Theoretical and Practical Aspects of Encrypted Containers Detection - Digital Forensics Approach

Ireneusz Jozwiak[1], Michal Kedziora[2], and Aleksandra Melinska[3]

[1] Wroclaw University of Technology, Wroclaw, Poland
[2] Wroclaw University of Technology, Wroclaw
email: Michal.kedziora@pwr.wroc.pl
[3] Wroclaw University of Technology, Wroclaw, Poland

Abstract. This paper covers problem of detecting encrypted files in evidence data during digital forensics investigations. We present comparison of popular detection methods like file signature and extension analysis, metadata analysis and searching operation system artifacts. We present research on theoretical and practical use of some indicators that can suggest encryption used like entropy, chi-square test, Arithmetic Mean and Monte Carlo Value for Pi.

1 Introduction

This paper describes theoretical and practical aspects of detecting encrypted data and files during computer forensics investigation. We start with simple and obvious methods like file signatures analysis, but our main research is done with advanced mathematical statistical methods of encryption detection that can be used nowadays.

Digital forensics investigation is wide and complicated process. One of popular definition says that computer forensic is "the use of scientifically derived and proven methods toward the preservation, collection, validation, identification, analysis, interpretation, documentation, and presentation of digital evidence derived from digital sources for the purpose of facilitation or furthering the reconstruction of events found to be criminal, or helping to anticipate unauthorized actions shown to be disruptive to planned operations"[17], so we can see clearly that one of the main objective of computer forensic investigation is to search and analyze any data and files which may contain any interesting information.

Naturally one of the first steps, which people do to make their information safe is to hide it. It is not so unusual that users move valuable files into system folders or rename them to make them inconspicuous[6]. Of course usually this methods are not much effective, further more in the most cases those files are highlighted and revealed on the beginning of investigation. Some of techniques used for that aim will be described later e.g. hash analysis, binary search, time line analysis and signature analysis. More advanced users are aware that hiding data in the best case will delay revealing of evidences, but in most cases it will make it even faster[7].

W. Zamojski et al. (Eds.): Dependable Computer Systems, AISC 97, pp. 75 – 85.
springerlink.com

That is one of the reason why users encrypt valuable data[13]. It should be said here clear that encryption is not reserved for criminals who want to hide evidences of illegal actions. Encryption is part of computer security, we can encrypt data to make it safe from being steeled, from competition, sometimes legal regulations force us to encrypt fragile data. Whatever reason of encryption is, the first question digital forensic investigator will ask himself will be why suspect encrypts his data, maybe he want to hide something important. That is why it should be priority to find those files in computer forensic investigation process.

There is plenty of free and commercial tools to encrypt and decrypt data[19] and to create encrypted containers where users can copy files to make them encrypted. We decided not to describe different cryptographic algorithms because from practice we can observe that it does not matter. It is because in the most computer forensic cases there is no need to perform full cryptanalysis process. It is because most of cryptographic algorithms are based on Auguste Kerckhoffs rule that says that algorithms security must be based on well-known algorithm and secret password. For sure this rule is positive, but there is a huge sore point of this approach – password. Research are pointing that most users use very week passwords[20]. Microsoft research which involved half a million users over a three month period have given result of average password bit strength equal 37.86. This means passwords where short, included mainly lowercase letters, without uppercase, digits and special characters. Other research based on password lists revealed by hackers can conclude that most of password are short words which can be find in dictionary, or words related to personal details of user (name of pet, date of birth). For sure it is huge issue for security professionals how to convince users to use harder passwords, but for forensic investigators it is easy ability to guess password and analyze decrypted data. Next problem with passwords (or more with users) is that one password is almost for sure used in several applications (mail, webpage, logins). This means that during investigations we can find hardly protected passwords from e.g. mail or internet browser (where they are typically stored in clear text), and then use this passwords to decrypt files. Third way to get encrypted data password during digital forensic investigation is to ask suspect (or ask prosecutor to do it)[9][10]. In some law systems for example in UK, if you will not give passwords that authority believe you possess, you will have to serve a sentence 5 years in prison for failing to comply with police or military orders to hand over either the cryptographic keys, or the data in a decrypted form. Actually 5 years imprisonment is reserved for terrorism cases, and 2 years sentence in any other cases[8].

