Tatiana Diyachenko

STATISTICAL ANALYSIS OF THE UNIFORMITY OF CRYPTOGRAMS IN THE DYNAMIC CRYPTOSYSTEMS

Thesis

CENTRIA UNIVERSITY OF APPLIED SCIENCES

Degree Programme in Information Technology

November 2015

Centria
UNIVERSITY OF APPLIED SCIENCES

ABSTRACT

Unit **Kokkola -Pitarsaari**	**Date** [November 2015]	**Author/s** Tatiana Diyachenko
Degree programme Bachelor of Engineering, Information Technology		
Name of thesis STATISTICAL ANALYSIS OF THE UNIFORMITY OF CRYPTOGRAMS IN THE DYNAMIC CRYPTOSYSTEMS		
Instructor Grzegorz Szewczyk		**Pages** 39 + 3 Appendixes
Supervisor Grzegorz Szewczyk		

The topic of this thesis was first introduced as a part of a research concerning the application of cryptography in special cases. The tools for the work were developed according to the main scope of the project – pseudo random number generator, encryption using one-time pad algorithm and χ^2 test analysis.

The research work required knowledge on cryptology, introduction into its important tasks and requirements. The essential basics are introduced in this thesis. They allow not only to familiarize oneself with cryptology in general but also to understand the difference between various enciphering systems.

The main aim was to create the software allowing byte-by-byte encryption of plaintext to be encrypted using simple XOR operation with one-time pad algorithm. For the key, pseudo random number generator was used. Each type of file passed up to 15 rounds of encryption. After each round χ^2 goodness of fit test for distribution of bytes was performed. Results of the analysis of influence and necessity of particular amount of rounds for different types of plaintext are provided as the result of the research.

Keywords: One-time pad, chi2, encryption, goodness of fit, pseudo random number generator, statistical analysis.

1 INTRODUCTION 1

2 TERMINOLOGY 3

3 DEVELOPMENT OF CRYPTOGRAPHY 5

3.1 Cryptography as an art 5

3.2 Cryptography turn into science 6

3.3 New tasks for cryptography 9

4 PUBLIC KEY ALGORITHMS 11

4.1 Diffie-hellman key exchange protocol 11

4.2 RSA encryption system 12

4.3 Rabin's scheme 14

4.4 The El Gamal digital signature 15

5 SYMMETRIC CIPHERS 16

5.1 Data encryption standard 17

5.1.1 Details of des encryption 18

5.2 Advanced encryption standard 19

5.2.1 History of development of aes 19

5.2.2 Details of aes encryption 20

5.3 Stream type encryption 20

5.3.1 Synchronous stream encryption 21

5.4 Symmetric and public key systems differences 22

6 ONE-TIME PAD 23

7 PSEUDO RANDOM NUMBER GENERATOR 25

8 AIM OF THE RESEARCH 27

9 TOOLS USED IN RESEARCH 28

9.1 CHI2 TEST 28

9.2 ENCRYPTION SOFTWARE 29

10 RESULTS 30

11 CONCLUSIONS 35

REFERENCES 37

APPENDICES 39

1 INTRODUCTION

Despite the fact that question of security of information was crucial for people from ancient times, cryptology as a science experienced rapid growth and development only in recently. Exposure of electronic means of communication had a significant impact on this growth. In addition to the task of keeping information secure, Internet and computers lead to new fields being studied in cryptology – authentication, integrity and nonrepudiation. So, not only secure transfer of message is studied nowadays but also the possibility of creating a digital signatures or long-term storage of information.

Cryptology, being young as a science but old as an art, is a very broad field with many branches and topics to discover exist. It is sometimes said that simplest solution tends to be the most efficient or secure. To some extent, this can be true in cryptography as well. If studying principles of work of famous ENIGMA encryption machine, it is seen that mathematically it was based on the system of permutations which is not as sophisticated and systems used nowadays but it was widely and successfully used for a long time.

From first sight it might seem that cryptology is about keeping everything in secret – original text, way of encryption, algorithm, key and any other details which is not correct. As Kerchhoff’s principle, formulated back in 1883 century in his “La cryptographie militaire”, states, system should not require secrecy and even if it falls in arms of an enemy, it still should stay secure. The meaning of it I s that it is the key that should definitely not be public, everything else unrevealed should have no impact to secrecy.

