



2007

## Identifying Non-Volatile Data Storage Areas: Unique Notebook Identification Information as Digital Evidence


Nikica Budimir

*Centre of Excellence in Defence Industry Systems Capability, University of South Australia*

Jill Slay

*Centre of Excellence in Defence Industry Systems Capability*

Follow this and additional works at: <https://commons.erau.edu/jdfsl>

 Part of the [Computer Engineering Commons](#), [Computer Law Commons](#), [Electrical and Computer Engineering Commons](#), [Forensic Science and Technology Commons](#), and the [Information Security Commons](#)

### Recommended Citation

Budimir, Nikica and Slay, Jill (2007) "Identifying Non-Volatile Data Storage Areas: Unique Notebook Identification Information as Digital Evidence," *Journal of Digital Forensics, Security and Law*. Vol. 2 : No. 1 , Article 4.

DOI: <https://doi.org/10.15394/jdfsl.2007.1018>

Available at: <https://commons.erau.edu/jdfsl/vol2/iss1/4>

This Article is brought to you for free and open access by the Journals at Scholarly Commons. It has been accepted for inclusion in Journal of Digital Forensics, Security and Law by an authorized administrator of Scholarly Commons. For more information, please contact [commons@erau.edu](mailto:commons@erau.edu).



# **Identifying Non-Volatile Data Storage Areas: Unique Notebook Identification Information as Digital Evidence**

**Nikica Budimir**

Centre of Excellence in Defence Industry Systems Capability  
University of South Australia,  
Mawson Lakes, South Australia  
Australia  
budny001@students.unisa.edu.au

**Jill Slay**

Centre of Excellence in Defence Industry Systems Capability  
Mawson Lakes, South Australia  
Australia  
MAWSON LAKES, SA Australia  
Jill.slay@unisa.edu.au

## **ABSTRACT**

The research reported in this paper introduces new techniques to aid in the identification of recovered notebook computers so they may be returned to the rightful owner. We identify non-volatile data storage areas as a means of facilitating the safe storing of computer identification information. A forensic proof of concept tool has been designed to test the feasibility of several storage locations identified within this work to hold the data needed to uniquely identify a computer. The tool was used to perform the creation and extraction of created information in order to allow the analysis of the non-volatile storage locations as valid storage areas capable of holding and preserving the data created within them. While the format of the information used to identify the machine itself is important, this research only discusses the insertion, storage and ability to retain such information.

**Keywords:** unique identification; non-volatile storage areas; BIOS; MBR; notebook identification; digital evidence; forensic computing.

## **1. INTRODUCTION**

Notebook theft has become a major international issue. Once a notebook computer is stolen, the affected user often does not know how to identify his/her notebook computer. They are usually only able to describe the colour and markings on the notebook computer for identification purposes. As the number of notebook computers sold increases, so does the number of stolen

notebook computers, and as the number of identical notebook computer models multiplies, the recovery process is made more complex. Governing bodies, such as the police, lack the standards and tools to be able to uniquely identify a notebook computer, leaving little chance of establishing a link between a recovered notebook computer and its owner. Commercial tools, available for notebook computer protection, include prevention from unauthorised access or data protection through encryption; these however do not address the issue of uniquely identifying a notebook computer in case of theft.

There is little ability to rely on physical serial numbers as a means of identification since, anecdotally, the first task a criminal undertakes is that of removing and swapping hard disk and CD/ DVD drives and replacing with alternative stolen parts; hardware components are exchanged to remove standard means of identification, making the identification of the notebook computer difficult. Stolen notebook computers are often reformatted and this reduces the likelihood of identifying them. Due to this problem, it has become necessary to identify locations, and means, by which a notebook computer can be uniquely identified to increase the rate of linking of a notebook computer to its owner.

The identification of recovered stolen notebook computers is hindered by the lack of permanent identifiable markers. Being able to provide a location for a permanent marker would increase the rate of identification of notebook computers to their original owners. By marking the locations at which the markers would be stored, in a forensically sound manner, law enforcement would also be able to use the evidence in the court of law to not only apprehend and convict but possibly also discourage future thefts of notebook computers.

This research focuses on the identification of a notebook computer that was previously stolen and later recovered by law enforcement agencies. The problem faced by law enforcement is two fold. Firstly the investigators looking to identify these machines are often not computer experts and, secondly, the number of notebook computers that are recovered without distinguishing features is increasing, and looking through each machine trying to find information about the actual owner of the machine is a tedious and time consuming task. In order to facilitate a more streamlined, forensically sound, more user-friendly and faster approach to matching a notebook computer with its owner, a proof-of-concept tool needed to be designed. This tool would be able to uniquely identify a notebook computer and store this unique information somewhere on the notebook computer. The location of this information would need to remain both secure and unchanged even after hardware removal or formatting of the hard disks. We make the assumption that that the primary disk drive is formatted using NTFS.

