To appear in *Proceedings of the Society for Optical Engineering Conference*. Orlando, FL.

Digital evidence obfuscation: recovery techniques

J. Philip Craiger*[a], Jeff Swauger[b], Chris Marberry[b]
[a]National Center for Forensic Science & Department of Engineering Technology, University of Central Florida, Orlando, FL 32816
[b]National Center for Forensic Science, University of Central Florida, Orlando, FL 32816

ABSTRACT

Criminals who use computers to commit crimes often hide the fruits of the commission of those crimes. Hiding files on a computer can take on many forms, from file names and extensions to more technical methods such as encryption and steganography. Encryption and steganography have the potential to severely impede the recovery of digital evidence. We discuss encryption and steganography below and describe potential methods of coping with each. The techniques we discuss require no special knowledge or advanced hardware or software; however, the use of these techniques does not guarantee the recovery of obfuscated information.

Keywords: Digital forensics, computer forensics, encryption, steganography

1. Forensic Analysis of Encrypted Data

In simple terms, encryption takes a digital artifact (text, picture, audio, video, etc.) and transforms it so that it is unreadable. The transformation process requires a cipher – a mathematical encryption algorithm -- and a key, typically a password or passphrase. The cipher takes the file and key and performs the necessary calculations to make the file unreadable. To decrypt requires reversing the transformation process so that the file is once again readable. This requires knowing which cipher was used, and most importantly, the key that was used in the transformation process.

Encryption is used legitimately and legally by business, industry, governments, military, and individuals, to ensure privacy of information by keeping it hidden from those not authorized to read the information. Unfortunately, it can also be used by criminals to hide the fruits of their crimes.

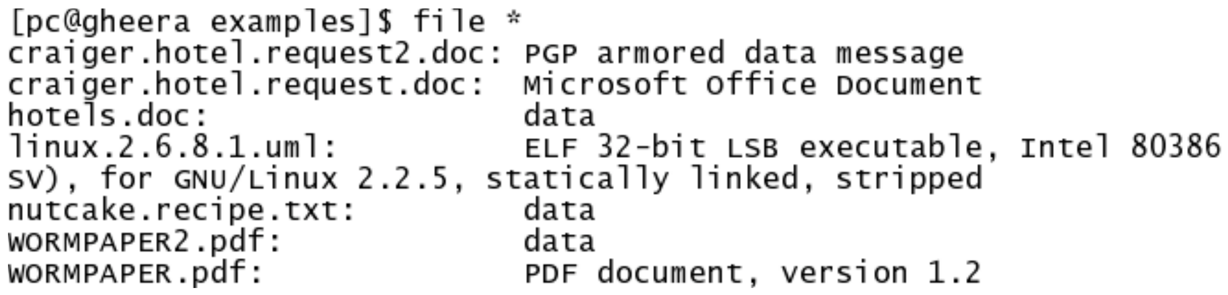
Suspects can encrypt any digital file, from a single individual file to an entire hard drive. Regardless of what data is encrypted, it is necessary to know both the cipher used for encryption and the password in order to decrypt the ciphertext.

1.1 Individual file encryption

Encrypted files must be identified first before processing can occur. Several digital forensic tools have the ability to determine whether a file has been encrypted by using the file's header information. Header information is digital information contained within the beginning of a file that indicates the file type: This method only works if the file headers have not been modified, and whether the file has a recognizable header.

There is no single, simple, 100 percent accurate way of determining whether an individual file is encrypted, or whether the file merely resembles an encrypted file. It is difficult (or impossible) to determine whether a file is encrypted by simply eyeballing its contents. In the following demonstration we have several encrypted files and one compressed -- using zip compression -- whose header information was removed. When the header information is removed from a file, applications can no longer correctly determine the file type. It also means that the file is corrupted and most likely cannot be opened by the application that created it.

