International Journal of Applied Information Systems (IJAIS) – ISSN : 2249-0868
Foundation of Computer Science FCS, New York, USA
2nd National Conference on Innovative Paradigms in Engineering & Technology (NCIPET 2013) – www.ijais.org

Study of Significance of Data Hiding and Data Finding Techniques from the Context of Digital Forensics

Manish H. Bhagwani
Research Scholar
MATS University, Raipur, CG, India

Rajiv V. Dharaskar
Director
MPGI, Nanded, MS, India

ABSTRACT:

This paper focuses on the initial definitions of data hiding and digital forensics followed by study of different data hiding techniques, their need, characteristics, parameters of evaluating those techniques and advantages. The ways of how to get the hidden data during investigation by computer forensic investigator are also discussed.

Keywords:
Data Hiding, Digital Forensics, Computer Forensic Investigator

1. INTRODUCTION

With the widespread use of multimedia information, digital forgeries and unauthorized sharing (piracy) of digital content have emerged as a growing among multimedia content creators, distributors and users. This trend has been further exacerbated by a diverse set of factors including: the growth of the Internet, the proliferation of low-cost and reliable storage devices, the deployment of seamless broadband networks, the availability of state-of-the-art digital media production and editing technologies, and the development of efficient multimedia data compression schemes. Today multimedia information piracy alone has subjected the entertainment industry to multi-billion dollar revenue losses. Traditional digital content protection techniques, such as encryption and scrambling, alone cannot provide adequate protection of copyrighted contents, because these technologies are unable to protect digital content once they are decrypted. Therefore, there is an urgent need to develop robust content protection technologies to develop systems capable of providing diverse services such as, secure media streaming between users and content servers, ownership protection, unauthorized copy prevention, unauthorized content usage, content authentication, content preservation, and content usage tracing. Data hiding, a form of steganography, embeds data into digital media for the purpose of identification, annotation, and copyright. Several constraints affect this process: the quantity of data to be hidden, the need for invariance of these data under conditions where a "host" signal is subject to distortions, e.g., lossy compression, and the degree to which the data must be immune to interception, modification, or removal by a third party.

2. LITERATURE SURVEY

Bender et. al. [1] explored both traditional and novel techniques for addressing the data-hiding process and evaluated these techniques in light of three applications: copyright protection, tamper-proofing, and augmentation data embedding. High embedding capacity, minimal embedding distortion, robustness to adversary attacks, low false positive rate, low error probability of received data, etc. are the desired features of a data hiding scheme. The development of data-hiding methods requires many design and quality tradeoffs, i.e., inaudibility of the embedded information, embedding capacity, and robustness of the embedded information. Fidelity, capacity, and robustness together define what is called the *data-hiding performance triangle*. These three data-hiding performance measures are interdependent and it is hard to maximize them simultaneously. In general, based on the application of data-hiding at hand, one or two vertices of this performance-triangle can be optimized while keeping the remaining parameter(s) within acceptable bounds.

According to Hal Berghel [2], data hiding has been with us as long as there have been digital computers and networks. Some of us are long enough in the tooth to remember data hiding on tracks above 80 of the ubiquitous 5 ¼” double-sided, double density floppy drives in the late 1970's. It was not uncommon to store a program key on the upper regions of the disk for copy protection of PC software. The simplicity of this scheme was elegant: the DOS operating system would only recognize the first 80 tracks, so the program key would be lost during any DOS copy procedure. This became one of the more common techniques of data hiding in the early microcomputer era, although its effectiveness was short-lived because applications programs could access the out-of-standard tracks directly by bypassing the operating system function calls and accessing disk controller directly. This gave rise to a cottage industry of copy-protection-defeating software as “bitsmiths” quickly developed controller-based copy software that rendered this form of out-of-standard copy protection obsolete.