## **2. FORENSIC COMPUTING RESEARCH AND RECOVERY OF MOBILE DEVICES**

There are several expanding areas of research in forensic computing. The term mobile no longer applies to the telephone alone; new technologies have emerged on the market, including mobile phones, PDAs and notebook computers. Armstrong et al (2004) report on the CSI/FBI survey conducted in 2003, and compares the results with Australian statistics, which showing the serious problem of notebook computer recovery not only in America but world wide. In the state of NSW, Australia, 10,000 notebook computers were stolen, and only 336 were recovered, making computer theft a serious problem (Armstrong, Wynne & O'Shea 2004). Looking more closely at the figures in the CSI/FBI survey it shows that around a third of interviewed participants have experienced notebook computer theft with financial losses of around \$11 million (DeMaria 2002). Armstrong et al (2004) stress the lack of security implementations on such machines and the need for tools to aid in recovery of these machines (Armstrong, Wynne & O'Shea 2004). In order to increase the percentage of notebook computer recovery, a new system needs to be devised. Recovery can be understood as the confiscation of a machine from the suspected thief and the returning of the machine to its owner. This research focuses on the later interpretation.

To effectively recover a notebook computer there needs to be a means of identifying a notebook computer. This research looks to place an identifier somewhere on the machine itself. Though it is not part of this research, it is important to consider what type of information needs to be stored in the identifier. There have been several concepts proposed on what should be placed in an identifier that would potentially identify one notebook computer over another. Before looking at what it needs to contain it is important to state that the nonvolatile memories are limited in unallocated space. Bursky (2003) suggests that improvements are "on the way" with Hitachi demonstrating a working 1GB flash memory chip solving the restriction on available space (Bursky 2003). As this technology is unlikely to be deployed and mass distributed in general purpose machine in the near future, the research will regard the limitations of available space as a valid restriction. Examining the literature different authors suggest diverse information for the identifier, but one paper of interest suggests 15 different key identifiers that could be collected in order to sufficiently identify the notebook computer (Slay et al. 2004). Information such as serial numbers, model numbers, type of notebook computer, processor speed, types of hardware contained on the notebook computer, the date purchased, the time zones and so forth are all mentioned as possible identifiers. This information in general is good enough to identify a machine; a problem with this type of identifier is that it would be hard if not impossible to collect all of them with software alone. The reason for this is that if the tools should remain simple to use without user interaction the serial

numbers that are not accessible with software would not be retrievable. Should the choice be made to allow user interaction with the tool, then information such as the users name and contact details could be also collected as suggested in the paper by (Slay et al. 2004). Such information needs to be stored on the machine in a manner that would allow the information to remain on the notebook computer even after reformats and removal of peripherals.

### **3. REVIEW OF EXISTING TOOLS AND APPLICATIONS**

There have been several attempts at addressing the issues of notebook computer security. Each of these applications addresses one or two parts of the problem. Firstly some of the applications look to address data theft, as often trade and government secrets could be stored on these machines. The tools described below all address data security through different means of either encrypting or installing kill switches that just wipes all the data on the machine.

#### **3.1 Beachhead Solutions – Mobile Data Vulnerability (Beachhead 2006) and PointSec – PointSec Laptop(PointSec 2006)**

This application is a software based implementation installed on the target notebook computer. It employs a multilayered approach to data security by applying both encryption and user's behaviour to detect possible unauthorised access to the data. This as a result would initiate a wiping action that can either delete some sections of the hard disk or the complete hard disk. This can be done both with the machine online or offline. This tool is only of use if the person is not concerned with the cost involved when losing a notebook computer but rather with what data is stored on it. It fails to address the identification process all together.

It is important for corporations to keep track of these machines to reduce the cost of data being compromised. The tools listed below are all designed to track the machines through the use of dedicated server that ping the machine to discover its whereabouts.

#### **3.2 Absolute Software – AbsoluteTrack (AbsoluteSoftware 2006a)**

This tool again requires a dedicated server to track the notebook computer. This application however uses the BIOS as its repository for the application that reports to the server. While this implementation is more robust and the application is likely to remain on the machine it still does not address the need to connect to the internet. If the machine never connects to the internet it is of little use as a tracking application.

#### **3.3 Absolute Software – Computrace LoJack for Laptops (AbsoluteSoftware 2006b)**

This application is installed on a notebook computer and constantly pings a

server to verify its locations. If the notebook computer is stolen and reconnected to the internet the server already knows that it needs to start tracking of the notebook computer. A secret email is sent to the server containing information such as the IP address, the ISP through which it is connected and potentially other information. One interesting feature is that the software is stored in the HPA/DCO which would allow it to remain on the hard disk even in the event that the target notebook computer is reformatted.

While these tools circumvent the identification process all together by tracking the whereabouts of the machine itself and notifying the law enforcement where it is at any given time to be able to recover the machine, they how ever do not address the issue of a person removing the software of the machine or never connecting to the internet to allow the machine to report to the tracking server. Should the machine then get recovered by the law enforcement, no identifiable information would be found on the

#### **4. THIS RESEARCH**

After a review of needs and existing tools, research questions answered here are thus:

##### **What types of non-volatile storage areas exist within a notebook computer running a Windows XP Operating System?**

This includes examining traditional non-volatile memories such as the Basic Input Output System (BIOS), Master Boot Record (MBR) and hidden partitions, but the research will look further to identify locations such as the advanced configuration and power interface (ACPI) and other potential locations.

##### **Is it possible to write to these storage areas?**

Once identified, we attempt to write to these storage areas, either through the utilisation of third party tools, or if not available, with custom designed tools and processes.

##### **Which storage location can hold enough data to uniquely identify a notebook computer?**

Once the first two questions were answered, enough information was provided to successfully answer the third question. Three key criteria were addressed in order for the storage to be considered, these criteria have been outlined below.

