2009

A Software Response to Hard Drive Encryption

Justin John McCarty III
West Virginia University

Follow this and additional works at: <https://researchrepository.wvu.edu/etd>

Recommended Citation

McCarty, Justin John III, "A Software Response to Hard Drive Encryption" (2009). *Graduate Theses, Dissertations, and Problem Reports*. 2829.
<https://researchrepository.wvu.edu/etd/2829>

This Thesis is protected by copyright and/or related rights. It has been brought to you by the The Research Repository @ WVU with permission from the rights-holder(s). You are free to use this Thesis in any way that is permitted by the copyright and related rights legislation that applies to your use. For other uses you must obtain permission from the rights-holder(s) directly, unless additional rights are indicated by a Creative Commons license in the record and/ or on the work itself. This Thesis has been accepted for inclusion in WVU Graduate Theses, Dissertations, and Problem Reports collection by an authorized administrator of The Research Repository @ WVU. For more information, please contact researchrepository@mail.wvu.edu.

A Software Response to Hard Drive Encryption

Justin John McCarty III

**Thesis submitted to the
College of Engineering and Mineral Resources
at West Virginia University
in partial fulfillment of the requirements for the degree of**

Master of Science in Electrical Engineering

Dr. Roy Nutter, Ph.D., Chair

Dr. Jim Mooney, Ph.D.

Don McLaughlin, Associate Professor

Lane Department of Computer Science and Electrical Engineering

Morgantown, West Virginia

2009

**Keywords: TPM, Encryption, Hard Drive Encryption, Computer Forensics, LEAP, Live
Analysis**

Copyright© 2009 West Virginia University

Abstract

A Software Response to Hard Drive Encryption

Justin John McCarty III

Current computer forensic practices primarily advise analysts to review and search all digital evidence within designated labs. This practice does not take into account the possibility of encryption software being present on the suspect machine. As a result, possible evidence may become unrecoverable to the investigator. This paper will review the current standard operating procedures used by law enforcement, complications with current procedures with regard to encryption, and the design of a free to use utility for law enforcement to facilitate live analysis.

Table of Contents

Abstract	ii
Table of Figures.....	v
1 Introduction	1
1.1 Methods of Encryption	1
1.1.1 Advanced Encryption Standard	1
1.1.2 RSA Encryption	1
1.1.3 Hardware Encryption of Storage	2
1.1.4 Software Encryption of Storage	3
1.2 TPM Explanation.....	4
1.2.1 TPM Basics.....	4
1.2.2 Uses of TPM.....	4
1.2.3 Keys.....	6
1.3 BitLocker	7
1.3.1 Technique used to encrypt.....	8
1.3.2 Protection methods available	8
1.4 Overview of current law enforcement practices.....	9
1.4.1 Warrantless Search (“Knock-and-talk”)	9
1.4.2 Warrant.....	10
1.5 Potential problems due to Encryption with evidence collection	10
1.5.1 Loss of keys.....	10
1.5.2 Transparent operation	11
1.5.3 Single Chip Marriage.....	11
2 Investigation Techniques	12
2.1 Conventional methods of finding digital evidence.....	12
2.1.1 Linux Boot	12
2.1.2 Cold Image capture.....	12
2.2 Live methods of finding digital evidence	12
2.2.1 Live image capture.....	13
2.2.2 Live analysis.....	13

3	Statement of purpose.....	13
4	Literature Review	13
5	Overview of LEAP	14
5.1	Introduction	14
5.1.1	Need.....	14
5.2	Objective	14
5.2.1	Abbreviations/Definitions	15
5.3	Requirements.....	15
5.3.1	Implementation Requirements	15
5.3.2	Engineering Requirements.....	16
5.3.3	Requirements Specifications.....	16
5.3.4	Competitive Benchmarks	17
5.4	Constraints	18
5.4.1	Economic.....	18
5.4.2	Ethics.....	18
5.4.3	Social.....	18
6	LEAP 2.0 Detailed Design	18
6.1	Execution of Application	19
6.2	Start Up Operations	19
6.2.1	Encryption Check	20
6.2.2	Information Retrieval.....	22
6.2.3	Report	22
6.2.4	System Analysis	23
6.3	Component Breakdown	24
6.3.1	Graphic Viewer	25
6.3.2	Parse Flash Cookies.....	27
6.3.3	File Explorer.....	27
6.3.4	View/Modify Report	30
7	Conclusions	30
8	Acknowledgements.....	30
9	Appendix.....	31
9.1	Engineering/Law Enforcement Need Correlation chart	31

9.2	Requirements Specification	31
10	Works Cited.....	32

Table of Figures

<i>Figure 1: Sample Boot Process</i>	<i>6</i>
<i>Figure 2: LEAP Device in My Computer.....</i>	<i>19</i>
<i>Figure 3: Administrative Request</i>	<i>20</i>
<i>Figure 4: Encryption Check Request</i>	<i>21</i>
<i>Figure 5: Encryption Results.....</i>	<i>22</i>
<i>Figure 6: Report Information Form.....</i>	<i>23</i>
<i>Figure 7: Program Selection.....</i>	<i>24</i>
<i>Figure 8: System Analysis Main Screen.....</i>	<i>24</i>
<i>Figure 9: LEAP Navigator</i>	<i>25</i>
<i>Figure 10: Main Graphics Viewer Interface</i>	<i>25</i>
<i>Figure 11: Picture Details.....</i>	<i>26</i>
<i>Figure 12: Primary File Search Interface.....</i>	<i>28</i>
<i>Figure 13: Hash Entry</i>	<i>30</i>

1 Introduction

1.1 Methods of Encryption

This section contains a broad overview of the terms and technologies referenced and examined in this paper. These explanations are intended to provide the reader with the minimal working knowledge needed to understand a software response to hardware encryption. Full detailed descriptions are outside the scope of this document.

1.1.1 Advanced Encryption Standard

The Advanced Encryption Standard (AES) was introduced by the National Institute of Standards and Technology (NIST) on November 26, 2001. The AES algorithm (originally named Rijndael) is a symmetrical key algorithm that is designed to operate on 128-bit blocks of data (National Institute of Standards and Technology, 2001). It should also be noted that the AES algorithm is both publicly available and authorized by the NSA for “Top Secret” information.

1.1.1.1 AES Multi-Bit Encryptions Keys

Although the AES algorithm is designed to operate on 128-bit blocks of data, the encryption key used may appear in three different “flavors” for use (National Institute of Standards and Technology, 2001). These “flavors” represent the different possible encryption key length to be used in the encrypting and decrypting process. The key may be 128, 196, or 256 bits in length. As such, the total number of possible key combinations is approximately $3.4e^{38}$, $1.0e^{59}$, or $1.15e^{77}$ for each “flavor” respectively.

1.1.1.2 Symmetrical Key Use

The AES algorithm is designed to utilize a small, single key for both the encryption and decryption of data. This provides both a large advantage and disadvantage for the algorithm. The primary disadvantage to this approach is that the initial party would need to either use a pre-arranged key decided upon by all involved parties or would need to be able to exchange the key in confidence with other users. Both approaches are vulnerable to key interception or miscommunication and jeopardize the encrypted data. However, in exchange, the data may be encrypted and decrypted quickly with minimal equipment requirements. The AES algorithm is designed to be used by low powered processors and embedded smart cards (KetuWare).