When we put all this together we can conclude that in contemporary computer forensic investigations, the biggest problem is not to decrypt encrypted files but to find those encrypted. The chapter is divided into three parts. The first part will present methods of detecting encrypted files based on file metadata. These are simple and fast methods to discover encrypted files. The second part will present advanced statistical algorithms we can use to detect not only encrypted files but also encrypted data hidden in unallocated space. In the third part we present

indirect methods to detect that encrypted files can be found on the disk. Those methods do not point to encrypted files but they tell that is high possibility that those files are located on a disk.

2 Signature Analysis

Methods presented in this section are based not on encrypted data inside a file, but on file metadata. This methods are not quite accurate but there are fast and in some cases they can give immediate results.

First method is based on searching files with specific extensions[11]. Encryption tools usually create unique filename extensions to identify them by operation system. This process is called Application Binding and it is very comfortable situation for user because operation system can associate encrypted file with specific software and decrypt it with minimal user action. From computer forensic point of view searching files by specific extension is the easiest way to find specific encrypted files. Some example of encrypted files extensions: .aes (AES Crypt encrypted file), .asc (Pretty Good Privacy armored encrypted file), .axx (Encrypted file), .cef (SanDisk CruzerLock encrypted file), .cry (Cryptainer encrypted volume file format), .dez (DES encrypted zip file), .drc (DriveCrypt container file), .fcfe (Microsoft Access encrypted database), .jbc (BestCrypt file), .pgp (Pretty Good Privacy encrypted file), .sde (Steganos Disk Encryption file format). We must know that Windows Operation Systems uses extensions to associate files with applications but in Unix not extension but header is mostly used. In MAC OS is used combination of extension, header and 32bit metadata information called file type code. Process of identifying and comparing extensions, headers and footers of files is called file signature analysis. It is part of standard digital forensic investigation procedure that is why it can be successfully used also to detect files which signatures point to be encrypted files. The signature analysis can also detect files which extensions were changed on purpose, to mislead investigation (it is common to change .jpg or .zip files into .sys, or .temp). Performing signature analysis we can have four outputs[5]. First is when header of file is known and extension is also known and mach. This is absolutely normal situation where file e.g. example.zip as Zlock pro encrypted ZIP has file header equal 50 4B 03 04 14 00 01 00 63 00 00 00 00 00 and extension ZIP. Second situation is when header and extension is known, but extension not match with header, corresponding to last example it would be example.dll, where someone intentionally has changed extension to mislead investigation. For sure this attempt will be discovered in a short time. Third situation is when file header and extension are both unknown, this situation can occur by dealing with e.g temp files. Fourth situation is when header is unknown and extension is known but doesn't match. This case can be spot during processing True Crypt files. Encrypted True Crypt containers usually have default .tc extension but it is not mandatory. True crypt containers can have any extension to work properly and file itself does not have any signature. Often true crypt encrypted containers have extensions of other files to mislead investigation. It can also be used as a trace because relatively small amount of files are without any

header. As we can see signature analysis is a powerful tool to discover encrypted files in operation system, but still we have to realize that this method does not detect encryption used but points that file was created by cryptographic tool. To confirm that file has encrypted data inside we have to adopt statistical algorithms.

3 Statistical Data Analysis Algorithms

To detect encrypted data in the most accurate way, we have to understand mathematical fundaments of cryptography. As a definition encryption function εk with key k, where c is cipher text, p is plaintext is equal to:

$$\mathrm{c} = \varepsilon_{\mathrm{k}}(\mathrm{p}) \tag{1}$$

In the beginning of cryptography history there where simple substitution ciphers which weren't very strong. Cryptanalyst could easily guess massage by analysing the frequency distribution of the cipher text. In the course of time ciphers were evolving into block ciphers, which were not so easily broken with frequency analysis. One of the techniques to prevent cryptanalysis of cipher, was called whitening[15]. At the end cipher text started to look like random data stream. It helped to make cryptographic algorithms stronger, but it also made possible to identify encrypted data by measuring randomness of data[4]. In fact there is no other simple explanation to keep random data then the file is encrypted massage. Statistical data analysis encryption detection algorithms are based on statement that encrypted massage is similar to pseudorandom data[14]. There are several methods and techniques to test data sequences to be random. Most of them are based on Golomb's randomness postulates[12]. The first postulate tells that in the cycle N of s, the number of 1 differs from the number of 0 by at most 1. Second postulate tells that in the cycle N, at least half the runs have length 1, at least one fourth have length 2, at least one eighth have length 3, etc., as long as the number of runs so indicated exceeds 1. Moreover, for each of these lengths, there should be equally many gaps and blocks. Third postulate tells that the autocorrelation function C(t) is two valued. That is for some integer K,