We ran the UNIX file command against the files in this directory. The UNIX file command reads a file's header information, the first few bytes of a file, and compares it against a known list of headers to determine the type of file. As Figure 1 demonstrates, the file command contains enough information to conclude that request2.doc is as an encrypted file of type PGP armored. For the remaining encrypted files, however, the headers do not contain enough information to determine the type of file by using the file command.



```
[pc@gheera examples]$ file *
craiger.hotel.request2.doc: PGP armored data message
craiger.hotel.request.doc:  Microsoft Office Document
hotels.doc:                 data
linux.2.6.8.1.uml:          ELF 32-bit LSB executable, Intel 80386
SV), for GNU/Linux 2.2.5, statically linked, stripped
nutcake.recipe.txt:         data
WORMPAPER2.pdf:             data
WORMPAPER.pdf:              PDF document, version 1.2
```

Figure 1. UNIX file command run against several files

This suggests a significant problem for law enforcement: Which of the thousands of files on a computer are encrypted, and which are not?

1.2 How to determine if files are encrypted

How does one determine whether a file is encrypted? One method used by a number of commercial forensic applications is to calculate the entropy of the contents of the file. In this context, entropy is a measure of the randomness of the contents of a file. There is a significant amount of redundancy in human language, and a significant lack of redundancy may be an indication that a file is either encrypted, or compressed. However, we have found this is method is less than perfect in practice in identifying obfuscated files, particularly for files on which compression has been used.

An alternative is to run a forensic application that uses header information against all files on a hard drive first to determine if any files are obviously encrypted. As demonstrated previously, accurately determining the type of file can be hit-or-miss depending upon what type of encryption is used. If an investigator has on good authority that the suspect is known to use encryption, then some trial-and-error involving attempts to decrypt the files with a variety of encryption applications may be necessary.

1.3 File System-level Encryption

There exist several applications that can encrypt a volume (partition) or even an entire hard drive. These applications are available for several operating system including Windows, Linux, and Macintosh. These applications work the same as individual file encryption, using a cipher and a key to encrypt data.

Some of these applications can leave visible clues that a suspect has employed hard drive encryption. For instance, encryption applications display their icon in the Window's taskbar. (Note: There is no guarantee that a volume or disk encryption application will place its icon in the Window's taskbar.) Lack of an icon, however, doesn't guarantee that the file system is not encrypted. An investigator can be certain a file system or disk is encrypted if the investigator attempts to access files and is presented with dialog box requesting a password for file access.

When the user attempts to access a file on an unmounted encrypted file system, the encryption program will ask the user for the password. If the user does not have the correct password, the encrypted file system remains unmounted and the files cannot be accessed. However, if the encrypted file system is mounted, then the files can be accessed. (Mounting a file system makes the files on the media accessible to the operating system and applications.)

1.4 Windows Encrypted File System (EFS)

The Windows 2000, XP, and 2003 Server operating systems can encrypt volumes using Microsoft's Encrypted File System (EFS). The Encrypted File Systems is a hybrid cipher that uses a symmetric algorithm for data encryption and a public-key for protecting the symmetric key, where access to the public key is protected via a password. Windows 2000 defaults to 56-bit Digital Encryption Standard (DES), whereas Windows XP and 2003 Server default to the Advanced Encryption Standard (AES) with a 256-bit key, a much more secure cipher than either 56-bit DES or 3DES.

The Encrypted File Systems supports third-party data recovery through the concept of a Data Recovery Agent (DRA). A DRA is a designated authority -- most often a security or network administrator, but could be anyone -- whose private key is used to protect the data in addition to the primary's public key, i.e., the original data owner's public key.