There is now “best” or “worst” type of encryption algorithm or code. Stated algorithms are more proven to be more secure than others. There are algorithms that are to be used in one situation and are unsuitable for others. Main types of encryption today are symmetric, public-key and hybrid ones. Widespread and trusted algorithms are broken, new ones are defined or created. It is important to understand the way cryptology had during centuries to the state it is now. What scientists tried, for what purposes and what were the results of their researches. On one hand, even discussion of one simple and base algorithm can be very deep and long. On the other, to understand the necessity of the research and its importance, at least basics of most famous and widespread cipher algorithms should be introduced.

As long-time storage of information does not require safe transfer of information, but means large amount of data to be kept, one-time-pad, symmetric type of encryption is more efficient to the research than asymmetric. It is important to understand the difference, advantages and disadvantages of both of them. Though asymmetric types are introduced briefly and for general understanding of the problem, symmetric standards – AES and DES – are described in more details.

The main aim of the thesis was to perform a number (up to 15) of encryption rounds using one-time pad algorithm for different types of plaintext. For text, .doc files were chosen with two types of text contained, simple text (this research paper was used as sample) and a file containing a repeated symbol only. Also, an image file .jpg, a sound file .mp3 and a video file .mp4 were tested. Additional simple test was made for the files of .pdf format. For easiness of understanding of results, also introduction into χ^2 analysis and distribution of bytes are included in the research. As a result of the research, collected data is provided as well as analysis of required encryption rounds for each type of proceeded files.

2 TERMINOLOGY

Though cryptographic terminology is widely-known and so easy for understanding, to avoid misunderstanding and clarify the meanings, short introduction and definitions are provided. All terms are defined according to book by computer science and encryption specialist Bruce Schneier, as in his book “Applied Cryptography, Second Edition: Protocols, Algorthms, and Source Code in C” (Schneier, 1996) and Robert Churchhouse’s “Codes and Ciphers” (Churchhouse, 2002).

Original message, which can be any kind of unencrypted file, is called a plaintext. Sometimes it is also referred as a cleartext. For PC, any stream of bits can be a plaintext, including a text file, image bitmap, and a stream of digital sound or video image. Intention of the plaintext can be both transmission and storage. (Churchhouse, 2002.)

A single symbol of the plaintext is called a monograph. Respectively, pair of adjacent symbols is a digraph; trigraph is a set of three symbols. Unspecified number of adjacent symbols is a polygraph. (Schneier, 1996.)

Process which transforms the message to hide its substance and make it unintelligible is called encryption or encipherment. The reverse process, which is intended to recover original plaintext, is called decryption or decipherment respectively. (Schneier, 1996.)

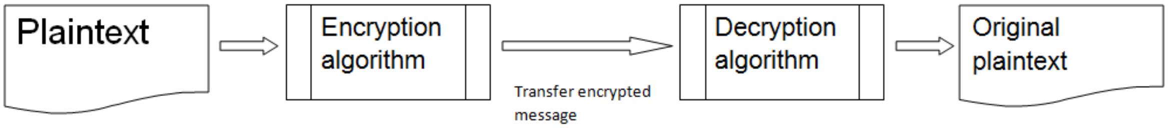
A cipher system or encryption system is any system that is applied for encryption of the original message. Mathematical function, used for encryption and decryption is called a cryptographic algorithm or a cipher. (Churchhouse, 2002.)

Security of the cipher system can be based on keeping the way of algorithm work in secret. This kind of algorithm is called restricted. Currently such algorithms are not widespread due to low efficiency and flexibility. Additionally, restricted algorithms have no possibility of quality control or standardization. Each group of users require their own unique algorithm, and it has to be changed every time the user leaves the group. If none of the members of the group is a cryptographer, level of efficiency or security can be proved. (Churchhouse, 2002.)

Therefore, modern cryptography is dealing with the problem using a key to encrypt and decrypt message. Key allows higher level of flexibility and provides possibility for standardization. If the key used for both encryption and decryption is the same, such algorithm is called symmetric. In special algorithms, different keys are used for these operations; they are named public-key algorithms. (Schneier, 1996.)

The message, resulted in process of encryption is ciphertext. The science and art dealing with keeping messages secure and protected, also the study of design, use, strengths and weaknesses of cipher systems is cryptography. On the other hand, science and art of solving cipher system and algorithms of encryption is cryptanalysis. (Schneier, 1996.)