- Storage has enough unallocated space
- Storage can be written to
- Storage maintains persistent state

The criteria addressed the need for the storage capacity, the ability of the storage to hold enough data to successfully identify a notebook computer and persistence; to remain after power downs, reformat and hardware exchanges.

Only when all three criteria were satisfied, have the non-volatile storage locations been considered suitable for the storing of notebook computer identification information.

## 5. STORAGE LOCATIONS

There are several areas in a notebook computer that might be used a storage areas that are potentially non-volatile in nature and capable of holding information as suggested by Slay et al. (2004).

### 5.1 Master Boot Record (MBR)

The MBR is the data structure created on a hard disk when the first partition is created. It is located at cylinder 0 head 0 sector 1 and contains the boot code to run the partition (NTFS.COM 2006). The location of the MBR as the information above suggests is located at the start of the hard disk. The first 63 sectors of any hard disk are reserved for the MBR code. A snap shot of the complete MBR is shown in Figure 1, which shows the address range of the MBR code.

It is clearly visible that the boot code only takes up the first sector of the MBR. This leaves 62 remaining sectors that are reserved on the hard disk but remain unused (ranish.com 1998). FDISK is a tool used to create and format the MBR. FDISK is a MS-DOS command that was developed by Microsoft to allow the forming of hard disks (Microsoft 2005). After examining the tool is was evident that it did not modify the MBR but rather remove and insert a fresh copy. The reading and writing feature however is a valuable function that needs to be examined during the experimentation stage.

```
Entire MBR record in hex and ASCII
OFFSET 0 1 2 3 4 5 6 7 8 9 A B C D E F *0123456789ABCDEF
000000 fa33c08e d0bc007c 8bf45007 501ffbfc *.3.....|..P.P...
000010 bf0006b9 0001f2a5 ea1d0600 00bebe07 *.....
000020 b304803c 80740e80 3c00751c 83c610fe *.....t....u....
000030 cb75efcd 188b148b 4c028bee 83c610fe *.u.....L.....
000040 cb741a80 3c0074f4 be8b06ac 3c00740b *.t....t.....t.
000050 56bb0700 b40ecd10 5eebf0eb febf0500 *V.....^.....
000060 bb007cb8 010257cd 135f730c 33c0cd13 *..|...W...s.3...
000070 4f75edbe a306ebd3 bec206bf fe7d813d *Ou.....}.=
000080 55aa75c7 8bf5ea00 7c000049 6e76616c *U.u....|..Inval
000090 69642070 61727469 74696f6e 20746162 *id partition tab
0000a0 6c650045 72726f72 206c6f61 64696e67 *le.Error loading
0000b0 206f7065 72617469 6e672073 79737465 * operating syste
0000c0 6d004d69 7373696e 67206f70 65726174 *m.Missing operat
0000d0 696e6720 73797374 656d0000 00000000 *ing system.....
0000e0 00000000 00000000 00000000 00000000 *.....
0000f0 TO 0001af SAME AS ABOVE
0001b0 00000000 00000000 00000000 00008001 *.....
0001c0 0100060d fef83e00 00000678 0d000000 *.....x....
0001d0 00000000 00000000 00000000 00000000 *.....
0001e0 00000000 00000000 00000000 00000000 *.....
0001f0 00000000 00000000 00000000 000055aa *.....U.
```

Figure 1: Snapshot of MBR boot code/first sector

## **5.2 Basic Input Output System (BIOS)**

The BIOS is software that is stored on Read Only (ROM) chips. More recently it has been stored on flash memory. The BIOS software defines the capabilities of the system and what type of hardware is connected to it. Depending on the BIOS manufacturer, the size of the software residing on the ROM chip ranges from 256 KB to 1 MB. The test machine is an Award type BIOS and thus has 512 KB sized software. There are several tools, such as AwardFlash, WinFlash and AMI Flash that update the BIOS (eSupport.com 2004). These tools however cannot update the image of the BIOS but rather delete the original and replace it with a new version. A free open source version called UniFlash (Zary 2005) was discovered. This tool allowed the flashing of any type of BIOS regardless of its manufacturer. This tool will be used to attempt the flashing of the BIOS image to conduct further experiments on it.

## **5.3 Advanced Configuration and Power Interface (ACPI)**

The ACPI is an open industry specification co-developed by Hewlett-Packard, Intel, Microsoft, Phoenix, and Toshiba (Hewlett-Packard et al. 1999). It is an integral part of the OS; it manages all the power distribution to the computer's devices. If a device is unused, then power can be turned off to reduce power consumption (TheFreeDictionary.com 2005). The ACPI is comprised of an AML code. The code is stored in APCI Registers or APCI Tables; these however are relatively small in size and allow only the storing of code and no other data (Hewlett-Packard et al. 2004). It is possible to overwrite the APCI code in certain locations to perform a task in a different manner than its intended use, but this option is of little use (Heasman 2006). Due to this restriction it is unlikely to hold the marker for notebook computer identification.