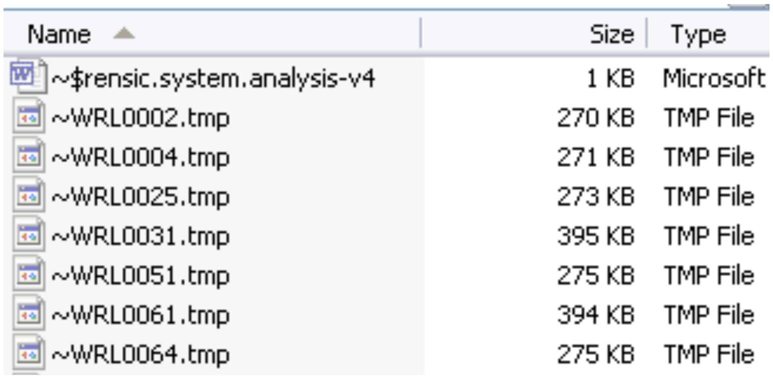
Consequently, in a networked environment data recovery of an EFS encrypted partition is very possible as long as someone has been designated as a third-party DRA. If a DRA does not exist, then the only person capable of decrypting the data is the primary owner. Should this be the case, other methods are required to gain access to the public-key's password. Some of these methods are described in a later section.

1.5 Breaking Encryption

Several techniques that can be used to recover digital evidence are described below. Each of these methods relies on recovering the password used for encryption.

1.5.1 First Step: Search for Data Remnants on Disk

If a suspect isn't careful he may inadvertently leave a copy, or remnants of a copy, of an unencrypted version of a file on disk after the file has been encrypted. This occurs -- without the suspect's knowledge -- because many programs store a temporary unencrypted version of a file on disk while the user is accessing the file. When the user closes the application that opened the file, the application deletes the temporary file. As with any deleted file, there is a good possibility that it can be recovered through various means such as searching unallocated and slack space on the suspect's hard drive (Craiger, 2005; Craiger, Pollitt & Swauger, 2005). Most applications transparently create temporary files so most users do not know that this process occurs. The figure below illustrates this phenomenon occurring while we were writing this paper.



Name	Size	Type
~$rensic.system.analysis-v4	1 KB	Microsoft
~WRL0002.tmp	270 KB	TMP File
~WRL0004.tmp	271 KB	TMP File
~WRL0025.tmp	273 KB	TMP File
~WRL0031.tmp	395 KB	TMP File
~WRL0051.tmp	275 KB	TMP File
~WRL0061.tmp	394 KB	TMP File
~WRL0064.tmp	275 KB	TMP File

Figure 2. Microsoft Word Temporary Files

Consequently, the first step to take with the recovery of encrypted would be to look for an unencrypted version located elsewhere on the file system. This can sidestep the exhaustive time consuming approach of the technique described below.

1.5.2 Social Engineering

The simplest method of overcoming encryption is to ask the suspect for the password(s). This technique can be very effective, particularly if it is used immediately after law enforcement serves a search warrant, a time when suspects may be psychologically weak, confused, or frightened. We know of numerous examples of suspects who openly shared passwords, and other relevant information, upon being served with a search warrant in the middle of the night.

Another approach relies on "social engineering." Social engineering is a term commonly used to describe how computer criminals gather information by tricking people (e.g., secretaries, network administrators, help-desk personnel, regular users, etc.), into divulging information necessary to break into a computer. In social engineering, the investigator attempts to use knowledge of the suspect to obtain the password, either by direct inquiry or by educated guessing based on information provided by the suspect. For example, suspects can often be induced to reveal private information about them that can be used to guess passwords. For example, often words such as pet's names, children's names, football team names, etc. are used as passwords. The effectiveness of social engineering is highly dependent on the interpersonal skills and insight of the investigator. Examination of the suspect's home and other details about their life can also be useful in searching for information relevant to the suspect's passwords.

If a suspect refuses to divulge his or her passwords, three other options are available. One is to use "password cracking" software. A second method is to search the suspect's computer, in the hope that the suspect's password is located somewhere on the computer (Craiger, 2005; Craiger, Pollitt, & Swauger, 2005). Finally, investigators may attempt to recover passwords from other accounts used by the suspect, in the hope that the suspect has used the same password on more than one account. Each of these methods is described below. Methods 1 and 3 are described below.