1.1.1.3 Used by many software programs

Due to the availability, low resource cost, and strength of the algorithm, AES is currently used by many publicly available software programs using encryption.

1.1.2 RSA Encryption

The RSA algorithm was developed in 1977 by Ronald L. Rivest, Adi Shamir, and Leonard Adleman (RSA Laboratories). The RSA algorithm is designed to use an asymmetrical key pair to provide both security and authentication. The keys used in RSA are generally either 1024 or 2048 bits in length.

1.1.2.1 Asymmetrical keys

Asymmetrical key algorithms utilize two keys to secure data. These keys are referred to as being either public or private keys. Each key is both unique in value and meaning when used with the RSA algorithm. The private key is designed to be used as an authentication marker for a user. The idea is that the key will uniquely identify a user and is kept secret. As such the key is never to leave the host user or device. While at the same time each user provides a public key. This key is used to verify the digital signature for a private key (RSA Laboratories, 2002).

In the encryption and decryption process, the public key (known to everyone) is used for encrypting a message. This message may now only be decrypted using the matching private key of a user. This is used to ensure that only the desired party can decrypt the data. However, due to the risk of “fake” messages being sent by other users, a “signing” process may be used by the two parties.

In the signing process, a hash value of the message and a user’s private key are combined to produce a digital signature. This signature is then verified by the receiving party by using the sender’s public key and signature to create a new hash value. If this new value and the hash value of the message match, then the user can be sure that the message is indeed correct and secure.

1.1.2.2 Scalability

As stated, the RSA algorithm uses relatively large encryption keys (1024 or 2048 bit) to encrypt and decrypt data. In addition, the complexity of the algorithm results in a more hardware intensive process that may become cumbersome for large quantities of data. As a result, it is common for data to be encrypted with a strong symmetrical key (AES) and to then transfer the symmetrical key using a public/private key pair. As a result, the data may be quickly encrypted/decrypted while at the same time allowing the key to be transferred securely with the users able to verify the authenticity of the data.

1.1.3 Hardware Encryption of Storage

Hardware based encryption is a method of encryption that implements an encryption algorithm in a confined physical device. This device may exist in many different forms but, will typically bypass the use of a system’s CPU and RAM and is not completely accessible by an Operating System. These hardware devices can provide a method of secure communication between components (both hardware and software) without exposing the data to unauthorized users.

Examples of hardware encryption include the use of TPM chips (explained in section 1.2) and secure hard drive controller boards. “Secured” hard drives may operate using many authentication methods (if enabled) to ensure restricted access to the data. This hardware authentication device may be as simple as requiring a user to provide a password to the drive during the boot process (Note: This is separate from any BIOS password security). In this form of operation, the drive may be moved between computers (that support the drive) and correctly used provided that user knows the password. However, if the drive uses another hardware component (TPM, Motherboard, etc.) in the authentication process this is not possible. Under these circumstances the drive will only operate provided that the

correct hardware is present and the user knows the password (if needed). In this way, the hard drive is typically referred to as “married” to the computer.

1.1.4 Software Encryption of Storage

Contrary to hardware encryption, software encryption is done in conjunction with a computer system’s operating system and does not rely on specific system hardware components for authentication (in a pure form) but instead is pure software. This enables the user with greater flexibility when encrypting and decrypting data. A user may choose from a broad range of software applications for software encryption. Some of these commonly used software encryption applications include:

- TrueCrypt
- PGP
- Wave Systems Corps.
- Windows BitLocker
- AxeCrypt

These applications provide users with the ability to use several different encryption methods for protecting data (including combinations of these). These encryption methods include:

- DES
- Triple-DES
- BlowFish
- AES
- RSA
- Etc.

1.1.4.1 Key Storage

Unlike hardware encryption methods, the encryption keys used by software methods do not possess inherent security when keeping keys a secret. The keys must be kept accessible at all times when the program is in use and must be stored “permanently” for future use if the keys are to be reused. As a result, these keys are often stored in files that are protected using a protected algorithm that requires little or no user input to decrypt. When the program is in use, the encryption keys may be kept in protected areas of RAM so that other programs do not have direct access to them (Hargreaves & Chivers, 2008).

1.1.4.2 Examples of software encryption capabilities

Software encryption programs offer a number of different features in addition to plain file encryption. The program interacts with the host operating system and has the ability to access restricted areas of the system. Consider “TrueCrypt”. “TrueCrypt” features can create encrypted “container files” and create a second bootable (false) operating system (Hargreaves & Chivers, 2008).

The “container file” is an encrypted space on the hard drive that appears to be a single unreadable file that actually contains other files. These files are accessible once the software “mounts” the opened (decrypted) file. The false operating system option provided by “TrueCrypt” is a form of full disk encryption. A user can have a main (true) operating system that is protected by the “TrueCrypt” encryption process. A second “clean” (false) operating system is simultaneously created on the disk in a separate space. When the system is started up, the user will be prompted to provide a startup password. If a user provides the correct password the “true” operating system will boot. If the wrong password is used, the “false” operating system (devoid of any hidden or illegal files) is booted.

1.2 TPM Explanation

Trusted computing is the ability to ensure that any system or program is exactly what it claims to be.

1.2.1 TPM Basics

TPM chips are the result of the collaboration among several industry representatives over the course of several years. This group, referred to as the Trusted Computing Group (TCG), is comprised of representatives from both hardware manufacturers and software developers. These representatives include AMD, Hewlett-Packard, IBM, Infineon, Intel, Lenovo Holdings Limited, Microsoft, and Sun Microsystems. The group’s goal was to create a chip that allowed a high level of security for users and still provided a high level of privacy.

All TPM chips that are manufactured and sold must conform to the standards set forth by the TCG. The standards define not only the hardware specifications of the chip but also outline functionality that must be available. In the original standards, referred to as 1.1b, functionality was defined but not how the functionality had to be implemented. As a result, any program developed to use a TPM chip had to be designed around a specific chip. This led to confusion between developers and reduced the “real world” effectiveness of the chips. In order to promote development and make the chips more appealing for use, the group developed the 1.2 specifications.

The 1.2 specifications refined and expanded the 1.1b specifications in two major ways. First, the 1.2 specifications do not define how a chip must implement functionality from a software point of view. In this regard, all 1.2 chips use the same programming interface for input and output. How the chip implements each function internally is more or less up to the manufacturer. In this way programs can be developed for all 1.2 TPM chips. In the 1.2 specifications, the TCG expanded the functionality of the chips to provide more uses and improve the protection of user privacy.

Though TPM chips are now being manufactured under 1.2 specifications, the specifications are not yet complete and are still being developed. Therefore developers typically are only using the functions provided in 1.2 that are backwards compatible with 1.1b.

1.2.2 Uses of TPM

The functionality of TPM chips is rather open ended. The TCG wanted the chips to be able to be used under any number of circumstances and by any number of programs. With regards to this paper,

only three commonly used functions will be analyzed. These functions are the identification of other trusted systems, sealing a system, and file encryption.

The ability to identify other trusted systems encompasses both hardware and software. Hardware identification comes in several forms with TPM chips. At a local level, it is the goal of the TCG for TPM chips to be used in any major component of a computer system. The chips would be able to interact and verify one another for further communication. An example would be if a TPM chip on a motherboard was programmed only to allow certain hard drives to be used. Whenever a hard drive would be plugged into the system, the motherboard would verify or reject the drive based on the hard drives own TPM chip. In this way the system could be sure that all communication with the hard drive was secure and legitimate. Another hardware implementation of system identification would be remote computer usage. A system would only remotely communicate with another system if it could be identified as secure by the chip. From the software perspective, a file could be coded in such a way that only authorized programs (verified with the TPM) could access it.