## **5.4 Host Protected Area/Device Control Overlay (HPA/DCO)**

The HPA and the DCO are areas on certain hard disks, which are usually located at the end of a hard disk. The OS is unable to see these areas and can only recognise the size of the hard disk through calling a READ NATIVE MAX ADDRESS. If the HPA or DCO are set to be 4 GB in size out of a 400 GB hard disk, the hard disk would appear to the OS as 396 GB big. Another command call that can be used on HPAs is SET MAX ADDRESS. This call would allow someone to increase the size of the HPA and hide data in it (Gupta, Hoeschele & Rogers 2006). Hard disk utilising HPA/DCO at present are usually using them to hide data on them, mainly for system restoration. The HPA and DCO make the location a valid potential storage area without any size restrictions. One problem however does present itself, in that there is no guarantee that the HPA or DCO will be available on any and every hard disk a notebook computer is shipped with. Unlike the MBR and the BIOS the HPA and DCO are not integral parts of the system. Due to this inconsistency the HPA/DCO will not be researched further.



## **5.5 Hidden Partitions**

Hidden partitions are a more traditional storage area option, which are more often used to store restore points and OS files to facilitate safe restoring of corrupt OS's. A primary partition can be created using FDISK and then further subdivisions of the primary partitions can be made. Partitions such as extended partition or logical drives can be created (Microsoft 2005). If a partition is hidden it is not detectable by the operating system; this means that creating a minimum sized hidden partition will allow storing any identification information, and raising no suspicion with chunks of the hard disk missing. Using GDISK, part of Symantec Norton Ghost allows the user to use GDISK just like FDISK but with the added feature to create a hidden partition (Symantec 2006). One problem however is raised, should the entire hard disk be formatted the hidden partition will be along with the primary and any other partitions on the hard disk formatted. Thus this option is not well suited for persistent storage areas that would hold identification information.

## **6. EXPERIMENTAL RESULTS**

### **6.1 Methods of Storing and Retrieving Data**

As stated above, the way in which the identifier can be stored is dependent on the storage medium. The different approaches to inserting data have been described in the following subsections.

### **6.2 Master Boot Record (MBR)**

A literature review discovered that the MBR record is located on the hard disk and more specifically at the beginning of the memory address range of the hard disk. Thus the first 63 sectors of the hard disk are reserved for the boot loader code. The boot code requires only one of the 63 sectors and therefore the remaining 62 sectors remain reserved and unallocated. In order to store the marker into the unallocated sectors FDISK was considered. FDISK is a "disk-partitioning program used in DOS and several other operating systems to create the master boot record and allocate partitions for the operating system's use" (DataRecovery 2006). There are several free tools such as XFDISK (Boeck 2004) or MBRWizard (Layton 2003) out on the internet as well as commercial tools such as Super FDISK (PTDD-Soft 2005) or Partition Magic (Symantec 1998) on the market that offer FDISK-like features that allow the formatting of the MBR but rarely any allow the modification of the boot code itself. Once a tool was discovered, the source code of the identified tool called FDISK 1.21 (Reifsnnyder 2001) was used to firstly understand the techniques applied when writing to and reading from the hard disk in order to be able to perform the desired functions of the custom tool. Custom code was written to extract the MBR image, as well as to modify the location of the boot code identified as storage area, and to store it back onto the hard disk.

The first part of the custom code was to call C libraries which have direct access to the hard disk. The BIOS.H library allows the reading or writing of 512 bytes at a time from a sector of the hard disk. Since, for the purpose of this research, the location of the MBR is always the same on any hard disk, it was possible to extract the first sector and store it into a 512 byte sized array to store the marker and write it back in using the before mentioned library's write function. If the marker needs to be stored into other sectors, the tool would read in further sectors store the data in them and write them out back onto the hard disk.

The second part of the application was to use the 512 bytes and modify the memory location where the marker is to be stored. This was done using standard C libraries.

Figure 2 shows the location on the MBR where the marker was stored. While only a 28 digit code was stored for testing purposes, no problems were encountered during the storing of larger markers to the remaining 62 sectors.

To summarise the processes performed:

- Load test machine into DOS and run custom code
- Extract the MBR image from hard disk using custom code
- Using custom written code modify MBR image
- Reinsert the MBR image onto the hard disk using custom code
- Restart System, test MBR operation, success if OS loading