Cryptographic systems in general have the same as structure, as presented on Graph1. Firstly, plaintext is ciphered using defined encryption algorithm. Then encrypted message is transferred using any, including unencrypted, channel. Finally, receiver decrypts the message using stated decryption algorithm to get original plaintext. (Schneier, 1996)



GRAPH 1. Cryptographic system concept (Schneier, 1996.)

3 DEVELOPMENT OF CRYPTOGRAPHY

Cryptography has a long history going back to the time of first uses of written communication. From ancient times, some senders wanted to hide the message from non-addressee. One of ways, used in ancient Greece, is described in "Codes in Ciphers" by Robert Churchhouse (2002): sender shaved slave's head and scratched the message on it. After the hair had grown, the slave was send to receiver, who shaved his head again and so could read the message. Clearly, this message is inefficient as requires extremely long time to deliver one message and insecure as anyone aware of the method could read the text just shaving the slave's head as well. Also, this way never possible to provide authentication for the message, this is essential for good cipher system. This is one of the possible examples of restricted algorithms showing its weaknesses and vulnerabilities.

3.1. Cryptography as an art

Most famous of ancient cipher systems is named after Julius Caesar, which he used for securing his communications, military and political. It is said, that he encrypted plaintext by replacing each letter by the one, standing three places further down the alphabet. Today, not only move by three places, but any cipher alphabet that is consisting standard sequence, is called a Caesar alphabet. This can be an example of a simple substitution cipher. Though yet of not high security and efficiency it is sufficiently more viable then Greek's way. (Churchhouse, 2004, 2; Kahn, 1996, 84.)

Later on, more sophisticated and developed algorithms were developed, involving specific mathematical transformations. Caesar's cipher got its development and evolution. Strengthened simple substitution by not one shift but several, according to algorithm resulted in so-called Vigenere ciphers.

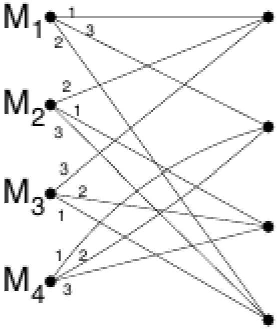
(Churchhouse, 2004, 28.) Famous machine doing encryption and decipherment developed in early 1920s by Arthur Schrebius, ENIGMA, also has in its basis simple substitution cipher with number of shifts.(Kahn, D., 1996, 219.)

First to start developing and exploring cryptology were Arabs. In a 14-volume encyclopedia Subh al-a 'sha, completed in 1412, cryptologic section has two parts – one about symbolic actions and allusions; second one dealing with invisible inks and cryptology. The encyclopedia also gave seven systems of ciphers, including both transposition and substitution ones. (Kahn, D., 1996, 88.)

3.2. Cryptography turn into science

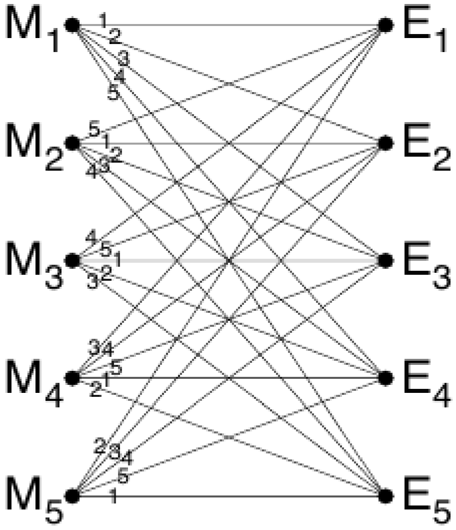
So, rather than a science, for long time cryptography was considered to be an art, required mostly in the military and diplomacy fields. First step to turning it into science is considered to be development of mathematical theory of communication and a related communication theory of secrecy systems by Claude E. Shannon during World War II. These two theories are usually referred to as starting point of information theory. (Oppliger, R. 2005, 14-15.)

In his work, Shannon defines fundamental terms of cryptography theory. He introduces line diagram as on graph 2 below, showing example of system. Possible plaintext messages are shown as dots on the left side, marked *M.* Possible cryptograms are presented as points at the right. Line connects plaintext and its cryptogram. (Shannon, 1949, 656-715.)