The concept of “sealing” a system, using TPM, seems relatively simple but is harder to implement. A system is sealed by measuring, or hashing, different pieces of the system and storing the resulting hashes on the TPM chip in areas called Platform Configuration Registers (PCRs). Then, when the system is restarted a hand-off style boot is preformed. The boot process is a hand-off due to the fact that before each step of the boot process, a hash is taken and compared against the hash value stored on the TPM. Using Figure 1 as an example, the process can be more easily explained. The boot process starts with the “Core Root of Trust Measurement” (CRTM). This area is always considered to be trust worthy during the boot process. Before moving to the second stage, the BIOS is hashed (arrow 1) and compared to the stored value in the PCR (arrow 2). If the value matches, the boot process allows the BIOS to take control (arrow 3) and proceed with booting. If the value does not match, at any point, the system will not boot.

Figure 1: Sample Boot Process

When at the BIOS, the systems hardware and boot loader are hashed (arrows 4 and 5) and again compared to values stored in the PCRs (arrow 6). This part of the process allows the system to verify that no hardware has been added or removed from the system and be potentially compromised. At the same time, the integrity of the boot loader is confirmed to be free of malicious code. This processes of measuring and comparing continues until the system is able to finally load the operating system.

The final major use for TPM, as discussed by this document, is file encryption. File encryption has been around for years. With the introduction of TPM chips, files can be secured in a much stronger fashion. This is due to the fact that when files are encrypted or decrypted using TPM it is done completely within the chip itself. In this way, the encryption key used by the chip cannot be easily found or seen “in the clear”. The process is independent of both processor and RAM.

1.2.3 Keys

In order to fulfill the requirements of trust, verification, and privacy as stated before, TPM chips use a series of different encryption keys. Each action taken by a TPM chip is associated with a set of keys at all times. The style of key generated and used by the chip

Keys stored on a TPM chip are classified as either migratable or non-migratable by nature. When a key is classified as migratable it means that it is possible to transfer the key from one TPM chip to another. In this way keys used for functions like file encryption can be backed up in case of hardware malfunction. It should be noted that while migrating a key, privacy is still protected at all times as the keys are never seen “in the clear”. When a key is classified as non-migratable it can never leave the chip. As such, if a file had to be decrypted using one of these keys and the chip failed, the file would essentially be lost forever.

When using encryption, an item is locked using either a symmetrical or asymmetrical key system. In a symmetrical key system the same key used to encrypt a file is used to decrypt it. This is analogous to locking a door with a simple dead bolt. Unlocking or locking the door requires a single style key. However, it is possible for multiple copies of a key to exist and all can be used with the lock. In an asymmetrical system key pairs are used to encrypt and decrypt information. These pairs work in a one way only fashion. For example, if key “A” is used to encrypt (lock) a file only key “B” can be used to decrypt (unlock) the file. In the same way, if key “B” encrypts a file, only key “A” can then decrypt it. As such, a greater inherent safety factor is provided by this method. Therefore TPM chips use an RSA encryption scheme with asymmetrical keys. This is explained in further detail in another section.

In order to provide better protection and privacy, TPM chips use several different keys to fulfill different function requirements. In this way, there is never any single “master key” that can compromise security and give a user full access. The following is a list of the different style of keys provided by a TPM chip.

- **Endorsement Key:**
The Endorsement key of a TPM chip is unique in its functionality. The key is placed on the chip during the process of being manufactured. The primary function of the endorsement key is ensure other TPM chips that a piece of information could only have been encrypted using a TPM chip and is trusted. The private portion of this key may never leave the chip or be replaced. If the key is lost due to hardware damage, it is impossible to be recovered.
- **Storage Root Key (SRK):**
The Storage Root Key (SRK) is a unique key that is associated with the owner of a TPM chip. This key is non-migratable and considered the parent of all other keys. In this way the SRK is guaranteed to always be present on the chip. As the parent, the SRK must be verified before using any child key in the TPM. It should be noted that due to the SRK being tied to the owner of the chip, if the owner changes the SRK will also be changed. Any keys encrypted with the old SRK will be permanently lost.
- **Storage Keys:**
Storage keys are used by a TPM chip to securely store any type of key (including other storage keys). These keys can be classified as migratable or non-migratable as needed by the chip.
- **Binding Keys:**
Binding keys are used by a TPM chip so securely store symmetrical keys used by a user. In this way, symmetrical keys are given asymmetrical protection and are never seen “in the clear” where they could be compromised.
- **Identity Keys:**
Identity keys are a special class of non-migratable keys that are used to ensure that a stored hash value or other non-migratable key is used for a specific TPM chip. These keys are always created as children of the SRK to further ensure the key exists within the TPM chip.

1.3 BitLocker

With the introduction of the “Microsoft Windows Vista” operating system, Microsoft developed and deployed the operating system native encryption program BitLocker. BitLocker is a versatile encryption program that provides a user with full disk encryption for secure access to files. However,

BitLocker is distributed only to a select subset of the “Vista” (and upcoming “Windows 7”) version types. Currently, BitLocker is only shipped with the Ultimate and Enterprise versions.

1.3.1 Technique used to encrypt

In order to encrypt data, BitLocker makes use of an algorithm called “AES-CBC + Elephant Diffuser”. This algorithm is based on the AES model of encryption that utilizes a “Cipher Block Chain” (CBC) method. The CBC is designed so that every encrypted block of memory is dependent on the block directly preceding it. As a result, the blocks are considered to be “chained” and must all be present in order to be correctly decrypted in full. However, it is possible to decrypt portions of the encrypted text by altering arbitrary bits and sacrificing previous encrypted blocks (Ferguson, 2006). In order to overcome this, Microsoft introduced the concept of a diffuser into the algorithm.

The diffuser in the algorithm is designed to scramble the plain text values of a block prior to being encrypted with the AES-CBC. The method was developed by Microsoft for the specific purpose of BitLocker. Though the technique is not as proven as the base AES-CBC, the addition of the diffuser can be shown to be at least as strong as AES-CBC (Ferguson, 2006).

As stated previously, BitLocker is a feature that comes pre-packaged as part of the Vista (Enterprise and Ultimate) operating systems. As such, the BitLocker process is, at its core, a software based encryption method. In order to operate, the BitLocker program requires a separate partition (at least 1.5 gigs) for the encrypting and decrypting process. Once this requirement has been met the program may then begin the process of encrypting the entire system volume(s).

1.3.2 Protection methods available

The BitLocker application provides users with a variety of different methods to secure their data. These methods include the use of a basic key, dongle support, TPM support, and combinations.

1.3.2.1 Keys

The single key method of protection requires a user to supply a passphrase for the system to use for authentication. This passphrase is required by the system during the boot process or when modifying the BitLocker system. This key may consist of 4 to 20 characters. In addition to the passphrase, BitLocker will provide the user with a 48 character “recovery key” that may be used to access the system. The recovery key is used when the system’s hardware or system files have changed. For example, the recovery key will need to be provided by the user if the hardware has changed or the system returns from the hibernation state. This recovery key is always generated and upon creation the user is prompted to save a back up of the key and/or print the key for storage.