mbr.mbr																	
Offset	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F	
00000000	FA	33	C0	8E	D0	BC	00	7C	FB	8E	D8	BE	00	7C	8E	C0	ú3Á1B4  ú10%  IÁ
00000010	BF	00	06	B9	00	01	F3	A5	E9	00	8A	BE	AE	07	B9	04	ú ' ' ówé 1%0 ' 1
00000020	00	83	C6	10	80	3C	80	74	09	80	3C	00	75	5D	E2	F1	1E  <it  < u án
00000030	CD	18	8B	D6	49	74	0A	83	C6	10	80	3C	00	75	4C	E2	í  ÓIt 1E  < uLá
00000040	F6	EF	05	00	EB	00	7C	8B	F2	8B	14	8B	4C	02	CC	E8	öú »  töi IL Ié
00000050	58	00	80	3E	A9	06	01	75	0E	8B	44	08	8B	54	0A	B9	X  >@ u  D IT ' 1
00000060	01	80	E8	8F	00	EB	10	B8	01	02	CD	13	73	09	4F	74	èi   è .   s Ot
00000070	1F	33	C0	CD	13	EB	F0	26	81	BF	FE	01	55	AA	75	15	3ÁI èè& çp U3u
00000080	BF	00	7C	B8	50	00	8B	EE	06	53	CB	BE	35	07	EB	08	ú  ,P  i SE35 è
00000090	BE	4F	07	EB	03	BE	66	07	B4	0E	BB	07	00	8A	04	CD	%O è %f ' »   I
000000A0	10	46	80	3C	00	75	F6	EB	FE	00	60	B4	41	BB	AA	55	F < uèèp ' Áv3U
000000B0	CD	13	72	3C	81	FB	55	AA	75	36	83	E1	01	74	31	B4	í r< áU3u6 á t1'
000000C0	48	BE	10	80	C7	06	10	80	1A	00	CD	13	B4	08	CD	13	H%  ç     '
000000D0	8A	C6	FE	C0	8A	D9	80	E3	3F	F6	E3	8B	C8	A1	20	80	1E3Á U18?ó& E
000000E0	8B	16	22	80	F7	F1	3D	00	04	7E	05	C6	06	A9	06	01	"  s-ñ= ~ % @
000000F0	61	C3	00	00	60	0E	1F	51	32	ED	BE	00	80	C7	04	10	aÁ ' Q2i%  ç
00000100	00	89	4C	02	89	5C	04	8C	44	06	89	44	08	89	54	0A	IL   \  D  D  T
00000110	C7	44	0C	00	00	C7	44	0E	00	00	59	BF	05	00	B4	42	çD çD Yé ' B
00000120	8A	D5	CD	13	73	0D	33	C0	CD	13	4F	75	F1	BE	4F	07	ÓI s 3ÁI Ouñ%O
00000130	E8	65	FF	61	C3	0A	0D	50	61	72	74	69	74	69	6F	6E	èèy3Á Partition
00000140	20	74	61	62	6C	65	20	69	6E	76	61	6C	69	64	00	0A	table invalid
00000150	0D	45	72	72	6F	72	20	72	65	61	64	69	6E	67	20	73	Error reading s
00000160	65	63	74	6F	72	00	0A	0D	4F	70	65	72	61	74	69	6E	ector Operatin
00000170	67	20	73	79	73	74	65	6D	20	6D	69	73	73	69	6E	67	g system missing
00000180	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	
00000190	00	00	00	00	00	00	00	00	00	00	AF	31	32	33	34	35	12345
000001A0	36	37	38	39	30	31	32	33	34	35	36	37	38	39	30	31	6789012345678901
000001B0	32	33	34	35	36	37	38	38	EF	8F	03	8F	00	00	80	01	23456788i
000001C0	01	00	0C	FE	FF	FF	3F	00	00	00	02	70	4A	04	00	00	þýý? pJ
000001D0	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	
000001E0	C1	FF	0C	FE	FF	FF	41	70	4A	04	FE	A3	5D	00	00	00	Áý þýýApJ þé
000001F0	C1	FF	0E	FE	FF	FF	3F	14	A8	04	C1	3E	00	00	55	AA	Áý þýý? ' Á> U3

Figure 2: Master Boot Record screenshot circled area indicates stored marker

### **6.3 Basic Input Output System (BIOS)**

The supporting documentation supplied by the manufacturers of different BIOS's indicate that, for each BIOS manufacturer, there is also a custom tool that performs the updating of the BIOS (eSupport.com 2004) To be able to make use of these tools to attempt the modification of the BIOS rather than the complete update, it was necessary to completely understand the techniques and the processes involved. The large number of different BIOS manufactures meant that there would also be countless flashing tools, and that meant that it was not possible to pursue this approach. After an extensive online search, an open source, universal, Pascal based BIOS flashing tool called UniFlash (Zary 2005) that was capable of extracting and reinserting a BIOS image onto the CMOS Chip was discovered. While this tool was not able to modify the BIOS image itself, it did allow the extraction of any type of BIOS image. Since the image itself was simply a binary text file, it was possible to modify it with a standard hex editor. Furthermore, to refrain from using too many third party tools/applications, a small C based program was developed to access this image and add the marker to a specified location in the image. Due to the size of the BIOS being approximately 512 KB, it was not possible to load a single array with the image content like in the case of the MBR, but rather a temporary array that would load 512 bytes at the time until the area identified as the storage location for the marker was reached, modified and once again stored into the original image for reinsertion to the CMOS Chip. Once the marker has been successfully stored in the image, the universal tool reinserted the image into the Chip. Figure 3 shows the BIOS image and the marker stored within it. To ensure that the BIOS was still functioning the test system was restarted and observed. The system booted into the OS without any complications indicating that the change to the image resulted in no abnormal functionality.

To summarise the processes performed:

- Load test machine into DOS and run custom uniflash
- Extract the BIOS image from the CMOS chip using custom uniflash tool
- Using custom written code modify BIOS image
- Reinsert the BIOS image onto the chip using custom UniFlash tool
- Restart System, test BIOS operation, success if OS loading