$$N \cdot C(t) = \sum_{i=0}^{n-1} (2s_i - 1) \cdot (2s_{i+1} - 1) = \begin{cases} N, & t = 0 \\ K, & 1 \le t \le N-1 \end{cases} \tag{2}$$

These postulates are not sufficient to consider data as random but there are necessary to be fulfilled by all random data. Next we present chosen algorithms which can be used to determine if data is much or less random.

3.1 Entropy Based Detection

First factor we present will be entropy. Originally entropy comes from physics and thermodynamics and is a measure of the disorder or randomness of the

constituents of a thermodynamic system. Entropy was adopted into computer science where it represents measure of the uncertainty associated with a random variable[1]. From definition entropy H of a discrete random variable X with possible values $\{x_1,\ldots,x_n\}$ is equal:

$$H(X) = E(I(X)) \tag{3}$$

Where I is the information content of X. I(X) is a random variable and E is the expected value. If p denotes the probability mass function of X then entropy is equal to:

$$H(X) = \sum_{i=1}^{n} p(x_i)I(x_i) == -\sum_{i=1}^{n} p(x_i)\log_b p(x_i), \tag{4}$$

Where b is the base of the logarithm. Entropy value will be close to max value when the input will be random data. Any signs of data order will lower entropy value. We can predict efficiency issues while using this algorithm to compute entropy, it is because we have to compute logarithms. To avoid this some encryption detection tools are implemented with simple other simpler randomness tests[3].

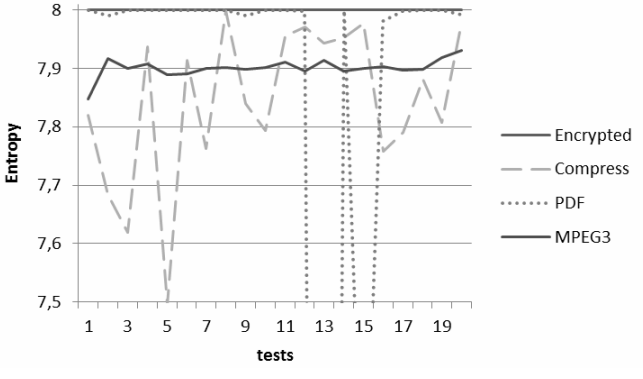

Fig. 1 Entropy test values

After testing various types of files we received range of encrypted data to be from 7,99984 to 7.99999. Compressed files (mainly we used files with zip and rar compression algorithms) where just slightly lower, but difference is almost unnoticeable. This tells as that method of encrypted data detection can have false positive hits with some compressed files. With other types of files we do not have such doubts. PDF files have entropy based of kind of data they consist, but in most cases entropy level is much lower than 7,99. MPEG3 files have entropy at level around 7,9 and its constant in different files, it can be explained by compression algorithm used in this file types.

3.2 Chi Square Test

Chi Square test is a statistical hypothesis test in which the distribution of the test is a chi-square distribution when the null hypothesis is true[2]. There are several tests build on this assumption, but the main use is to confirm randomness of data. From the definition chi-square[16] test with k probable outcomes, performed n times, in which Y_1, Y_2, Y_3… Y_k is the number of experiments which resulted in outcome, where the probabilities of each outcome are p_1, p_2,… p_k is:

$$\chi^2 = \sum_{1<s<k} \frac{(y_s - np_s)2}{np_s} \tag{5}$$

In result we should expect the lower chi square sum for more random data. From a chi-square, the probability Q that the X^2 sum for the test with d degrees of freedom is regular with null hypothesis and can be compute as:

$$Qx^2, \mathrm{d} = \left[2^{\mathrm{d}/2} \Gamma(\frac{d}{2}) \right]^{-1} \int_{x^2}^{\infty} (t)^{\frac{d}{2}-1} e^{-\frac{t}{2}} dt \tag{6}$$