1.3.2.2 Dongle

The second method of authentication is the use of a USB dongle. The dongle used for BitLocker can be any USB storage device and does not require any preparation. BitLocker will save a file to the drive that contains a set of encryption keys to be loaded before the operating system is permitted to boot.

1.3.2.3 TPM

Windows BitLocker was designed from the beginning to make use of the available functions of a TPM chip (1.2 compliant). In order to achieve this, Windows Vista was designed to natively interface with the chip in a high level way. This method allows BitLocker to verify the authenticity and stability of the system without user interaction. In this case, the authenticity of the machine can be comprised of both hardware configurations and system file status. If the TPM chip is unable to verify the system, the user may use the previously mentioned recovery key to re-authenticate the system. However, using this method of protection will permanently link the file system to the specific TPM chip. This will render simply moving the protected drive to another machine ineffective due to the new TPM chip not possessing the correct RSA key pair.

1.3.2.4 Combo(s)

The final encryption method available is the ability to use any of the previous methods in combination with each other. Combinations include any two methods or all three at the same time. This method provides a multilevel authentication process and the inherit security that comes as a result. When utilizing this method, it is most common to use the Key and TPM methods (when available) together. This is a result of the fact that the TPM provides a strong authentication while the Key provides light user verification.

1.4 Overview of current law enforcement practices

Although digital forensics is a relatively new field in law enforcement, several accepted methods are practiced in the field. Similar to other law enforcement practices, each case requires a different approach depending on the surrounding circumstances. The two major circumstances that may be considered are the warrantless and warranted investigations.

1.4.1 Warrantless Search (“Knock-and-talk”)

1.4.1.1 Explanation

A warrantless search, often referred to a “knock-and-talk”, is a method of investigation that requires a suspect’s consent to search. An officer must be allowed entry and access to a suspect’s system before any investigation may take place.

1.4.1.2 Requirements

This style of investigation has no set requirements for the officer to initiate. All that is required is the suspect’s cooperation and consent. It is typically used in situations where an officer is unable to obtain a warrant from a judge for a search but feels there is probable cause for investigation. However, if contraband is located during the investigation, an officer has the ability to immediately request a warrant to perform an official legal search.

1.4.1.3 Restrictions and allowances on the officer

When operating under the scope of a warrantless situation, an officer has many allowances that are not available with a strict warrant. One such allowance is that an officer may search anything the suspect gives consent for. This allows the officer to investigate computer systems, storage devices, and any other electronic equipment. It is possible to relate this style of search to that of a traffic stop. If an

officer believes the driver is hiding anything in the vehicle, he may request to look inside. If the driver refuses, then the officer cannot investigate further without probable cause.

Despite the limited freedom presented by this method, the officer is also under some restrictions. For any evidence to be used in court, the officer is required to document any actions or changes performed on system. Further, if the suspect refuses an inspection, the officer must then obtain a warrant before the search for evidence may proceed further. Failing to observe these restrictions may result in any evidence found to be thrown out or the entire case itself.

1.4.2 Warrant

1.4.2.1 Explanation

A warrant search is the typical search executed by law enforcement during investigations. A warrant must be obtained from a Magistrate, State Judge, or Federal Judge. The requirements to obtain the warrant differ between the different levels of judge. Typically a warrant from a Magistrate is easier to obtain but is not as strong in court as the others. Likewise, a federal warrant from a Federal Judge requires more information to obtain and is strongly upheld in court. In general, to obtain the warrant an officer must put forth a written request to be approved. Within the request is the reason for the search and the scope of the proposed warrant. If the request is rejected an officer is able to resubmit an amended request.

The obtained warrant will expressly state the power and scope of the authorized search to be performed. Officers must conform to these boundaries or risks having any evidence found become inadmissible in court. If the officer would like to expand the scope of the warrant (including items to be searched) then the warrant must be amended by a judge or a new warrant must be obtained.

1.4.2.2 Requirements

As stated, an officer may only be obtained through written request to a judge. The request will include the reasons for the request and the scope of the search. The scope of the search must specify explicitly the locations to be searched and any and all computer equipment to be searched/seized. Obtaining a warrant is typically only possible with strong existing evidence or probable cause.

1.5 Potential problems due to Encryption with evidence collection

As encryption becomes more accessible to users and easier to set up, it's use in forensics cases is bound to increase. Several cases have already gained notoriety in the press due to the presence of encryption and how the old laws are being interpreted in response. This introduction of encryption into computer crime presents new problems in the field of digital forensics. Current law enforcement procedures are ineffective and may actually cause a loss of evidence if used.

1.5.1 Loss of keys

The major issue with current techniques is the risk of encryption key loss that may occur during a search and seizure. If encryption is present on the machine and currently in use, the loss of these keys may result in being unable to recover any or all data from the system. One major way the keys may be lost is with the sudden loss of power due to law enforcement intervention. If a preview of the system

is not performed by a technician, standard practice is to immediately remove power to the system to try and prevent data loss. Under this practice, if any encryption keys are stored in RAM the data will be lost due to RAM's inability to retain data without power. Due to most encryption techniques revolve around software, it is highly probable that encryption keys will be present in RAM.

With the loss of the keys, an officer may not be able to access the data. In this situation the officer would need to rely on the suspect having a backup or print out of the key that can be located. If this is not the case, then a legal gray area emerges. Under current law interpretation, courts are divided on the issue of suspects being obligated to turn over electronic passwords. (CITE THE TWO CASES FOUND).

In the case of hardware encryption, such as TPM, the system may require re-authentication on startup or in order to access restricted disk space. If the suspect system is "cloned" as a forensic image, it will not be able to authenticate the machine even if the encryption keys are known. Even if the forensic image is placed in the suspect system it may result in the same situation of being unable to locate the keys for use.

1.5.2 Transparent operation

An often overlooked feature of current encryption methods are the ability to run in a transparent operation mode. In this style of operation it may not be overtly obvious that an encryption program is running in the background of the machine. It would also be impossible to tell if hardware encryption is currently in use without a detailed knowledge of all system components currently in use. These situations may be due to either the ability of the system to run silent or due to execution only as needed.

When an encryption program is running silent it is operating with minimal user interaction. Often times, a user is only required to provide information upon the applications start up process and then remains in the background as needed. This is further augmented by software being able to integrate itself with the host operating system. When integrated into the operating system it may become impossible to differentiate between the program and other system process. For example, BitLocker does not show up in either a list of running process or services when in use by the machine.

Execution based on events is a second method available that obscures evidence of encryption being present on the system. A common application of this process would be if the system only authenticates on events such as start up or returning from a hibernation state. With this scenario, software applications such as BitLocker and hardware like TPM will not appear to be running during normal system operation. Another common event is if a software application (such as TrueCrypt) is only run during the mounting or un-mounting of encrypted file space. When mounted, the encrypted file space may appear simply as connected drive and not as a software source.

1.5.3 Single Chip Marriage

As previously stated, it is possible to utilize encryption methods that are unique to a specific system. An example of this capability is the use of TPM encryption chips. One method provided by a TPM is the ability to authenticate based on hash values taken from the system itself. With this style of

operation, if any components differ from the “trusted” values the system will not boot correctly. Similarly, any files that are encrypted using the TPM chip cannot be moved from one system to another and be decrypted. This holds true even if the same application is used with the same user passphrase (if used). The RSA keys used between the application and the TPM chip will not match and therefore cannot be used for the decryption of the files(s).