BACKUP.BIN																
Offset	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0007FDF0	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
0007FE00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
0007FE10	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
0007FE20	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
0007FE30	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
0007FE40	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
0007FE50	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
0007FE60	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
0007FE70	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
0007FE80	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
0007FE90	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
0007FEA0	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
0007FEB0	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
0007FEC0	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
0007FED0	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
0007FEE0	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
0007FEF0	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
0007FF00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
0007FF10	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
0007FF20	00	00	AF	31	32	33	34	35	36	37	38	39	30	31	32	33
0007FF30	34	35	36	37	38	39	30	31	32	33	34	35	36	37	38	EF
0007FF40	1A	C2	64	DD	41	4D	49	42	4F	4F	54	20	52	4F	4D	00
0007FF50	00	00	00	00	00	00	00	00	00	1D	47	00	00	00	00	12
0007FF60	00	02	1D	51	00	00	00	00	00	00	00	00	00	00	00	00
0007FF70	E8	A4	0A	CB	E8	67	15	CB	E8	79	15	CB	E8	9E	15	CB
0007FF80	E8	B0	15	CB	E8	40	15	CB	E8	3C	15	CB	00	00	00	00
0007FF90	00	00	BE	98	FF	E9	17	1F	CB	00	00	00	00	00	00	00
0007FFA0	EA	F3	04	00	00	00	07	00	39	01	00	00	00	00	00	00
0007FFB0	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
0007FFC0	00	00	00	00	00	00	00	00	00	00	00	00	00	E9	38	00
0007FFD0	00	F4	FC	FF	24	53	49	50	00	0C	00	00	00	00	00	00
0007FFE0	4B	54	36	30	30	33	30	30	00	00	00	00	00	00	00	00
0007FFF0	EA	CD	FF	00	F0	30	33	2F	30	33	2F	30	34	00	FC	00

Figure 3: BIOS screenshot circled area shows the stored marker.

## 7. EVALUATION

### 7.1 Master Boot Record (MBR)

The Master Boot Record fully addresses the identified set of criteria which will fully determine whether the non-volatile memory is to be chosen as a valid location for the storing of the marker. It successfully addressed storage size, by discovering that there is approximately 31 KB of unallocated space, the location can be written to as proven in the testing stage and also that it maintained persistent state after extensive testing by extracting the MBR image and checking that the marker was still in the location it was stored in and that it was not modified. Thus the Master Boot Record was shown to be a valid non-volatile storage location for safe storing of notebook computer identification information.

### 7.2 Basic Input Output System (BIOS)

The BIOS was a more complicated non-volatile memory to consider as a location capable of storing markers. Due to the number of different BIOS

manufacturers, it is difficult to test all types and confirm them as a valid storage location. As a proof of concept, analysis of a test machine's Award BIOS was sufficient enough to at least prove that it was possible to successfully store a marker onto the CMOS chip.

In answering the first question, "What types of non-volatile storage areas exist within a notebook computer running a Windows XP Operating System?", the BIOS was identified as a non-volatile memory and in answering our second research question, "Is it possible to write to these storage areas?", we showed that it was possible to write to the BIOS image rather than just removing one version of the image and storing a new version all together. Question three, "Which storage location can hold enough data to uniquely identify a notebook computer?", with similar results to the MBR, was shown as a valid non-volatile storage by modifying the BIOS image and restarting the test machine, running it for a set period of time and extracting the image out of the CMOS chip again. Verification could then be achieved by verifying that the marker did not change in location or size.

The results of the BIOS testing stage were tested against the set selection criteria, which consisted of the following: storage has enough unallocated space, storage can be written to and storage maintains persistent state, and the BIOS implementation was found to be a non-volatile memory that has enough sufficient amount of unallocated space to which the marker can be written to. This was discovered through the literature review and the testing stage and the total amount of unallocated space in the Award BIOS was found to be roughly 64 KB of space. The second selection criterion was successfully addressed as mentioned previously by writing to the BIOS image and reinserting it on to the CMOS chip. And the third criterion was addressed and said before by writing to the BIOS and running the system to test for anomalies and at the same time test the possibility of the marker of being modified or removed from the BIOS. After several restarts the contents of the BIOS image did not change including the location and size of the stored marker. Thus the BIOS was found to be a valid non-volatile storage location capable of storing notebook information.

## **8. PROOF OF CONCEPT APPLICATION**

By implementing the identified non-volatile storage areas and using the unique marker, notebook computer identification can be achieved by formulating a concept tool and applying these features. This tool, would through a number of other tools also generated for the concept tool allow the identification of a notebook computer to its owner. The steps and tools involved in this process are illustrated below in Figure 4.