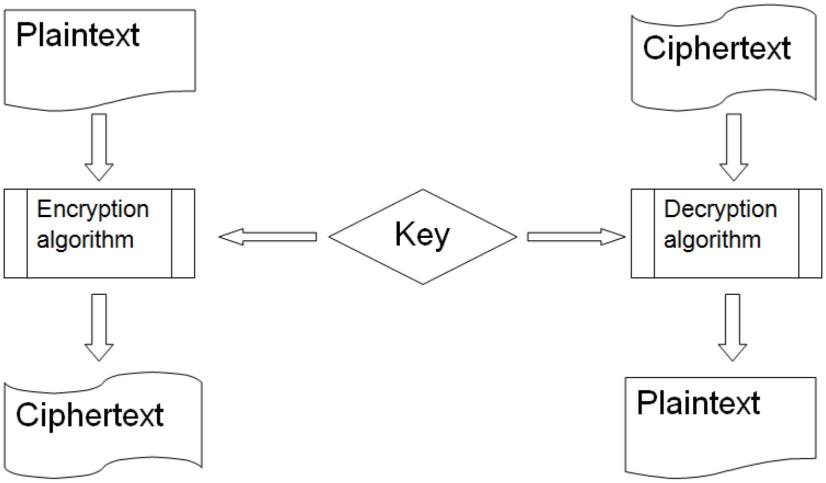
GRAPH 2. Secrecy system in Shannon's theory(Shannon, 1949, 656-715).

The same way, Shannon defines perfect system, as shown in graph 3. He states, that system is perfect when total probability of all keys encrypting plaintext M_i into cryptogram E is equal to all keys for plaintext M_i. The statement should be true for all M_i, M_j and E. (Shannon, 1949, 656-715.)



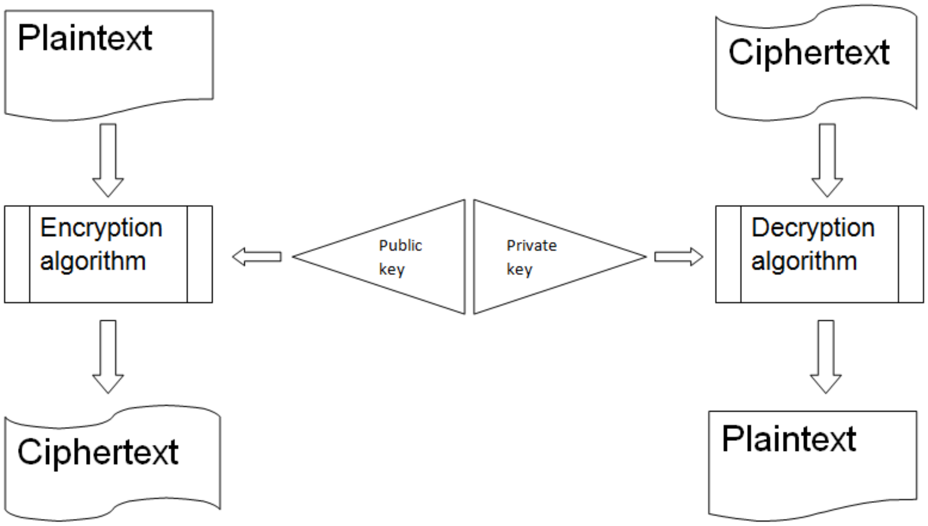
GRAPH 3.Perfect system in Shannon's theory (Shannon, 1949, 656-715).

Second step of big influence on development of cryptography as science was idea of public key. Before it was developed and introduced by Diffie and Hellman in 1970’s at Stanford University a symmetric encryption, as shown on Graph 4 was used. Diffie and Hellman paper introduced the idea of public key encryption, as on Graph 5 and also proofed its feasibility by proposing a new key agreement protocol. Their idea for public key was the notion that key could come in pairs and, additionally to this, it could be infeasible to generate one key from the other using known algorithm. Though, it should be impossible to generate the key the back way. (Churchhouse, 2004, 161-162; Dorichenko & Yashenko, 1994, 35.)



GRAPH 4. Symmetric encryption (Schneier, 1996).