Where Γ is a factorial function to complex and real arguments:

$$\Gamma_x = \int_0^{\infty} t^{x-1} e^{-t} dt \tag{7}$$

We have performed several tests with different encrypted, compressed, MPEG and PDF files. Encrypted files had chi square on constant value of near to 256.Any other files had Chi Square value thousands or millions higher. For compressed files average value was 16276778 It is 63581 times higher than Chi Square value of encrypted files. For MPEG and PDF files values where successively 9157435 and 5131275. Chi Square is extremely sensitive, that is the reason why it is used to check pseudorandom generators. It is also accurate way to detect encrypted files and distinguish them from any other.

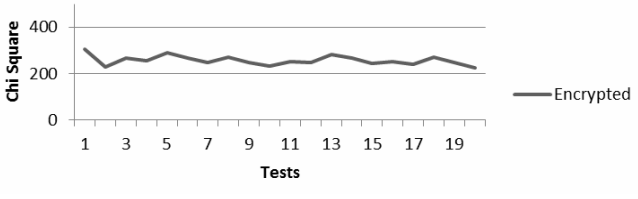

Fig. 2 Chi Square test values

3.3 Other Algorithms

There are other random detection tests like Frequency tests. For example monobit test which checks if number of values in data stream is near equal, for two bit we have:

$$X = \frac{(n_0 - n_1)^2}{n} \tag{8}$$

For bytes when we get 255 possibilities we can use modification of frequency test which is ease to implement based on equation:

$$X_h = n * \frac{\min(hist)}{\max(hist)} \tag{9}$$

This method of calculating entropy is based on histogram of data we are interested with. We take minimum value from histogram and divide it to maximum value. The smaller difference between these two values, there is the more random data. Output of division will give entropy near one for random data, and less random data will have near zero value. Output is multiplied by 8 to make it compatible with definition of entropy. To make output more comparable to entropy calculated from definition, outcome is multiplied by 8. Third simple of implementation of frequency test is summing all the bytes and divide it by the file length in bytes. We should get value about 127.5 for byte stream or 0.5 for bit stream. Any other value mean that data is not random from Golomb's randomness postulates.

We have performed a series of experiments on encrypted, compressed and other files. Result was that encrypted data is in conformity with random data frequency test and value is 127,5 ± 0,5. Compressed values are divergent but rarely in detection threshold of encrypted files. MPEG files have mean values much below encryption mean values in range between 124 and 125.

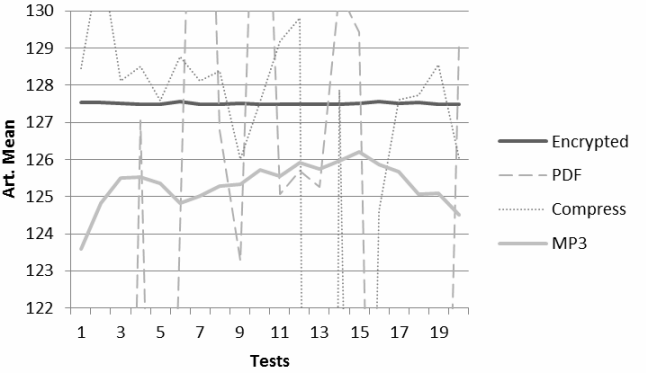

Fig. 3 Frequency Mean test values

Next algorithm is serial correlation X_{ac} which idea is to examine the correlations between the shifted sequences and it is computed from equation:

$$x_{ac} = \frac{2(A(d) - \frac{n - d}{2})}{\sqrt{n - d}} \tag{10}$$

Where $A(d) = \sum_{i=0}^{n-d-1} s_i \oplus s_{i+d}$, d is fixed integer, and s is a tested sequence.

Monte Carlo Value for Pi is algorithm in which each following 6 bytes sequence is use as 24 bit X and Y coordinate. If the distance from random point is less than the radius of a circle placed in the square, sequence is called hit and the number of hits can be used to calculate the value of Pi. If the sequence is random, value should be equal to Pi value 3,14159265.