Data hiding schemes are characterized by the number of defining properties. In general, a data hiding scheme is assumed to withstand against common data manipulations, such as, addition of while and colored noise, lossy compression, digital-to-analog conversion, rescaling, requantization, resampling, filtering, format conversion, encryption, decryption, scrambling, etc. In addition, robustness against active adversary attacks as long as attack-channel distortion is below noticeable distortion in the distorted media are also desired attributes of the robust data hiding schemes. However, the relative importance of each attribute depends on the data hiding application. For example, audio watermarking system for copy control application should withstand against short time energy ratio attack an active adversary might use this attack to deceive the detector. However, such robustness might be irrelevant for broadcast monitoring applications. Therefore, the performance of a given data hiding scheme should be evaluated based on the underlying application, a data hiding scheme is designed for [5-8].

Criminals with sensitive information such as crime records tend to hide/encrypt this information so that even if their computers are collected by police department, there is no evidence that can be used against them. The most common ways are data encryption and steganography. File system, in addition, can also be used to hide data. It is possible to hide data in NTFS file system and some analysis techniques can also be applied to detect and recover hidden data [4].

3. DATA HIDING TECHNIQUES, EXAMPLES, ANALYSIS

This section deals with the study of various methods of hiding data and files.

3.1. File extensions

A widely used and popular method of hiding a file type is to simply change the extension at the end of a filename. Try it:

1. Change the .doc extension on an unimportant Word document to .xls. Click Yes when the warning message appears.

The icon changes from a Word icon to an Excel icon.

2. Double-click the file to try to open it.

Because the extension indicates that the file is an Excel file, Excel opens. But the file fails to open because Excel can't open Word files.

3. Launch Word and then open the file with the .xls extension.

The file opens.

4. Change the .xls extension back to .doc and notice that the icon changes too.

5. Double-click the file to open it.

It opens.

To find out whether an extension has been changed, you need to compare the file header to the file extension to make sure that they match. The *file header* is a sequence of bits at the beginning of a file and is used by programs to determine whether they can open the file. During a computer forensic investigation file headers are extremely valuable because they allow us to locate the contents of deleted files, user activity logs, registry entries, and other relevant artifacts [3].

Even when the file extension is changed, the appropriate program still opens the file. On the other hand, when the file header is changed, the program no longer recognizes the file. Advanced users can change the file header easily by using a hex editor to make the file readable or unreadable. A hex editor is a program that can access data directly where it is stored without the need to know what type of format it is. Hex editors literally read data byte by byte and have the ability to change files at the byte level.

3.2. Hidden files

All operating systems assign attributes to files. One particular type of attribute is the ability to hide files, or more precisely, to mark files as hidden, which is comparable to files being marked for deletion. Hidden files are no more hidden than deleted files are deleted. If you use Microsoft XP or Vista, you can show any hidden files by selecting the Show Hidden Files and Folders option in the Folder Options dialog box. If you have an older file system, such as Microsoft Disk Operating System (DOS), use the Attrib command to either hide the file or make the file viewable.

3.3. Hidden shares

Hidden shares are shared areas on a network where files are stored but the shares are hidden. Hidden shares can be found on a local computer, but with networks everywhere, savvy criminals can use hidden shares on remote computers rather than risk using their own machines. Finding hidden shares is a bit more difficult than finding hidden files, but if you have the proper software, such as Legion V2.1 (www.packetstormsecurity.org), the process is straightforward. In addition to hiding shares, users sometimes also put passwords on hidden shares to protect them in depth. You can add a dollar sign symbol ($) to the end of the share so that it appears hidden and not visible from a network browser.

3.4. Alternate data streams

The uncommon data storage concept of *alternate data streams (ADS)* started with Windows NT version 3.51 and was introduced as a compatibility fix for the Macintosh HFS system. The implication of this fix is that you can piggyback data onto an existing file without changing the attributes of the first file — with the exception of the time stamp. These data streams allow multiple forms of data to be associated with a file. A clever user can hide extremely wicked files in this manner because the files don't show up using a DIR (directory) command, nor do they appear in Windows Explorer. A few antivirus programs can pick up ADS information, but for the most part the majority of the computer world is forgetful to the existence of ADS. One ADS scanner that one can try is from Pointstone (www.pointstone.com).