Since Diffie and Hellman introduced their idea, a great number of public key systems were developed. On Graph 5 concept of public key system is introduced. Most of them have been broken and so are no longer in use anymore. Some of algorithms where considered impractical, for example if too large key or if the ciphertext resulted is sufficiently larger that original plaintext. Some of the systems, as RSA, ElGamal and Rabin are in use. (Churchhouse, 2004, 161-162, Schneier, 1996.)



GRAPH 5. Public-key encryption (Schneier, 1996).

3.3. New tasks for cryptography

With rapid development and widespread of computer sciences and Internet communications, cryptography and cryptanalysis also experienced wide deployment and evolution. Cryptanalysis made a great step forward with the development of computer systems, as processing of data is automated to a higher percent. (Moldovyan & Moldovyan, 2006, 12-19; Schneier, 1996, 10.) Though simple-substitution systems based on Caesar's cipher are not of much spread with modern level of computer processing possibilities, key-using algorithms are commonly and widely used in today's cryptography, both symmetric systems and public-key ones. (Schneier, 1996.)

Though automation and computerization allowed solving a number of issues in cryptology, several new questions were defined. Currently cryptography does not only deal with providing information confidentiality. It is not enough anymore to ensure the information being incomprehensible for third parties. Today, it should also provide possibility for the authentication of the information and message source; integrity – to verify that message has not been modified in transit and also nonrepudiation. So computer information technologies require cryptography also for such tasks as creating digital signature systems, different types of voting systems, user authentication protocols, coin-tossing protocols, long-term storage of the data. (Moldovyan & Moldovyan, 2006, 12-19; Schneier, 1996, 10; Dent & Mitchell, 2004, 22-25.)

4 PUBLIC KEY ALGORITHMS

Public key algorithms are sometimes also referred to as asymmetric algorithms. They are designed so that the key used for enciphering the plaintext is different from the key used for decryption. Additionally, decryption key should not be possible to be found out at a reasonable time and effort from the encryption key. Due to this condition, encryption key can be made public and so available for wide use. Usually, encryption key is called public key and decryption key is referred as private or secret key. (Schneier, 1996, 14.)

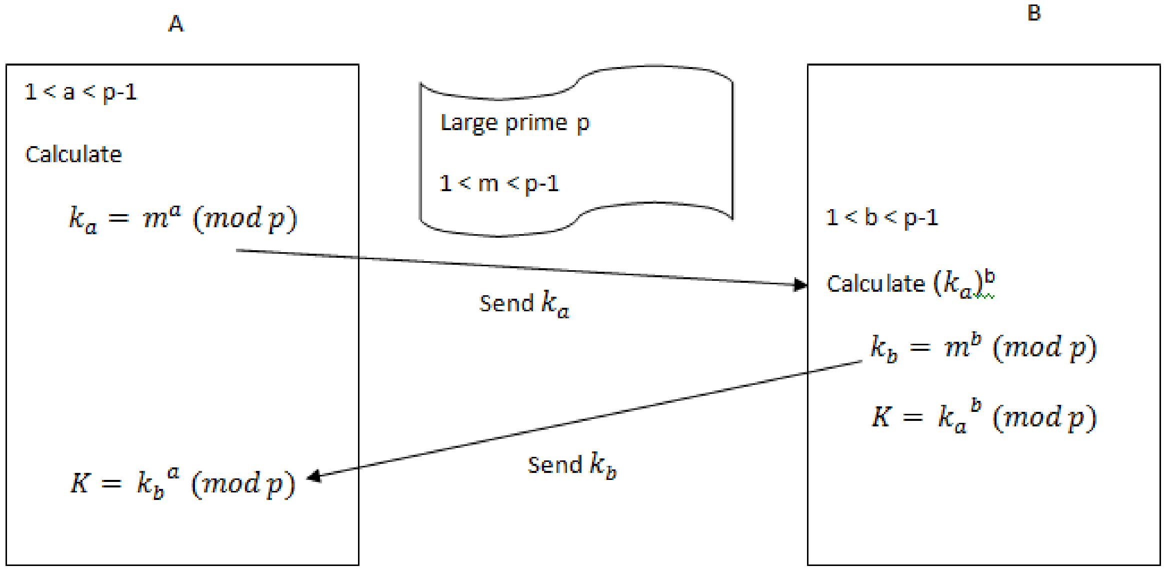
4.1. Diffie-Hellman key exchange protocol

As mentioned above, first introduction of idea of public key dates back to 1970s, when Diffie-Hellman system was introduced. Named key protocol requires users exchanging message without prior exchange of secret keys. Both parties, exchanging information, receive a secret number within set range and then according to mathematical algorithm, new value is calculated. Resulted value is used as common key which both parties can use for encryption though not knowing each other’s secret key. (Moldovyan & Moldovyan, 2006, 37; Silverman, Pipher & Hoffstein, 2008, 59-65.)