After performing series of tests we can conclude that Monte Carlo Pi algorithm is efficient in detecting encrypted data. All encrypted files are values near Pi value of 3,14. Compressed files have clearly lower (or higher) values, MPEG and PDF files are much outside encrypted file values.

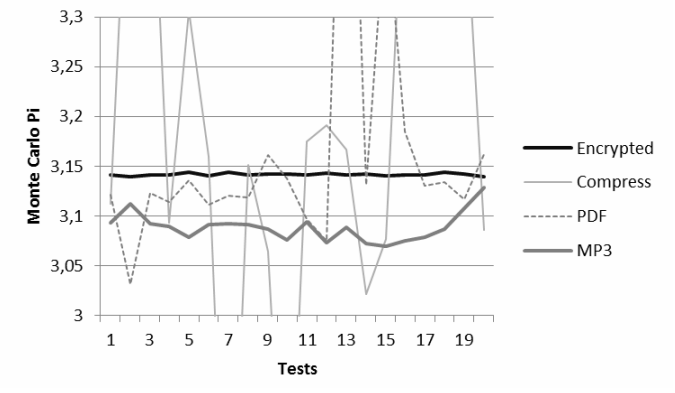

Fig. 4 Monte Carlo Pi test values

Other test algorithm which can be used for detection of random data are poker test which is generalization of frequency test for sequence values, two bit test which check occurrences of 00, 11, 01, 10 subsequence's, or runs test which checks number of chosen data blocks inside data sequence.

3.4 Data Length

One of the problem correlated with encrypted data detection based on statistical algorithms is length of input. If the data input will be to short algorithms will not have enough data to give accurate result. Main cryptographic tool used nowadays is True Crypt, fortunately minimum size of NTFS formatted encrypted container is 3792 KB. We prepared series of tests to check how file length corresponds with algorithms accuracy. We have performed described algorithms on several files from 148576 KB to 524288000 KB. Chosen results are presented on Table 1.

Table 1 Detection values with different file lengths

File Length	Entropy	Chi Square	Art. Mean	Monte Carlo Pi
1048576	7,99984	231,9458	127,4971	3,14201
8388608	7,999979	246,9846	127,516	3,141936
9437184	7,999979	270,3573	127,5294	3,143995
10485760	7,999982	255,3956	127,4814	3,141654
15728640	7,999987	291,6301	127,496	3,143619
19922944	7,99999	284,8628	127,5186	3,141409
20971520	7,999992	224,4152	127,4918	3,141268
26214400	7,999992	283,1507	127,5276	3,141438
31457280	7,999994	243,771	127,5057	3,14201
62914560	7,999997	225,1659	127,4941	3,1417
209715200	7,999999	264,4227	127,5002	3,141689
314572800	7,999999	265,5303	127,4986	3,141408
524288000	8	281,2614	127,5017	3,141526
1048576	7,99984	231,9458	127,4971	3,14201
8388608	7,999979	246,9846	127,516	3,141936
9437184	7,999979	270,3573	127,5294	3,143995

As we predicted length has some effect on entropy based algorithm, an average mean, and Monte Carlo Value for Pi, but it is not effective for chi square test and. Furthermore even for 148576 KB input is accurate enough to detect encrypted data and reject other files.

4 Indirect Methods

Main disadvantage of statistical methods when searching for encrypted data is amount of time necessary to search all hard disk area. In contemporary forensic investigations we often have to analyze Terabyte hard drives and it can make days to finish. Forensic investigators cannot afford to use all detection techniques because it would take to long amount of time, that is why we should consider use of some indirect methods which can say as there is high possibility to find encrypted data on the computer. These methods are not supposed to detect encrypted file themselves, but they detect e.g. artifacts of cryptography tools used by system. First technique is to search for any files or records which are related to encrypting tools. Most effective method to search files is to use hash analysis which is one of the basic tool used in computer forensic investigations. In first stage of analysis hash value is computed for every file on evidence disk image. Most often 128 bit Message Digest 5 Algorithm (md5) is used in this purpose. Second stage of hash analysis is to compare computed hash values with hash sets. This helps to exclude files which are not related to case, or highlight those which are related to e.g. malware, illegal software or in our case files created by cryptographic tools.