2 Investigation Techniques

As computer forensics became a more prevalent field and tool for law enforcement investigations, standards for the seizing and searching of digital evidence came to exist. The techniques were created to ensure both consistency and legality of the processes used to gather evidence.

2.1 Conventional methods of finding digital evidence

Most often, law enforcement is advised to investigate captured machines in a forensics lab using “cloned” copies of the suspect drive.

2.1.1 Linux Boot

The Linux boot style of procedure is used as both an investigation technique and a cloning technique. For the purposes of forensics, the Linux system is typically a standalone “live boot” CD that does not require installation on a drive. This allows investigators the flexibility of searching despite the currently installed “host” operating system or lack thereof. Using the secondary operating such as Linux enables an investigator to gain access to sections of the system that are typically locked out by an operating system such as Windows. Also, live Linux operating systems do not (by default) alter system files on a hard drive during the boot process or during operation.

2.1.2 Cold Image capture

Often times, when a machine has been seized in an investigation, the system is not investigated on site by a technician. The investigation and analysis of the system is performed in a forensics lab using a “Cold Image” or clone. The cold image process is a technique used by investigators to preserve evidence. In this process, the systems hard drive(s) are loaded into a machine utilizing write blocking capabilities. The write blocker prevents the investigator or his system to write anything to the drive and compromising the evidence. The investigator would then use cloning software, such as Linux DD or AccessData’s FTK Imager, to duplicate the full contents of the hard drive to a new disk. It should be noted that the term “cold image” refers to the fact that the evidence drive is not booted to during this process.

2.2 Live methods of finding digital evidence

In recent years, the practice of live investigations has become more common. A trained computer forensics investigator on site may be able to locate time sensitive data and also capture volatile data from the system. The two main categories of these live methods are “live image capture” and “live analysis”.

2.2.1 Live image capture

Live image capture is the ability of an investigator to capture a forensics image of a running machine in the field. This method would be appropriate if a system was found running when the machine is seized. As a reminder, standard operating procedure is typically such that if the machine is off, do not turn the machine on but instead send it to a laboratory.

2.2.2 Live analysis

The practice of live analysis is a growing method used by investigators to obtain information from running systems. Live analysis can be used under both a warrant enabled search or a “knock-and-talk”. A “knock and talk” live analysis may allow an investigator to discover questionable material that may then be used to apply for a warrant.

3 Statement of purpose

As encryption becomes more accessible to the common user, it will change how law enforcement interacts with suspect machines. Current computer forensic procedures will prove ineffective against hardware and software based encryption methods. As a result new procedures need to be developed and used live and on site in order to gather necessary data. This paper will explore the current options for users to encrypt data, the law enforcement practices currently in use, and the development of the LEAP utility to assist law enforcement in live forensics.

4 Literature Review

Live computer analysis is currently still an emerging field in computer forensics. As a result, the publication of papers covering live techniques is not very common. In addition, the use of techniques to overcome encryption for law enforcement is not widely publicized. Therefore articles that directly relate to this paper are few.

The paper “Recovery of Encryption Keys from Memory Using a Linear Scan” (Hargreaves & Chivers, 2008) discusses the ability to obtain encryption keys, specifically TrueCrypt, from an image of live RAM memory. The author(s) were able to successfully recover the encryption keys used by the program for protecting files. However, the process used is designed to operate as a proof of concept only. The process required that a TrueCrypt container file be currently running in memory and that the program is operating in the standard configuration. Given these results, the author(s) conclude that the collection of volatile memory should be included in forensic procedures in order to preserve potential evidence.

The article “Lest We Remember: Cold Boot Attacks on Encryption Keys” (Halderman, et al., 2008) explored the ability to recover stored encryption keys from a running machines RAM. Unlike the previous paper, the extraction technique used is performed on site using a secondary machine. The article details the process of reducing a system’s RAM to below freezing temperatures in order to preserve transient data. The cold RAM is then analyzed using a network boot to extract possible

encryption keys. Though the process is successful in the article the practice is not practical for real world use. The process used is extremely technical, complicated and damaging to the suspect machine.

Finally the article “First commercial tool to crack BitLocker arrives” (Protalinsk, 2009) provides a brief overview of a piece of commercial software to overcome BitLocker. The software detailed uses a RAM image obtained from a running system for analysis. The image is parsed for possible keys that may be used by BitLocker to encrypt the disk. Due to the program requiring a live image of RAM, it indirectly supports the use of live forensics in investigations. In addition, this program may be used to obtain keys when the system is running in a non-administrative environment. However, more details are needed before a conclusive result can be obtained.

5 Overview of LEAP

5.1 Introduction

5.1.1 Need

Current accepted computer forensics procedures dictate that a suspect machine should be powered off and/or restarted into a forensics safe environment. During this process, transient evidence will be lost as well as any current connections into encrypted file space. In order to preserve this evidence, it is imperative to use a “live forensics” environment and gather any possible data. However, current tools available to law enforcement are unequipped to deal with new advancements in file encryption and are unintuitive.

5.2 Objective

The purpose of this LEAP tool is to provide law enforcement with a new tool to collect as much data as possible while operating under a “live environment”. The tool will be loaded onto a currently running machine for analysis. While running, the tool will conduct some automated information gathering before allowing the user to interact with the environment. While interacting with the suspect machine (via the tool) the officer will be provided with constant feedback to ensure correct operation. The user will have the ability to run automated collection processes as well as execute custom searches. During the custom searches, the user will have the ability to export any possible evidence to an external drive for posterity. When the user has finished using the program, a detailed report and log of actions will be created.

5.2.1 Abbreviations/Definitions

- **TPM – Trusted Platform Module**

On motherboard chip used to encrypt/decrypt files. Capable of identifying a machine based on hardware components and restrict access based on hardware settings and security certificates.

- **OS – Operating System**

System software that allows a user to interact with computer components and to manage running programs.

- **Transient Data**

Data that is no longer accessible once the computer is powered off or that is easily changed by the operating system.

- **User**

For the purpose of this document, the user will be considered to be a law enforcement officer unless otherwise stated.

- **User Interface**

Visual output for a program that allows a user to interact and execute the program.

5.3 Requirements

In order to provide a product that meets the requirements for law enforcement use, those requirements must first be established. The requirements were designed after deliberation with the West Virginia Cyber Crime division and the National White Collar Crime Center. This section will outline the major requirements to be fulfilled by the product.

5.3.1 Implementation Requirements

Table 1 illustrates the implementation requirements for this product. These requirements are derived from project proposal submitted for the National Institute of Justice by Dr. Roy Nutter.

Implementation Requirement No.	Implementation Requirement Description
1	Provide a free utility to law enforcement
2	Be up to date with current encryption programs
3	Collect transient data in a organized manner
4	Follow accepted live forensics practices
5	Should operate on a major portion of computers
6	Useable by non-technical users

Table 1 Implementation Requirements

5.3.2 Engineering Requirements

For each entry in Table 1, one or more “Engineering Requirements” were created to ensure each requirement was met. These requirements reflect the technical aspects of the product and can be found in Table 2.