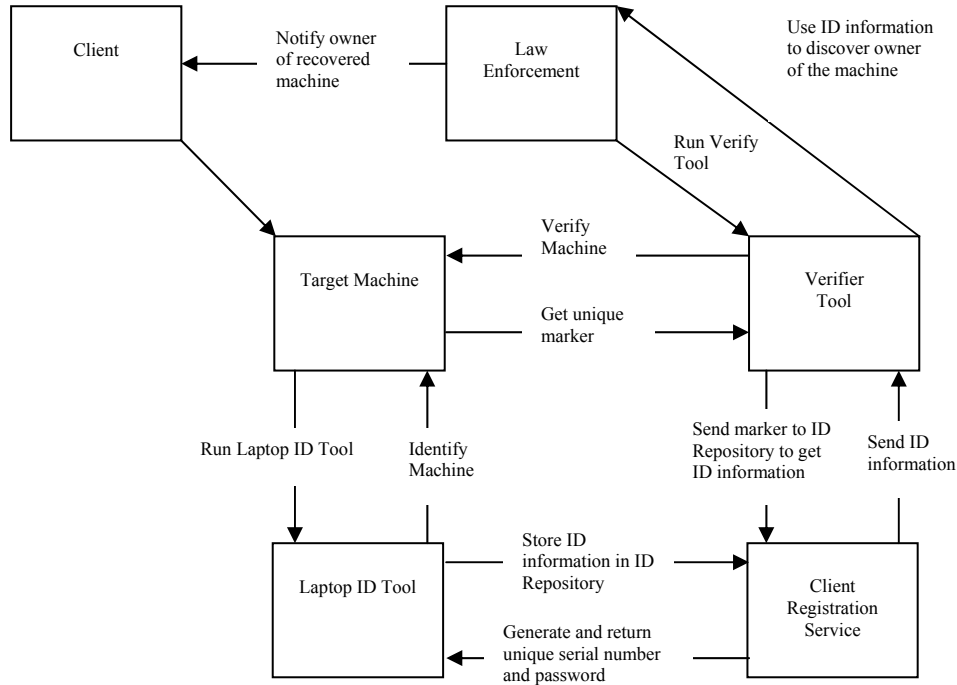


Figure 4: Architectural/procedural overview of the concept application

## 9. DISCUSSION

Through the literature review, several potential non-volatile storage area locations were identified. Further research on the individual storage areas resulted in identifying that the storage areas are capable of holding the identification data, that they can be written to and that these areas are persistent in state. Therefore, any data stored in such areas would remain there even after the target notebook computer is shut down. These requirements greatly reduced the number from those initially proposed to only the MBR and the BIOS. After extensive research, it became evident that should these memories be used to store information about a notebook computer in conjunction with a forensic tool, several issues needed addressing. There were some limitations both to the storage locations and the concept tools that were designed.

### 9.1 Limitations in Non-volatile Storage area locations

The two identified non-volatile storage area locations that conform to the set criteria have a number of limitations that potentially introduce new issues that need to be addressed before they can be used in a forensic application. Firstly the nature of the storage area areas, at least in the case of the BIOS, means that there will be regular updates to BIOS boot code. These updates would result in

the flashing of the BIOS image with the stored marker being lost and re-flashed with a newer version. While these updates are not frequent, and most users do not perform such updates unless a major fault was discovered by the manufacturer, and the owner is instructed to perform these updates, it is still necessary to address this problem. One further limitation affecting the MBR in particular is the use of Windows FDISK feature “/mbr” when formatting a hard drive. While this feature is not part of a standard reformat, it can however be used to format the MBR as well. This presents a problem to the safety of the marker stored in the MBR. This also would need to be addressed if future research is to be conducted.

## **9.2 Forensic Soundness of Proof-of-Concept Application**

Due to the current implementation of the marker in the MBR and the BIOS, there are no mechanisms in place to prevent a person with the right knowledge from removing the marker from the BIOS or MBR. This, however, has been stated as necessary future research. As a concept tool, this application has not been tested against guidelines for forensic tool development, and while no changes have been recorded to the state of the target machine during an investigation, no forensically sound methods were applied to verify these outcomes. Future work may address this as a potential issue. Once these problems have been addressed the tool is potentially of immense benefit to the forensic community and also may increase the return rate of recovered notebook computers.

## **9.3 Feasibility of a Universal Software Application**

As discussed in the literature review, the problem of notebook computer identification is a growing concern for the law enforcement. While the number of recovered notebook computers increases the lack of uniquely identifiable markers on these machines makes the task of returning the machine to their owners impossible, leaving the law enforcement with rooms full of unclaimed machines. Being able to provide a tool that will not only enable police to identify the machines to their owners but also speed up this identification process makes this concept a promising area of research.

The tools discussed, compared with the proof of concept application, appear quite dissimilar. The concept application focuses on the identification of the machine, so that when it is recovered by the law enforcement, it can be returned to its rightful owner. The commercial applications focus more on the tracking of the machine through the use of a tracking server and the encryption of the data stored on the machine. The commercial applications are focusing solely on the commercial or corporate sector where the data is valuable but not the hardware. The concept application is targeting users who use the notebook computers for private use. This should allow the tool if commercialised to be a leader in the market.

## **10. CONCLUSION**

This research identified several valid locations in which it is possible to store the marker to uniquely identify a notebook computer. As stated above, we have also through a literature review discovered the storage area locations, possibly non-volatile in nature, that could be used to identify a notebook computer through safe storage area of unique notebook identification information. Several types of storage areas were identified, including traditional storage areas such as hidden files and hidden partitions as well as non-traditional potential storage areas like the ACPI, HPA, DCO, as well as the BIOS and the MBR. Due to the nature of some of these non-volatile potential storage areas, this research indicated that they were unlikely to hold enough data if any for notebook identification. Regarding the traditional storage areas such as hidden files and the hidden partitions, a simple hard disk format would have removed these without any problem, making these a useless option. This narrowed the list of possible storage areas to only the MBR and the BIOS. We then focused on these two locations to test these against identified criteria. During the evaluation stage of the research, the two storage areas were tested and found to satisfy firstly that the storage can hold enough data to uniquely identify the notebook computer, secondly that the storage can be written to and lastly that the storage can maintain its state even after power downs. These locations have also been shown not to be erasable through conventional means and require direct and advanced interaction to remove the marker. This requires the person to know exactly where the marker is and the methods that would remove the markers.