1.6 Automated Tools: Password Crackers

Password cracking is a term that implies an automated method of guessing passwords. There are three commonly employed modes of password guessing: a) heuristic or rule-based attacks, b) dictionary attacks, and c) brute force attacks. These are described below in order of the complexity and amount of time it typically takes to perform the attack.

1.6.1 Rule-based Attacks

Rule-based attacks make use of rules-of-thumb that users often follow when creating passwords. Rule-based attacks can be based on information retrieved through social engineering, as mentioned earlier. The reason these attacks are so effective is that human behavior is often predictable: Users create passwords that are meaningful to themselves personally because these passwords are easy to remember. Examples include birthdays, anniversaries, names of children, simple keyboard combinations ('qwerty,' 'abc123,' etc.), names of pets, and commonly-used passwords such as the word "password" or the username.

Organizational password policies often mandate that users create passwords composed of upper- and lower-case letters, numbers, and special characters in order to increase the difficulty of guessing passwords. Despite these policies, some users create password combinations that follow a formula that is guessable. For instance, a username followed by a number or special character, such as 'kvmcbride1,' or 'kvmcbride!' are common.

A rule-based attack generates passwords using a defined set of characters. Thus, guesses for a username 'kmcbride' might include: kvmcbride1, kvmcbride2, kvmcbride3 … kvmcbride*, kvmcbride/, and so on. Rule-based attacks work remarkably well because passwords that are created from a truly random set of characters are very difficult to remember. When passwords are difficult to remember, users often write their passwords down, and these passwords may be located through other procedures.

1.6.2. Dictionary Attack

A *dictionary attack* uses a list of words from a dictionary as the basis for guessing passwords. If the suspect uses a word contained in the dictionary, it will be guessed fairly quickly, no matter how long the word: 'Antarctica' will be guessed more quickly than 'cat' because it comes prior to it in a dictionary. Depending upon the size of the dictionary and the speed of the computer, in many cases, a dictionary attack may run through all the words in the dictionary in less than a minute.

Dozens of dictionaries can be downloaded from the Internet, including specialized dictionaries containing names of: cartoon characters, sports teams, or mythical or fictional characters from TV shows, movies, or literature. Dictionaries are also available in multiple languages. We suggest that law enforcement agencies download several sets of dictionaries, including some of the largest dictionaries available, and use all of these dictionaries before moving to the more time-consuming brute force attack.

1.6.3 Brute Force Attack

The most time-consuming type of password attack is the brute force attack. A brute force attack looks at combinational possible combinations of letters, numbers, and special characters, and uses them in guessing the password. A password eight characters in length that uses upper and lower case letters (52), numbers (10), and special characters (32), means there are 94^8 or 6,095,689,385,410,816 possible combinations. Brute force attacks on passwords more than eight characters in length are generally unfeasible. Use brute force attacks as a last resort only.

1.7 Passwords on Disk

Accessdata's Password Recovery Toolkit (PRTK) is a commercial password cracker (www.accessdata.com) that works for many encryption problems. PRTK has a very interesting password cracking methods that works in concert with Accessdata's Forensic Toolkit (FTK). This method is based on the fact that the user's password may appear somewhere on the suspect's hard drive. For instance, the suspect may have included the password in a document, or an ill-behaved encryption application placed the password in RAM, and was subsequently written to a swap or hibernation file. FTK will extract and index every word on a hard drive and output this list to a text file, which can is then imported into PRTK and used as a dictionary. If the password appears anywhere on the hard drive, in allocated, unallocated, or slack space, then the PTK will break the password.