Engineering Requirement No.	Engineering Requirement Description
1	Operate under the Windows XP OS
2	Operate under the 2.0 .NET Framework
3	Operate under the Windows Vista OS
4	Maintain an active log of all actions taken
5	Provide feedback to the user regarding found files
6	Locate files currently accessible in encrypted file space
7	Retrieve information regarding possible encryption
8	Retrieve any possible encryption keys
9	Use no software that requires paid licensing

Table 2 Engineering Requirements

5.3.3 Requirements Specifications

For each of these engineering requirements in Table 2, a direct correlation can be shown to the implementation requirements of Table 1. These correlations are shown in Table 5. Table 5 shows exactly which implementation requirement(s) is met through each engineering requirement. Each correlation has been supported by a justification.

Engineering Requirement No.	Implementation Requirement No(s).	Justification
1	5	Currently the Windows OS contains a majority market share for personal computer use. Within this market share, Windows XP has been widely accepted for many years.
2	5	Since 2006, the Windows 2.0 .Net Framework has become standard for use with Windows XP.
3	2, 5	The Windows Vista OS introduced easy to set up encryption for users to activate. Under Vista, an entire hard drive may be encrypted and/or linked to hardware components.
4	4	In practice, any actions preformed on a suspect machine must be logged and documented. This log allows a full record of any and all actions taken by the user or program.
5	4, 6	Under typical circumstances, the product will be used by individuals who have had exposure to computers but may not possess detailed knowledge. Therefore, the program should be intuitive to use and still maintain good forensic practices.
6	2, 3	Any files that are located in an encrypted file space (or memory) may become inaccessible if the machine is logged off or powered off at any time. The data should be able to be both located and copied to external sources for preservation.
7	2, 3	A user should be notified if any potential files are found to contain encryption. This would allow a user to take the appropriate steps to preserve the evidence.
8	2, 3	If possible, encryption keys should be recovered so that the data in the encrypted file space can be recovered at a later time. This would allow for a more thorough analysis under controlled conditions.
9	1	Use of licensed (paid) software would require users to pay to use the product.

Table 3 Requirements Specifications

5.3.4 Competitive Benchmarks

For this comparison, the popular forensics software “Forensics Toolkit” (FTK) will be used. This software is widely used by many law enforcement groups in the United States. The data below was provided by AccessData’s website. (AccessData)

Benchmark	Comparison Results
Cost	At this time, AccessData has removed the cost of the program from their site. LEAP is a completely free program.
Ease of Use	FTK requires extensive training to effectively use the program. LEAP provides simple point and click interface.
Support of Encryption	FTK supports the detection of 6 encryption programs and styles. LEAP is designed to locate 15+ installed applications that may point to the use of encryption.
Live Analysis	FTK is designed to only provide a network based live analysis. This requires extensive knowledge and training, and additional hardware. LEAP is designed to run from a single USB enabled device on the suspect machine.

Table 4 Competitive Benchmarks

5.4 Constraints

In order to achieve the goals of the research project and law enforcement, the LEAP program must abide by certain constraints. These constraints encompass economic, ethical and social aspects of use and functionality.

5.4.1 Economic

The LEAP program was to be free of cost. This was the result of the overall goals of the research project that produced the program. In order to remain free, the program does not make use of any licensed software. All software created and used for LEAP are publicly available to all users. In addition, the LEAP software does not violate any license agreements presented by the software in question.

5.4.2 Ethics

In addition to economic concerns, LEAP must also conform to three primary ethical restrictions. The first main concern is that through the course of use the program must not violate any laws regarding privacy and searching. In addition, the program must keep a detailed log of any and all actions taken by the user and program. Finally, the program must take precautions to make sure that no data is written to the suspect drive by the program.

5.4.3 Social

The program needed to meet three main social goals in order to be successful. First, the LEAP program needed to be easily used by non-technical users. This is due to the target audience not necessarily having a technical background or extensive experience. Secondly, the program needed to be flexible in its use. It needs to allow users to use the functions provided by the system as needed on a case by case circumstance. Finally, the program must provide intelligent feedback to the user during operation. This ensures that the user knows what is being done at all times by the program and confirms the program is operating as intended.

6 LEAP 2.0 Detailed Design

This section will cover the technical aspects of the LEAP program version 2.0. It will cover both user interaction and the actions taken by the program. Due to being a technical method overview no source code will be used in this section. A breakdown of the programs functions and specifications can

be found in the Requirements Specification section of the Appendix. It should be noted that the LEAP program is built using the .NET 2.0 libraries. As such, at a minimum these libraries must be installed and running on the suspect machine.

6.1 Execution of Application

The LEAP program is designed to be implemented on mobile storage technology USB storage devices such as external flash drives and hard drives. As a result, the program is packaged with a simple Windows “autorun.inf” file. This file is designed for two main applications. The primary application of the file is to initiate the Windows auto run functionality to immediately start the program on USB insertion. This allows the user to begin use of the program without the need to locate the executable on the mobile drive. The secondary use is a backup function in case the auto run capability is not enabled on the suspect machine. The file instructs the operating system to both name the drive “LEAP” and use the LEAP icon for identifying the drive. This makes the drive easily located if the user must manually execute the LEAP executable.

Figure 2: LEAP Device in My Computer

6.2 Start Up Operations

Upon execution, the LEAP program will begin to guide the user through the applications start up process. The start up process (detailed in the following subsections) is used as the primary source of

system information and is designed to be almost completely autonomous. The user only needs to provide basic instructions for it to execute.

It should be noted that due to some possible operating system restrictions, LEAP is designed to be run as an administrative user. Because LEAP is not being certified by Microsoft, this is achieved by embedding an application manifest into the executable at compile time. The manifest provides information to the operating system needed to run in “administrative mode”. As a result of the “User Account Control” (UAC) system introduced in Windows Vista, the user may be prompted to allow the program. This notification is dependent on the user’s UAC settings and permission level. An example of this prompt is illustrated in Figure 3.

Figure 3: Administrative Request

6.2.1 Encryption Check

The first step in the start up process is for the program to attempt to check for known installed encryption programs. The program will prompt the user upon beginning to this first step. It should be noted that the encryption check also includes a check for the activation of Windows BitLocker on the system. For simplicity, each check is broken into individual processes with unique response areas.

Figure 4: Encryption Check Request

The first check run by the program is for BitLocker encryption. This check uses the “cscript.exe” executable found included with Windows. The executable is run outside of the Windows shell to ensure the integrity of the data returned. LEAP passes a Windows command script to the program that instructs the operating system to return the current state of the BitLocker program. The script is “c:\Windows\System32\manage-bde.wsf -status”. The output of this script is parsed to obtain the current state of BitLocker as known by the operating system. If BitLocker is found on the system, the program will again use the “cscript.exe” executable to obtain the stored keys for the system. The script passed to return the keys is “c:\Windows\System32\manage-bde.wsf -protectors -get ”. This will return the stored keys and protection methods for all drives currently using BitLocker. The user will be notified of the results of search by a color coded box located next to “BitLocker Detection”. The possible states for this box are:

- Green – BitLocker was not found on the system.
- Yellow – BitLocker cannot be conclusively determined as running or not.
- Red – BitLocker has been detected on the system.

Any keys located by the program will be reported to the user in the form of both a pop up message box and saved file on the external drive. Both the file and message box will contain all known drives using BitLocker, the current protection methods in use, and the keys stored by the system to access the drives.