### **10.1 Future Directions**

This project resulted in the design of a proof-of-concept tool to prove the feasibility of the use of non-volatile storage areas as locations that would hold the marker to uniquely identify a notebook computer. This tool successfully stored information to the identified locations but did not incorporate fail safes against purposeful or accidental deletion of the markers. Both the BIOS and the MBR possess a checksum that is run every time the notebook computer is booted to test for malicious code in the respective locations. These checksums would need modification to incorporate the stored marker as an integral part of the BIOS without which the OS could not load. Thus any purposeful attempts to remove the marker would result in rendering the OS unusable. Furthermore the software designed has potential use in forensic investigations and as such it needs to be forensically sound. It needs to be tested against accepted standards for the design and development of a forensically sound tool.



## 11. REFERENCES:

- AbsoluteSoftware (2006a), AbsoluteTrack DS, <<http://www.absolute.com/PDF/AbsoluteTrackDS.pdf>>. accessed 22nd October 2006.
- AbsoluteSoftware (2006b). Computrace LoJack for Laptops, <<http://www.lojackforlaptops.com/learn-more-lojack-for-laptops.asp>>. accessed 22nd October 2006.
- Armstrong, H, Wynne, M & O'Shea, T. (2004). 'Who has the keys to the vault? Protecting secrets on laptops'. Proceedings of the 2004 IEEE IA Workshop, USMA WestPoint New York.
- Beachhead, S (2006). Mobile Data Vulnerability, <<http://www.beachheadsolutions.com/mobile.php>>.. accessed 20th, October 2006.
- Boeck, H (2004). xTended FDISK 0.9.3, <<http://www.mecronome.de/xfdisk/download.php>>. accessed 4th, June 2006.
- Bursky, D (2003). 'Nonvolatile memory: more than a flash in the pan', Electronic Design, vol. 51, no. 15, pp. 41-6.
- DataRecovery, O (2006). FDISK Glossary, <<http://www.ontrack.com/glossary/>>. accessed 13th August 2006..
- DeMaria, MJ (2002). 'Gone in 6.0 seconds [laptop security]', Network Computing, vol. 13, no. 20, pp. 77-90.
- eSupport.com (2004). BIOS Utilities - Flash Loaders, <http://www.unicore.com/techsupport/award/awardutils.htm>>.accessed 5th July 2006,
- Gershteyn, P, Davis, M & Shenoi, S (2006), Detection and recovery of Hidden Data from Award BIOS Chips, Springer, Dordrecht, The Netherlands,
- Gupta, MR, Hoeschele, MD & Rogers, MK. (2006). 'Hidden Disk Areas: HPA and DCO', International Journal of Digital Evidence, Fall, p. 8.
- Heasman, J (2006). Implementing and Detecting an ACPI BIOS Rootkit, Netherlands.
- Hewlett-Packard, Intel, Microsoft, Phoenix & Toshiba (1999). ACPI - Advanced Configuration & Power Interface, <<http://www.acpi.info/>>., accessed 30th, June 2006.
- Hewlett-Packard, Intel, Microsoft, Phoenix & Toshiba (2004). ACPI Specifications 3.0a, <<http://www.acpi.info/DOWNLOADS/ACPIspec30.pdf>>.accessed 30th, June 2006.

- Layton, R (2003). MBRWizard 1.53, <<http://www.geocities.com/mbrwizard/>>. accessed 3rd, June 2006.
- Microsoft (2005). How to Use the Fdisk Tool and the Format Tool to Partition or Repartition a Hard Disk, Microsoft Corporation, <<http://support.microsoft.com/kb/255867>>. accessed 3rd, July 2006.
- NTFS.COM (2006). Master Boot Record (MBR), NTFS.COM, <<http://www.ntfs.com/mbr.htm>>. accessed 2<sup>nd</sup> July 2006.
- PointSec (2006). Security Products Laptop, <<http://www.pointsec.com/products/laptop/>>. accessed 21st, October 2006.
- PTDD-Soft (2005). Super FDISK 1.0, <<http://www.ptdd.com/manual2.htm>>. accessed 1st June 2006.
- ranish.com (1998). Partitioning Primer. <<http://www.ranish.com/part/primer.htm>>. accessed 1st, June 2006.
- Reifsnnyder, BE (2001). Free FDISK 1.21, <<http://www.23cc.com/freefdisk/index.htm>>. accessed 1st, June 2006.
- Slay, J, Broucek, V, Hannan, M & Turner, P (2004). 'Developing Forensic Computing Tools and Techniques within a holistic framework: an Australian Approach', in Proceedings of the 2004 IEEE IA Workshop, USMA WestPoint New York.
- Symantec (2006). Introduction To GDISK, <<http://service1.symantec.com/SUPPORT/ghost.nsf/docid/1998081911285525/>>. accessed 1st June 2006.
- Symantec (1998). Partition Magic, <[http://www.symantec.com/home\\_homeoffice/products/features.jsp?pcid=sp&pvid=pm80](http://www.symantec.com/home_homeoffice/products/features.jsp?pcid=sp&pvid=pm80)>. accessed 1st June 2006.
- TheFreeDictionary.com (2005). ACPI, Farlex, Inc. <<http://computing-dictionary.thefreedictionary.com/ACPI>>. accessed 1st June 2006.
- Zary, O (2005), UniFlash, <<http://www.uniflash.org/>>. accessed 1st June 2006.

