lawrence is located in California. This is based off of the streets depicted in that map file. If the incident did take place in California, consent to monitoring must be provided by ALL parties to the communication[13]. Since Miss Conlay did not consent to monitoring this exception is void as well.

Punishments for violations of Title III can include a jail term of up to 5 years and fines for criminal cases[14] or fines computed according to various formulas for civil cases[15].

It is also possible that the Federal Computer Fraud & Abuse act (18 U.S.C §1030) could be involved since the company where the incident took place is involved in financial transactions. This will require interpretation by legal council since the actions where not directed against the institution itself and the test for damages does not seem to be met. In order to establish a violation of §1030 we would need to be able to show damage in excess of $5,000 in 1 year, impairment of medical records, Physical Injury to a person, a threat to public health and safety, or damage affecting a governmental system (that fit within certain categories).

Another set of laws that could apply are any local Cyber stalking laws. The location of the incident (state) would be a know factor in a real investigation, for the purpose of this practical the location was not provided as part of the background. Based on the recovered evidence file Map.gif I did some location research, it appears that the location mentioned in Miss Conlay's e-mail and the scene of one of her encounters with Mr. Lawrence is located in California. This is based off of the streets depicted in that map file. If the incidents from this

[13] http://www.ncsl.org/programs/lis/CIP/surveillance.htm
[14] http://assembler.law.cornell.edu/uscode/html/uscode18/usc_sec_18_00002511----000-.html
[15] http://assembler.law.cornell.edu/uscode/html/uscode18/usc_sec_18_00002520----000-.html

case did take place in California then it is possible that California Penal Code § 646.9 – Stalking could also be in play. To establish this there is an 8 part test that the case must pass.

Recommendations

Based upon the results of this investigation, I recommend the evidence be reviewed by Corporate HR for immediate action. Referral to local law enforcement for any action is a decision that should be made by the HR department, possible in conjunction with corporate legal based on the strength of the evidence from the forensic examination. This recommendation is out of concern for Ms Conlay's well being, due to the escalating tone of the recovered documents.

During the investigation evidence was uncovered relating to the violation of both internal company policy as well as Title III (the federal wiretap statute 18 U.S.C §§ 2510-22). Evidence shows the use of an unauthorized network sniffer being used to capture e-mail communications. If law enforcement is not involved then Mr. Lawrence's actions should be referred to HR for appropriate internal disciplinary process for violation of company policy.

Evidence from this investigation support the claims made by Ms. Conlay.

Additional Information

Some resources I used in the preparation of this document where:

Mandia Kevin, Chris Prosise, and Matt Pepe. Incident Response and Computer Forensics, Second Edition. Emeryville: McGraw-Hill/Osborne, 2003
This book offered some good sections on Evidence Handling that where useful in helping shape my discussion of procedures.

"Computers are like Filing Cabinets…"Using Analogy to Explain Computer Forensics The National District Attorneys Association/American Prosecutors Research Institute website. 2002 http://www.ndaa-apri.org/publications/newsletters/update_volume_15_number_9_2002.html
This site contained some useful information on how to use analogy in explaining forensic concepts.

SANS Institute, Track 8 – System Forensics, Investigation & Response.

Volumes 8.1 to 8.5, Sans Press, October 2004
The Sans courseware was an invaluable reference during the preparation of this paper, it has descriptions of the use of all the tools described here as well as descriptions of the relevant legal issues.