1.8 Break Other Accounts

Humans are creatures of habit, as they saying goes. Most users employ the same password for many different accounts: Why remember 10 passwords for 10 accounts when I only have to remember one! A 'careful' user may use multiple passwords, but to reduce cognitive load, select passwords that fall into a category that is meaningful to the user, for instance, characters from Star Trek, a pet's name, etc. It may be useful for an agency to attempt to (legally) break passwords for other accounts to which the suspect has access to determine if the suspect uses a guessable pattern, e.g., ckirk!, mspock!, msulu! (i.e., Star Trek characters.)

1.9 Commercial and Freeware Tools

There are commercial and freeware tools that perform password cracking. There are several free password crackers that are very good and widely used, including John the Ripper (www.openwall.com) and Crack ftp.cerias.purdue.edu/pub/tools/unix/pwdutils/crack. One of the best known and efficient applications is L0phtCrack, a commercial tool from @stake (atstake.com). Password dictionaries may be found at several locations on the Internet, and are easily located by using Google to search for "password cracking dictionary."

2.0 Forensic Analysis of Steganography

Steganography is a term that means 'covered writing.' Steganographic algorithms take the bits that comprise a message and embed these bits within another file (Wayner, 2002). The most common example is hiding text within a graphical image, which is demonstrated below.

There are several steganographic algorithms. One of the more common uses the least significant bit of a byte from the file to be hidden; exchanging that bit with the least significant bit from a byte within the cover medium, or file the data is to be hidden in. On average, no more than 50 percent of the bits from the cover medium (also called the receptacle image) are changed in this process. Because they are the least significant bits, very little of the receptacle will display an obvious change.

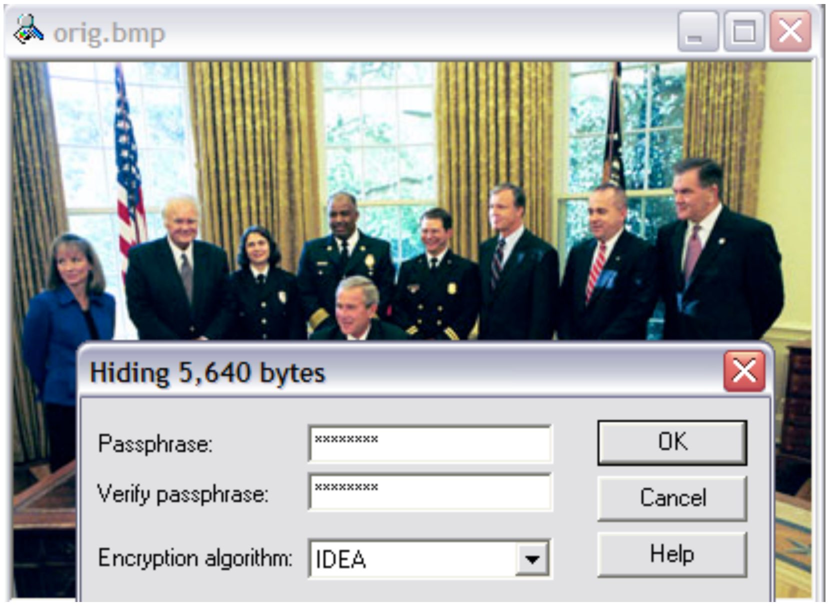
For example, in order to hide the letter 'Z,' we first convert it to the ASCII character 90 in decimal, or '01011010' in binary form. We need eight bytes of data to hide the ASCII value of "Z.". If we have the following eight bytes of data, inserting the ASCII character "Z" will yield:

```
1110101 → 1110100
1101101 → 1101101
1001100 → 1001100
0110110 → 0110111
0010111 → 0010110
0101000 → 0101000
0000001 → 0000001
1100001 → 0000000
```

Here we've hidden the character Z, which in this case only required changing 3 (for bytes 1, 4, & 8) of the least significant bits in the 8 bytes of data we selected. We can of course recover the hidden text by stripping the least significant bit from each byte and piecing the resulting bits back together.