3.5. Layers

The simplest example to demonstrate the use of a layer to hide the data is to overlay a picture on text in a desktop publishing program. At first glance, you can see only the picture. After you move the picture, however, the text underneath is revealed. Another simple example is to change the font color of a document to the same color as its background. Open the file and all you see is what appears to be a blank page. If you come across a blank file (a file which appears empty when you open it such as a blank Microsoft Word page), print it. Hidden text may appear on the hard copy.

3.6 Steganography

Steganography, a complex version of layering and data hiding, is a modern-day version of an ancient communication method. *Steganography* refers to covered writing, such as invisible ink. In the digital world, this technique involves hiding a message inside an unobjectionable image, music file, or video that is posted on a Web site, e-mailed, or stored on a hard drive. Imagine downloading an image of the Brooklyn Bridge from the Internet. As a suspicious investigator, one can use your s*teganography*-detecting software to extract the message it is hiding. The problem is that, because many algorithms are used in s*teganography*, and without knowing which one was used, extracting the hidden information is quite difficult.

4. ALGORITHMS DEFEATING DATA HIDING

Several methods can be used to defeat data hiding, and each one has its advantages and disadvantages. Password can be obtained by using one of these methods:

4.1. Brute force

In this procedure, every possible combination of the key is tried until the right one is found and the password is cracked. This is trial-and-error method. For simple hashes or algorithms, brute force works fairly well. As the key length increases, so do the number of possibilities. As can be seen from the following table, a 512-bit key has more than 154 zeros behind it.

Table 1. Possible Key Combinations according to Key length

Key Length	*Number of*

in Bits	*Possible Combinations*
8	256
40	1,099,511,627,776
128	18,446,744,073,709,600,000
256	$1.15792 * 10^{77}$
512	$1.3408 * 10^{154}$

4.2. Dictionary attack

This word-based trial-and-error method uses a dictionary of passwords or hashes that are compared to the hash value stored on the suspect's password file. Dictionaries contain not only standard words but also the names of celebrities, sports teams, TV shows, etc. Despite how often people are told to use good passwords, they don't. The most common passwords found in the field are *password, letmein, 123456,* and *qwerty.* Other popular passwords are the user's first name, the names of children or pets, addresses, phone numbers, and even Social Security numbers. Using a dictionary doesn't mean that you're limited to words or even letters. Most password cracking software uses letters, numbers, and even special characters as part of their dictionary attacks. In a good password-cracking software program using a decent dictionary, the word *hello* and the character substitution *h3110* are cracked in less than a second.

4.3. Rainbow tables

These extensions of dictionaries are much larger hash databases that reside either on the Internet or with a private party. Rainbow tables let us use a larger database of possibilities than could be stored on a forensic computer.

4.4. Keystroke logger

Sometimes the best solution isn't to try to crack the encryption but, rather, to resort to investigation. Use a keylogger to capture the encryption keystrokes when the suspect types them. This method works well when you know that the person you're watching in a case is using some form of encryption. Keylogger features vary, but they all record the keystrokes typed on a computer keyboard. You can install keyloggers manually or use Trojan software. In addition to software keyloggers, physical keyloggers are installed between the keyboard and the back of a computer. This type of device is more difficult to install but on the other hand it cannot be detected by antivirus, antispyware, or anti-malware software.

4.5. Snooper software

This type of software is used in the same fashion as software keyloggers except that snooper software logs not only keystrokes but also almost any activity that occurs on the computer. Everything from screen shots to printouts, to chat sessions to e-mails, and even how many times the computer is turned on is archived. This type of software takes up quite a bit of room on the storage device, but can be extremely useful when re-creating passwords or passwords on a suspect's computer. This method works well in a situation where it is known ahead of time that the suspect himself is using a computer for illegal activities.

4.6. Suspect questioning

Many a times, it may happen that the suspect is the only option to gain access to a password or passphrase. Although most people don't initially supply their passwords, after some legal arm-twisting, it sometimes may occur.