References

Carvey, Harlan. Windows Forensics and Incident Response. Boston: Addison-Wesley, 2004

Shinder, Debra Littlejohn. Scene of the Cybercrime: Computer Forensics Handbook. Rockland: Syngress, 2002

Mandia Kevin, Chris Prosise, and Matt Pepe. Incident Response and Computer Forensics, Second Edition. Emeryville: McGraw-Hill/Osborne, 2003

United States. Department of Justice. Searching and Seizing Computers and Obtaining Electronic Evidence in Criminal Investigations. Washington: GPO 2002

"Guide for Forensic Examinations". The International Association of Computer Investigative Specialists (IACIS) Website. 1 February 2005 http://www.cops.org/html/forensicprocedures.htm

"Computers are like Filing Cabinets…"Using Analogy to Explain Computer Forensics The National District Attorneys Association/American Prosecutors Research Institute website. 2002 http://www.ndaa-apri.org/publications/newsletters/update_volume_15_number_9_2002.html

Mattsson, Johny. Web Page, 2005 http://www.earthmagic.org/?software

"Federal Rules of Evidence" Cornell University Law School, Legal Information Institute website. 2004 http://www.law.cornell.edu/rules/fre/rules.htm

Appendix A - Timeline

Date/Time	File Size	MAC	Permissions	User	Group	MetaData Address	File Name or ils dead name
Mon Oct 25 2004 00:00:00	19968	.a.	-/-rwxrwxrwx	0	0	3	e:/her.doc
Mon Oct 25 2004 08:32:06	19968	..c	-/-rwxrwxrwx	0	0	3	e:/her.doc

```
Mon Oct 25 2004     19968   m..  -/-rwxrwxrwx  0  0   3  e:/her.doc
08:32:08
Tue Oct 26 2004     19968   .a.  -/-rwxrwxrwx  0  0   4  e:/hey.doc
00:00:00
Tue Oct 26 2004     19968   ..c  -/-rwxrwxrwx  0  0   4  e:/hey.doc
08:48:06
Tue Oct 26 2004     19968   m..  -/-rwxrwxrwx  0  0   4  e:/hey.doc
08:48:10
Wed Oct 27 2004    485810   .a.  -/-rwxrwxrwx  0  0   7  e:/WinPcap_3_1_beta_3.ex
00:00:00                                                 e (_INPCA~1.EXE) (deleted)
                   450560   .a.  -/-rwxrwxrwx  0  0  12  e:/WinDump.exe
                                                         (_INDUMP.EXE) (deleted)
                        0   .a.  rwxrwxrwx     0  0   7  <part1.dd-_INPCA~1.EXE-
                                                         dead-7>
                        0   .a.  rwxrwxrwx     0  0  12  <part1.dd-_INDUMP.EXE-
                                                         dead-12>
Wed Oct 27 2004    485810   m..  -/-rwxrwxrwx  0  0  10  e:/WinPcap_3_1_beta_3.ex
16:23:50                                                 e (_INPCA~1.EXE) (deleted)
                   485810   m..  rwxrwxrwx     0  0  10  <part1.dd-_INPCA~1.EXE-
                                                         dead-10>
Wed Oct 27 2004         0   ..c  rwxrwxrwx     0  0   7  <part1.dd-_INPCA~1.EXE-
16:23:54                                                 dead-7>
                   485810   ..c  -/-rwxrwxrwx  0  0  10  e:/WinPcap_3_1_beta_3.ex
                                                         e (_INPCA~1.EXE) (deleted)
                   485810   ..c  rwxrwxrwx     0  0  10  <part1.dd-_INPCA~1.EXE-
                                                         dead-10>
                   485810   ..c  -/-rwxrwxrwx  0  0   7  e:/WinPcap_3_1_beta_3.ex
                                                         e (_INPCA~1.EXE) (deleted)
Wed Oct 27 2004         0   m..  rwxrwxrwx     0  0   7  <part1.dd-_INPCA~1.EXE-
16:23:56                                                 dead-7>
                   485810   m..  -/-rwxrwxrwx  0  0   7  e:/WinPcap_3_1_beta_3.ex
                                                         e (_INPCA~1.EXE) (deleted)