2.1 Steganographic demonstration

Various steganographic algorithms and programs will hide digital data within almost any other form of digital data. Evidence can be hidden in audio files, various types of graphical images, text files, and other files. Below we illustrate hiding the text, consisting of nearly 13,178 characters, from a speech by Osama Bin Laden (http://english.aljazeera.net/NR/exeres/79C6AF22-98FB-4A1C-B21F-2BC36E87F61F.htm) within a 321,000 byte bitmap graphic.



Hiding 5,640 bytes

Passphrase: *********

Verify passphrase: *********

Encryption algorithm: IDEA

Figure 4. Original Image from www.whitehouse.gov

We are hiding a large document; therefore, we need a large number of bits in which to exchange the bits to be hidden. The example below uses the freeware steganographic program S-Tools to hide the text of the Bin Laden speech. A password was also used to make it more difficult to extract the contents of the file. (Note: only 5,640 of the 13,178 bytes need to change, given that on average, only 50% of the bits will change.)

Note that there is some loss of information because of the transposition of bits, which causes degradation in the quality of the image. However, this is typically not discernable to the human eye. Figure 3 shows the two images transposed.

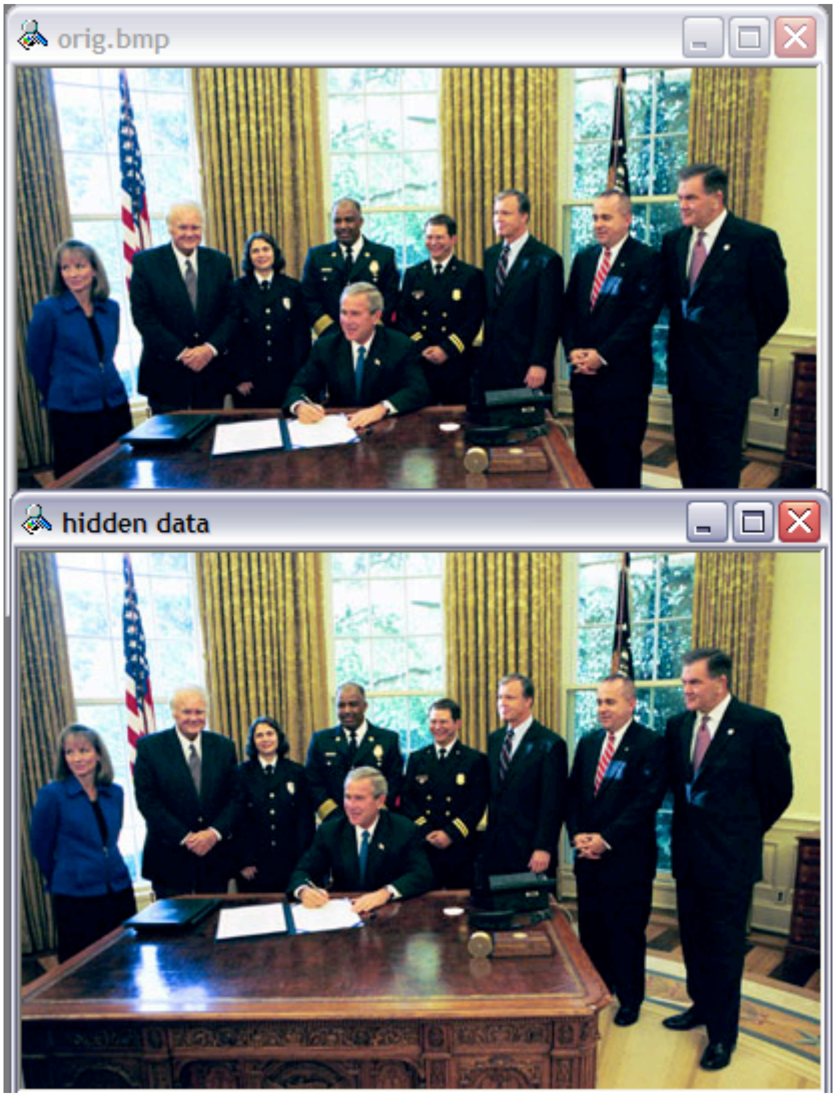


Figure 5. Comparison of original and stegoed file.