4.7. Application specific integrated circuit (ASIC)

This type of computer chip is specifically programmed to perform a task. The sole purpose of programming an ASIC decrypting system is to crack a specific type of encryption. Most computer forensic investigators don't have access to computers of this type, but government agencies do, and they can chew through a 40-bit encryption key in only seconds with the help of these types of circuits.

4.8. Cache checking

Certain applications and operating systems may put passwords in a *cache* temporarily. Users who allow their systems to save their passwords so that they don't have to type them repeatedly are often saving their passwords in plain text mode in a cache area. This type of practice gives an opportunity to the investigator to get the password by using appropriate software which can scan *cache*.

4.9 Finding Out-of-Sight Bytes

To hide information, criminals use special software programs to identify the least significant bits (LSBs) in a file and change them to contain hidden content without altering the file in a detectable way. The best candidates for steganography are byte-intensive digital pictures and audio files because they have a good supply of insignificant bits. Even a plain text document can hide content within the structure of the file. Certain areas in files (depending on whether they're video or audio or some other type) can be modified without compromising the quality of the file to the human eye or ear. The major forensic issue is exposing the presence of hidden data.

4.10. Clue Finding Methods

There are several methods to find clues as to whether a file *might* have a hidden message in it:

4.10.1. Look for steganography software on the suspect's computer

An open clue is to find steganography creating software on the suspect's computer. The trick is to recognize the different types (experience is needed here) or known hash values of steganography software using hash analysis. Many investigators have no clue how many steganographic software packages exist and may overlook the software as being "just part of the system." There are several steganography softwares available for Windows. One of them is JPHS. This software gives details about the original file, the hidden file, and, toward the bottom, the new file with the steganography.

4.10.2. Look for duplicate files

While doing a forensic analysis, the investigator may find a huge number of duplicate files. Steganography often produces duplicate files because the original file is often left behind by careless criminals. When two files are found that look the same or are named the same, you have some major clues to work with. The types of files found indicate the type of steganographic software that is used. Certain types of steganographic software work with only specific file types, such as video or audio files. Using forensics software, compare the files on a bit-for-bit scale with a hexadecimal editor to find the differences and further narrow the possibilities of which steganographic software was used. Because you now have two files to work with, the investigator can also run a statistical analysis to see which file falls outside the expected digital signatures of a typical file of its type.

4.10.3. Use steganographic detection software

Software such as Gargoyle (www.tucofs.com) can be used to detect files that have steganographic signatures. They may not always detect it, though, if a new algorithm was used or the algorithm is so good that it escapes detection. The investigator can use these basic tools to find files that have been used to hide data and to discover the steganographic software that was used. Unless the same software is used, the chances of extracting the hidden data are zero.

4.11 Cracking Passwords

Passwords are not of equal strength and may be only part of an attempt to authenticate a person's attempt to gain access to a computer or file they are protecting. From a user's perspective, a password is easy to remember but hard to guess. It can be a word, phrase, hash, or even biometric (something unique about someone biologically, such as a fingerprint or voice print). From a computer forensic investigator's perspective, a password is a barrier to get past to complete the investigation. In most password applications, the password isn't even used to authenticate; rather, a hash value is used. A *hash value* (or simply *hash*) is the result of applying a one-way algorithm to a password. The reason for the one-way algorithm is to keep would-be intruders from reverse-engineering the hash back into the password. In other words, when you type a password, the computer is hashing the data you typed and comparing the result to the hashed password that's already saved. If both hashes match, the password is the same one that was entered originally. Why use a hash in the first place? The most obvious reason is that storing plain text passwords isn't secure. Replacing plain text passwords with a one-way hash value can exponentially increase the security of passwords. To put this concept into perspective, suppose that an MD5 hash is used to hide a password. Roughly 8.5 billion combinations for an 8-character password exist, give or take a billion. Years would pass before anybody could hit all those combinations. An even more secure version of a password is a *passphrase,* a phrase or short sentence that increases the number of possible combinations to strengthen the cryptographic hash. PGP (Pretty Good Privacy), a type of encryption software, is famous for the use of a passphrase and the difficulty of cracking the PGP hash. The MD5 has only a 128-bit key size, but PGP with passphrases can use, for example, a 2048-bit key size. Simply, cracking the encrypted data or even the pass-phrase by using a brute force method is almost impossible.