Wed Oct 27 2004    450560   m..  -/-rwxrwxrwx  0  0  14  e:/WinDump.exe
16:24:02                                                 (_INDUMP.EXE) (deleted)
                   450560   m..  rwxrwxrwx     0  0  14  <part1.dd-_INDUMP.EXE-
                                                         dead-14>
Wed Oct 27 2004    450560   ..c  rwxrwxrwx     0  0  14  <part1.dd-_INDUMP.EXE-
16:24:04                                                 dead-14>
                        0   ..c  rwxrwxrwx     0  0  12  <part1.dd-_INDUMP.EXE-
                                                         dead-12>
                   450560   ..c  -/-rwxrwxrwx  0  0  12  e:/WinDump.exe
                                                         (_INDUMP.EXE) (deleted)
                   450560   ..c  -/-rwxrwxrwx  0  0  14  e:/WinDump.exe
                                                         (_INDUMP.EXE) (deleted)
Wed Oct 27 2004    450560   m..  -/-rwxrwxrwx  0  0  12  e:/WinDump.exe
16:24:06                                                 (_INDUMP.EXE) (deleted)
                        0   m..  rwxrwxrwx     0  0  12  <part1.dd-_INDUMP.EXE-
                                                         dead-12>
Thu Oct 28 2004     53056   .a.  -/-rwxrwxrwx  0  0  15  e:/_apture (deleted)
00:00:00
```

```
                      485810  .a.  -/-rwxrwxrwx  0  0  10  e:/WinPcap_3_1_beta_3.ex
                                                            e (_INPCA~1.EXE) (deleted)
                       19968  .a.  -/-rwxrwxrwx  0  0  18  e:/coffee.doc
                      450560  .a.  -/-rwxrwxrwx  0  0  14  e:/WinDump.exe
                                                            (_INDUMP.EXE) (deleted)
                      450560  .a.  rwxrwxrwx     0  0  14  <part1.dd-_INDUMP.EXE-
                                                            dead-14>
                           0  .a.  rwxrwxrwx     0  0  16  <part1.dd-_ap.gif-dead-16>
                      485810  .a.  rwxrwxrwx     0  0  10  <part1.dd-_INPCA~1.EXE-
                                                            dead-10>
                        8814  .a.  rwxrwxrwx     0  0  17  <part1.dd-_ap.gif-dead-17>
                        8814  .a.  -/-rwxrwxrwx  0  0  17  e:/_ap.gif (deleted)
                        8814  .a.  -/-rwxrwxrwx  0  0  16  e:/_ap.gif (deleted)
                       53056  .a.  rwxrwxrwx     0  0  15  <part1.dd-_apture-dead-15>
Thu Oct 28 2004        53056  ..c  rwxrwxrwx     0  0  15  <part1.dd-_apture-dead-15>
11:08:24
                       53056  ..c  -/-rwxrwxrwx  0  0  15  e:/_apture (deleted)
Thu Oct 28 2004        53056  m..  rwxrwxrwx     0  0  15  <part1.dd-_apture-dead-15>
11:11:00
                       53056  m..  -/-rwxrwxrwx  0  0  15  e:/_apture (deleted)
Thu Oct 28 2004         8814  ..c  rwxrwxrwx     0  0  17  <part1.dd-_ap.gif-dead-17>
11:17:44
                           0  ..c  rwxrwxrwx     0  0  16  <part1.dd-_ap.gif-dead-16>
                        8814  ..c  -/-rwxrwxrwx  0  0  16  e:/_ap.gif (deleted)
                        8814  ..c  -/-rwxrwxrwx  0  0  17  e:/_ap.gif (deleted)
Thu Oct 28 2004         8814  m..  -/-rwxrwxrwx  0  0  16  e:/_ap.gif (deleted)
11:17:46
                        8814  m..  -/-rwxrwxrwx  0  0  17  e:/_ap.gif (deleted)
                           0  m..  rwxrwxrwx     0  0  16  <part1.dd-_ap.gif-dead-16>
                        8814  m..  rwxrwxrwx     0  0  17  <part1.dd-_ap.gif-dead-17>
Thu Oct 28 2004        19968  ..c  -/-rwxrwxrwx  0  0  18  e:/coffee.doc
19:24:46
Thu Oct 28 2004        19968  m..  -/-rwxrwxrwx  0  0  18  e:/coffee.doc
19:24:48
```