The second part of the encryption detection process is the search for installed possible and known encryption programs. This search is a two stage process to separate the known programs from the unknowns. The program uses a variant of the file search method (covered further in section 6.3.3.1) to search for keywords in the systems program files. The keywords used for this process are included in the plain text file with the program so that users may edit the list as needed without needing to re-compile the program. This file is generally made up of generic keywords, such as “crypt”, to cover a broad spectrum of possible files and can be found in the top directory of the program.

Upon locating all possible files, the system will begin the second stage of the location process. This part utilizes a simple string filtering method to parse out specific programs and files. These keywords, like the generic keywords, are included in a plain text file with the program. The system will then display all found results to the user in the two provided text boxes (see Figure 5). All possible files will be displayed in the lower text box and flagged as evidence of “Possible Encryption”. Any known files will be displayed in the top box in red and flagged as evidence of “Known Encryption”. Also, like the BitLocker results, a color coded box is provided to the user for quick reference. The box states are:

- Green – No files match the list of possible encryption keywords.
- Yellow – Files were found to match the keywords for possible encryption.
- Red – Files were found the matched the keywords of known encryption programs.

Figure 5: Encryption Results

6.2.2 Information Retrieval

After the system performs the encryption check, the next step is the retrieval of general system information. This process uses the Win32 API interface built into the .Net framework to retrieve information such as hard drive partitions, system manufacturer, system model, and attached storage devices. The purpose for this information is to further identify a suspect’s machine in the reporting process. This ensures that any actions performed and recorded can be traced back to the system in question by physical components.

6.2.3 Report

The reporting functionality of the LEAP program allows an officer to maintain an accurate record of all actions performed by the program at anytime. Any action taken by the program either automatically or at the officer’s prompting is recorded with the date and time the action occurred. In addition, the officer is prompted to provide relative “housekeeping” case information such as suspect

information and the current time as relative to the officer. This information provides the officer with a quick case reference after the processing is finished. It should be noted that when the officer is first prompted for this data, that the “Case ID” chosen is used by the program to store any saved or exported files are a result of the investigation. This allows an officer to use the same external drive for multiple machines on scene while maintaining data separation from multiple machines.

Information stored by LEAP for reporting is kept in two locations at all times. The primary location is a temporary database that exists within the system’s RAM. This database structure allows the program to quickly store, categorize, and retrieve any recorded information in real time. The advantage to this method is that access times are faster because LEAP does not need to read or write to any physical storage device. The secondary location that information is stored in is a temporary “Log.CSV” file on LEAP device. The information stored in this file consists wholly of actions taken by the LEAP program. The file is designed to exist as a backup file for the officer in case the program is forced to un-expectedly shut down before a formal report is generated.

The screenshot shows a Windows-style dialog box titled "Report Information". It contains the following fields and controls:

- Officer Information:**
 - First Name: [Text Input]
 - Last Name: [Text Input]
 - Officer ID: [Text Input]
 - Actual Date: [MM/DD/YYYY format]
 - Actual Time: [HH:MM AM/PM format]
- Suspect Information:**
 - First Name: [Text Input]
 - Last Name: [Text Input]
 - System Date: [10/25/2009]
 - System Time: [10:47 PM]
- Case ID:** [Text Input]
- Notes:** [Large Text Area]
- Buttons:** "Preview" and "Continue" buttons at the bottom right.

Figure 6: Report Information Form

6.2.4 System Analysis

The final action performed by LEAP during the pre-processing stage is the optional performance of a “System Analysis”. This analysis runs selected free third-party command line applications. These applications are used to collect extensive system information from the system for future analysis. The programs provided are enabled or disabled depending on the operating systems version. This is due to a

number of the applications being unable to operate correctly under the newer Vista architecture scheme.

Figure 7: Program Selection

Figure 8: System Analysis Main Screen

6.3 Component Breakdown

This section will discuss the main components accessible by the user from the main LEAP interface. This interface, referred to as the "LEAP Navigator", is the primary gateway used to switch among the main functions of the program.

Figure 9: LEAP Navigator

6.3.1 Graphic Viewer

The “Graphics Viewer” is a utility provided to the user to be able to locate, browse and export image files on a suspect machine. The program is capable of either searching all connected drives (except the LEAP device) or a specific directory recursively. All located files are parsed for known image files extensions (“jpg”, “png”, “gif”, “bmp”, “tiff”, and “jpeg”) before being loaded into the actual user interface.

Figure 10: Main Graphics Viewer Interface

Once loaded, image files are displayed to the user in groups of 25 thumbnail images. This prevents the program from attempting to load an excessive amount of potentially large image files at one time. Also, each thumbnail displayed is generated by the system in real time. This allows the system to not rely on cached system thumbnails that may not be up to date. When browsing, the user is notified of the total number of pages (sets of 25 images) located and the current page the user is on. This information is located in the center of the bottom of the interface.

The graphics viewer also provides two other primary functions to the user. The first function is the ability to see a detailed view of a selected image. If a thumbnail is clicked, the pictured details form is opened using the given image. This form provides an expanded view of the selected image, two file hashes of the image, and the ability to directly open the directory of the image. The expanded view forces the image to open to the dimensions of 455x509 pixels. This allows for a large view of both smaller files (below 455x509 pixels) and those larger. The form also provides the user with both the MD5 and SHA1 hash of the selected file.

Figure 11: Picture Details

The final usage of the graphics viewer is to allow a user to export a given picture or pictures to the LEAP device. This user is able to use the checkboxes provided under each thumbnail to flag the image for exporting. In addition, the user has the ability to export all located image files through the use of the “Export” menu located at the top of the form. Each exported file has its SHA1 hash computed and saved to the LEAP device as well as the original location of the file and the exported file name. The exported file name is recorded due to the possibility that the program has had to rename the file in order to properly save all files. This would occur if two files that were exported contained the exact same name. Under this circumstance, each subsequent file with the same name will be exported as “marker.filename”. The marker for each file is an index value of 0,1,2, etc used to retain the distinguish each file while retaining the actual file name.

6.3.2 Parse Flash Cookies

The “Parse Flash Cookies” component is designed to retrieve a list of websites that have been visited on the suspect machine. This is possible because the heavy integration of flash in current websites and because temporary the data from these sites is not cleared by most web browsers. Within the application data section of the operating system, Macromedia Flash keeps a separate folder for each unique website visited that contains flash. As a result, LEAP is able to parse the folder names that exist within “\Macromedia\Flash Player\#SharedObjects” in the “Application Data” folder to obtain the “flash cookies”.

6.3.3 File Explorer

The “File Explorer” is a means for the officer to explore the suspects system within a controlled environment. The form is designed to display a directory tree of all located drives/folders and a summary of all contained files of a selected folder. All folder and file locations are done in real time each time the user selects a folder “node” from the directory tree. The selected folder is used as the base for both a .Net directory exploration (Directory.GetDirectory) and .Net file exploration (Directory.GetFiles). Each file that is returned is checked for Windows recognized file encryption based on its properties.

Files and folders located by the “File Explorer” are able to be exported to the LEAP device with a single click within a context menu. For file(s), each file is exported to the LEAP device’s export folder after being hashed and noted by the program. This information is stored within a CSV file and contains both the SHA1 hash and the original location of the file on the system. For directories, the user has the ability to export all files and directories located recursively below the selected file. This export process maintains the directory structure for all files below the selected folder. Any file(s) exported using these methods also undergo the same hashing process.