To demonstrate that the files are distinct, we calculated the MD5 cryptographic hash of each file:

```
f5bba91ee25a2036e064ab444ff0e10c  orig.bmp
e1b6ce63f2c438e7435845dea23c90bd  stego.bmp
```

Although both files are the same size, 321,654 bytes, the two cryptographic hashes of the original and hidden file are different, indicating that the two files have distinct content.

In order to recover the hidden text it is necessary to know the application that was used to hide the image, because different applications may use different algorithms to hide data. If someone attempts to extract the hidden content but doesn't know the password, the extraction process essentially fails.

2.2 Coping with Steganography

A major problem for law enforcement with respect to steganography is that the information is "hidden in plain sight." A document can be embedded in an MP3 file or a graphical image, and if the MP3 is played or the graphical image is viewed by an investigator, it will not be obvious that information is hidden within the medium. A web page may contain dozens or hundreds of graphic images, any of which may or may not contain an embedded message. It is impossible to determine through visual examination whether a graphic file contains vital evidence embedded as hidden data.

The best clue that a suspect has used steganography is often provided by a search for steganography tools on a suspect's computer. If steganographic tools or applications are located, the next task is to determine which files contain embedded information. The first and best option is to ask the suspect at the outset which files contain hidden information. If the suspect refuses to divulge that information, there are two options. First is to use automated tools that can detect, albeit imperfectly, hidden information. Second is to conduct trial-and-error experimentation as described in the prior section on encryption.

Several automated tools are available to evaluate the frequency of bits within a file to determine whether the file has embedded steganographic information. Stegdetect (www.outguess.com) is a freeware application that calculates statistics on a graphic as a means of determining whether an image contains hidden information. The listing below demonstrates the use of Stegdetect to determine if steganographic content is present in several files:

```
# Stegdetect *.jpg
testimg.jpg : jphide(***)
testimgp.jpg : jphide(***)
testorig.jpg : jphide(***)
testprog.jpg : jphide(***)
travel.jpg : negative
```

In the listing above we used Stegdetect to determine that four of the five files contain steganographic contents, and it includes the name of the application that was most likely used to hide the content. In this example, JPhide, a popular Windows-based steganography program, was identified. Stegdetect correctly identified the files that contain steganographic content, as well correctly determining that travel.jpg did not contain any embedded information.

Various other tools are available that use similar calculations to determine whether a file has hidden information. We have found that these tools are not foolproof; they can miss files with embedded information.

2.3 Manual Methods of Determining Steganographic Content

A suspect may leave clues as to files with steganographic content. For instance, in the presence of steganography programs, one should conduct a visual search for all graphical files. Two graphical files that appear to be the same visually but have different cryptographic hashes may be evidence of embedded information. The graphic file with the latest creation or modified date and time is more likely to the file with embedded content.

Steganography programs often use a password to increase the difficulty of extracting the hidden content. As discussed in the section on encryption, there are several avenues one can pursue for recovering passwords. The first and easiest is simply to ask the suspect for any passwords he or she uses. It may come as a surprise how often suspects freely provides passwords when asked.

REFERENCES

1. Craiger, J.P. Computer forensics procedures and methods. To appear in H. Bidgoli (Ed.), *Handbook of Information Security*. New York: John Wiley & Sons, 2005

2. Craiger, J.P., Pollitt, M., & Swauger, J. Digital evidence and law enforcement. To appear in H. Bidgoli (Ed.), *Handbook of Information Security*. New York: John Wiley & Sons, 2005.

3. Wayner, P. *Disappearing Cryptography Information Hiding: Steganography and Watermarking*. San Francisco: Morgan Kaufmann, 2002

* pcraiger@mail.ucf.edu, Voice: 407.823.3527