4.11.1. Disarming passwords to get in

a. Crack the easy passwords first

Human nature dictates that few people use different passwords for all the files or accounts they are trying to protect. Most people simply reuse their passwords repeatedly and change them slightly every time. The computer forensic investigator can take advantage of this situation because some applications are much easier to crack than others. Cracking a password in a word processing or spreadsheet program is so easy that certain shareware programs can accomplish this task quite easily. After you have one of these passwords, try the password you cracked on the more difficult algorithms to see whether you have a winner. You might be surprised at how often this technique works. If it doesn't work, try substituting characters or variations of the password.

b. Grab clues

When a user asks the browser to remember a site password to avoid having to type it repeatedly, the investigator can catch a break. Look in the cache for the passwords. Usually they're not the ones you want, but they can give him a clue to the target password or hints to how the user thinks. The Cain & Abel software shows a typical password cache dump. Pay attention to the line that reads Default Password: It shows you the password to access the Windows operating system.

c. Bring on the brute force crackers

If all else fails, you have to use password cracking software, such as Cain & Abel (www.oxid.it) or John the Ripper (www.openwall.com/john). They can crack a password by brute force or use a dictionary, depending on which clues you picked up during your search. Any hints you find to reduce the number of possibilities save you processing time in spades. If necessary, create a custom dictionary just for this case with all possible passwords that this particular user may have used. Be sure to check pet names and favorite teams.

Applications such as word processors, databases, and spreadsheets often save their data in formats that can be read with a hex editor. For example, one can view the file contents in raw form using a hex editor such as WinHex and not even have to break the password. Keep in mind that the formatting disappears and you see strange characters, but some of the data is in human-readable format. Other extremely technical methods exist for attacking a file and working around a password. The cost in time and money, however, often isn't worth the effort unless your organization's initials are in the three-letter formats FBI, DHS, CIA, or NSA.

5. CONCLUSION

Various data hiding techniques have been studied and ways to uncover the hidden data are also discussed from the context of computer forensic investigator. This study works as the basis for an investigator before moving towards using data recovery software on one side and digital forensics researchers on the other side. There is still scope of discovering the new ways of data hiding in the modern world of new gadgets and softwares. This paper will hopefully work as a launch pad for those persons who want a concise study on the topic.

6. REFERENCES

[1] Bende, W. Gruhl, D. Morimoto, N. Lu, A. 1996. Techniques for data hiding. In IBM Systems Journal **Volume:** 35, Issue: 3.4 Page(s): 313 – 336.

[2] Berghel, Hal. 2007. Hiding Data, Forensics and Anti-Forensics. In Communications of ACM, April 2007.

[3] Rowe, Jon. 2008. What are File Headers? (Signatures). At http://www.pinpointlabs.com/wordpress/2008/08/12/what-are-file-headers-signatures/. Articles from Pinpoint Labs Blog. 12-08-2008.

[4] Cheong Kai Wee. Analysis of hidden data in the NTFS file system. At http://www.forensicfocus.com/hidden-data-analysis-ntfs.

[5] Cox, I., Miller, M., and Bloom, J. 2001. Digital Watermarking. Morgan Kauf-mann.

[6] Wu, M. and Liu, B. Multimedia Data Hiding. New York, NY, Springer-Verlag, 2002.

[7] Eggers, J. and Girod, B. 2002. Informed Watermarking. Norwell, MA. Kluwer Academic Publisher.

[8] Sencars, H., Ramkumar, M., and Akansu, A. 2004. Data Hiding Fundamentals and Applications. Elsevier Academic Press.