In addition, selected files and folders are designed to be exported to LEAP’s “Graphics Viewer” for analysis of images. Details of this process can be found in section 6.3.1.

6.3.3.1 File Search

The “File Search” functionality of LEAP is designed be used as both a standalone support class and as a support form for the “File Explorer”. It is designed to provide the user with several methods to

search for files on a suspect machine. This section will cover the support form functionality due to the support class performing the same functions in the background without a user interface.

Figure 12: Primary File Search Interface

6.3.3.1.1 File Keyword Search

This function, file search, provides the ability to locate files and folders by use of keywords. The program is designed to locate a file based on combination of keywords in the file name that are provided by the user. Multiple keywords are supported at one time and are neither mutually exclusive nor mutually inclusive. Furthermore, a user is not required to provide a full word when searching. For example, the search phrase "th one" will return files and directories that contain "the", "then", "that", "someone", "something", etc. This allows a user to locate files using broad keywords with little restriction. This keyword search can be applied to either all connected drives (excluding the LEAP device) or to any top level folder and its containing folders. It should be noted that it is possible for the search to not access certain folders due to operating system restrictions. An example of this is the "System Volume Information" folder that is created when a Windows operating system is installed. Such files are not accessible.

6.3.3.1.2 File Extension Search

Aside from keyword searching, the user has the ability to locate files based on certain predefined file extensions. These filters may be used in any combination to suit the user's need. In the event that no specific file type(s) are selected, the program will default to searching for all file extensions. The predefined file types are as follows:

- Image Files
 - Jpg
 - Bmp
 - Png
 - Tiff
 - Jpeg
- Video Files
 - Wmv
 - Mov
 - Flv
 - Wma
 - Avi
 - Mpg
- Common Files
 - Txt
 - Doc
 - Docx
 - Ppt
 - Pptx
 - Xls
 - Xlsx
 - Pdf

6.3.3.1.3 File Hash Searching

An additional feature of the file search function is the ability to locate files based on a unique file hash. The file hash provided by the user may either be a MD5 or SHA1 style hash. This allows a user to locate a specific file that is suspected to be located on the suspect machine. For example, a file known have been downloaded from a peer to peer network may be located despite it being moved and renamed. In addition, the program allows the user to provide a CSV file that contains several file hashes of intermixed hash types. This file is comprised simply of the first column containing the actual hash and the second column containing the hash type. Once loaded, the program will verify that all provided entries are in the correct form and correct type. For example, if a hash is missing a value the user will be notified of the offending hash and will not attempt to load that entry. All other entries that are correct will still be loaded into the program.

Figure 13: Hash Entry

It should be noted that the use of large scale hash searching may take a very long time. As the number of files located in the search area increases, the time to search will drastically increase. This is due to the programs need to create an open file stream of every located file before a hash may be computed. As a result, it is not recommend that user performs a hash search over the entire system. Also, the user is able to use the file type filters in conjunction with the hash searching to reduce the number of files that need to be processed.

6.3.4 View/Modify Report

The “View/Modify Report” function allows the user to freely alter selected information for the end report. All information may be edited other than system date/time and the case ID. This is allows the user to fix mistakes in the report and to add additional information to the “Notes” section of the report. In addition, the user has the ability to view the current version of the report at anytime from this screen. Any changes made to the information will be immediately reflected in the report.

7 Conclusions

As hardware and software encryption become common, current law enforcement practices are obsolete. In order to handle encryption, law enforcement must use live forensic techniques. This MUST become part of standard operating procedure. In order to assist LE, I the LEAP utility was developed to provide both encryption detection and live previewing capability.

8 Acknowledgements

Funding for this project was made possible by Award No 2007-DN-BX-K126 awarded by the National Institute of Justice (NIJ), Office of Justice Programs, U.S. Department of Justice. Findings and information covered in this paper to not necessarily reflect the options of the Department of Justice. All findings and developed material are available through the West Virginia Cyber Crime Cooperative.

9 Appendix

9.1 Engineering/Law Enforcement Need Correlation chart

Engineering Requirement No.	Law Enforcement Need No(s).	Justification
1	5	Currently the Windows OS contains a majority market share for personal computer use. Within this market share, Windows XP has been widely accepted for many years.
2	5	Since 2006, the Windows 2.0 .Net Framework has become standard for use with Windows XP.
3	2, 5	The Windows Vista OS introduced easy to set up encryption for users to activate. Under Vista, an entire hard drive may be encrypted and/or linked to hardware components.
4	4	In practice, any actions preformed on a suspect machine must be logged and documented. This log allows a full record of any and all actions taken by the user or program.
5	4, 6	Under typical circumstances, the product will be used by individuals who have had exposure to computers but may not possess detailed knowledge. Therefore, the program should be intuitive to use and still maintain good forensic practices.
6	2, 3	Any files that are located in an encrypted file space (or memory) may become in accessible if the machine is logged off or powered off at any time. The data should be able to be both located and copied to external sources for preservation.
7	2, 3	A user should be notified if any potential files are found to contain encryption. This would allow a user to take the appropriate steps to preserve the evidence.
8	2, 3	If possible, encryption keys should be recovered so that the data in the encrypted file space can be recovered at a later time. This would allow for a more thorough analysis under controlled conditions.
9	1	Use of licensed (paid) software would require users to pay to use the product.

Table 5 Engineering/Law Enforcement Need Chart

9.2 Requirements Specification

Requirements Specifications may be obtained through permission of the West Virginia Cyber Crime Cooperative. (West Virginia Cyber Crime Cooperative, 2009)

10 Works Cited

AccessData. (n.d.). *AccessData Forensic Toolkit*. Retrieved from AccessData:
<http://www.accessdata.com/forensictoolkit.html>

Ferguson, N. (2006, August). AES-CBC+Elephant Diffuser: A Disk Encryption Algorithm for Windows Vista. Microsoft.

Halderman, J. A., Schoen, S. D., Heninger, N., Clarkson, W., Paul, W., Calandrino, J. A., et al. (2008). Lest We Remember: Cold Boot Attacks on Encryption Keys. *Proc. 2008 USENIX Security Symposium*.

Hargreaves, C., & Chivers, H. (2008). Recovery of Encryption Keys from Memory Using a Linear Scan. *The Third International Conference on Availability, Reliability and Security* (pp. 1369-1376). IEEE.

KetuWare. (n.d.). *Symmetric vs. Asymmetric Encryption*. Retrieved from
<http://www.ketufile.com/>

National Institute of Standards and Technology. (2001, 11 26). Announcing the ADVANCED ENCRYPTION STANDARD (AES). Federal Information Processing Standards Publication 197.

Protalinsk, E. (2009, December 4). *First commercial tool to crack BitLocker arrives*. Retrieved from Ars Technica: <http://arstechnica.com/microsoft/news/2009/12/first-commercial-tool-cracks-bitlocker.ars>

RSA Laboratories. (2002, June 14). PKCS #1 v2.1: RSA Cryptography Standard. RSA Laboratories.

RSA Laboratories. (n.d.). *RSA Laboratories - RSA Algorithm*. Retrieved from <http://www.rsa.com>

West Virginia Cyber Crime Cooperative. (2009). *Requirements Specifications for LEAP*. West Virginia Cyber Crime Cooperative.