Next method is to find any traces of cryptographic tools in operation system registry. Even if application is uninstalled there should be remaining keys, forensic investigators can use registry backups (snapshots) to examine registry in specific time line.

Third method is time-consuming but it should be used in the most demanding cases. It is based on performing binary search of unallocated disk area using keywords related to cryptographic tools. If any of cryptographic tools traces are found we can expect that encrypted files can be stored on evidence disk image, and we have good reason to use superior methods of encryption detection based e.g. on entropy.

5 Summary

In the chapter we have explained that in computer forensic investigations one of the most challenging problems is to find encrypted data stored on disk. Most accurate solution would be using all detection techniques presented in this chapter. Unfortunately in most cases we have limited resources and we cannot do it. We think optimal algorithm should consist of hash analysis to exclude known files and to find any files correlated to cryptographic tools. Subsequently indirect method of detecting encryption tools should be used to specify most probable encrypted files types. Next step is to perform signature analysis. As a result we should get all files pointed as encrypted from its signature, and those which doesn't have any known header. Output files should be checked by statistical analysis to confirm or deny that they in very high probability are encrypted files. One problem which can be significant when using statistical algorithms is distortion caused by header of a file. Even if data of file is encrypted, headers usually are in plain text, which can affect result of algorithm in the way that detection algorithm values will drop below detection threshold. Simple defense is to bypass first KB of file. In case of True Crypt it is not necessary because container does not possess its own header. Our tests confirmed that encrypted files are possible to discover using statistical methods based on randomness tests. Most accurate results were given by chi square tests, where values between encrypted and other files where thousand times higher. Surprising was low accuracy of entropy based techniques in differencing encrypted and compressed files. Using entropy would produce a large number of false positive hits. On the other hand compressed files should be excluded from encryption detection by using signature analysis. Our future research will include efficiency issues with large data sets, to choose the most practical statistical algorithm.

References

[1] Hamming, R.W.: Coding and Information Theory. Prentice-Hall, Englewood Cliffs (1980)
[2] Knuth, D.E.: The Art of Computer Programming, Seminumerical Algorithms, vol. 2. Addison-Wesley, Reading (1969)
[3] Ziv, J., Lempel, A.: A Universal Algorithm for Sequential Data Compression. IEEE Transactions on Information Theory 23(3), 337–343
[4] Park, S.K., Miller, K.W.: Random Number Generators: Good Ones Are Hard to Find. Communications of the ACM, 1192 (October 1988)

[5] Steve, B.: The Official EnCase Certified Examiner Study Guide. Wiley Publishing, Chichester (2008), ISBN: 978-0-470-18145-4
[6] Liu, V., Brown, F.: Bleeding-Edge Anti-Forensics, InfoSec World (April 3, 2006)
[7] Rogers, D. M.: Anti-Forensic Presentation, Lockheed Martin. San Diego (2005)
[8] Regulation of Investigatory Powers Act 2000, ch.23, UK legislation (July 28, 2000)
[9] Schneier, B.: Rubber-Hose Cryptanalysis. Schneier on Security (October 27, 2008)
[10] Soghoian, C.: Turkish police may have beaten encryption key out of TJ Maxx suspect. Surveillance State, CNET Networks (October 24, 2008)
[11] Huebnera, E., Bema, D., Wee, C.K.: Data hiding in the NTFS file system. Digital Investigation 3(4), 211–226 (2006)
[12] Menezes, A., van Oorschot, P., Vanstone, S.: Handbook of Applied Cryptography. CRC Press, Boca Raton (1996)
[13] Eoghan, C., Stellatos, G.J.: The impact of full disk encryption on digital forensics. ACM SIGOPS Operating Systems Review 42(3) (April 2008)
[14] Hamming, R.W.: Coding and Information Theory, 2nd edn. Prentice-Hall, Englewood Cliffs (1986)
[15] Haahr, M.: An Introduction to Randomness and Random NumbersRandom (June 1999); Random.org
[16] Walker, J.: Introduction to Probability and Statistics. A Pseudorandom Number Sequence Test Program, Fourmilab (January 28, 2008)
[17] Marco, S.G.: Corresponding The birth of a new industry: entry by start-ups and the drivers of firm growth: The case of encryption software. Research Policy 33(5), 787–806 (2004)