Parties *A* and *B* agree on usage of two integers *p* (which should be a large prime) and *m* (value between 1 and (p-1)). Both values can be public. Then A and B choose a secret value *a* and *b* respectively in the range between 1 and (p-1). And neither of chosen should have factor in common with (p-1). Finally, A computes the number by formula (1):

$$k_a = m^a \, (mod\, p) \tag{1}$$

The resulted value is send to B, who calculates the value $(k_a)^b$. Than the same operation is done by B, so $(k_a)^b = (k_b)^a = m^{ab}(mod\, p) = K$ is the public key which any party can use though not knowing each other's secret key. This encryption is summarized in Graph 6. (Silverman et. All, 2008, 59-65, Churchhouse Robert, 2002, 166-169.)



GRAPH 6. Diffie-Hellman system (Churchhouse Robert, 2002, 166-169).

Restriction for using this protocol is usage of prime number that should be very large, which makes it non-trivial task. Usage of prime number in algorithm makes break of cipher difficult and non-efficient so the system would be protected to satisfactory level. In general, as the algorithm includes discrete logarithm, it can be considered impossible to break the system is chosen prime number value is larger than 10^{200}. (Churchhouse Robert, 2002, 166-169.)

4.2. RSA encryption system

One more public key algorithm that does use prime numbers and is continuosly in use nowadays is RSA encryption system. RSA, named after R.Rivest, A. Shamir and L. Adelman, who developed and introduced the method in 1978, is public-key system that is also based on substitution system. This system encryption cannot be carried out by hand but computer with facilities for multi-length arithmetic can carry it out. RSA algorithm uses Fermat-Euler Theorem for its encryption and decryption process, which states that for each relatively prime number M and n, where M < n, equation $M^{\varphi(n)} = 1(mod\, n)$ is true. (Churchhouse Robert, 2002, 174-178.)

For RSA system, *n* should be chosen as a product of two large prime numbers *p* and *q*. Also, an integer *e (*encipherment key*)* should be chosen having no factor in common with (p-1)(q-1) or with *n*. For decipherment key *d,* it should satisfy condition that $\mathrm{ed} = 1(\mathrm{mod}(\mathrm{p}-1)(\mathrm{q}-1))$. Basis of security of RSA system is that if *p* and *q* are not known, *d* cannot be found. In a typical application of RSA system and digital signature, modulus used is at least of order 10^{100}, the encipherment and decipherment keys are around 10^{50}. So brute force calculations are out of possibility and there is currently no practical feasible way to solve the problem a modulus of greater than 512 bits long has been found. For special cases of prime numbers used to find *n,* problem decreases drastically, so some tests are always performed for implementing the system. (Churchhouse Robert, 2002, 174-178, Lek & Rajapakse, 2012, 159-185.)

Very important thing for RSA system is that it also does allow to perform verification procedure. *S* - signature corresponding to the message M – is raised to *e* integer power modulo n: $M' = S^{e}(mod\, n)$. Then if $M' = M$, then the message is recognized to be signed by user who previously provided the *e* public key. For RSA cryptosystem, signature generation procedure is the same as decryption procedure and, vice versa, signature verification is performed by the same scheme as the encryption procedure. (Moldovyan & Moldovyan, 2006, 42-44; Lek & Rajapakse, 2012, 159-185.)

4.3. Rabin's scheme

Though RSA system is widely spread and known to be secure, it is theoretically possible to break it. Since introduction of the system a public key cryptosystem that would be complex enough to be compatibly equivalent to solving the Integer factoring problem – which is not requirement in case of RSA, was searched. First person proposed such system was Michael O. Rabin in 1979. System he introduced is based on finding square root modulo a composite number. For the algorithm, two primes, *p* and *q* are required. For encrypting the message, formula $C = M^2 \, mod \, n$ is used. Since the receiver knows *p* and *q*, he can compute possible decryption using the Chinese remainder theorem. (Oppliger, 2005, 347-352.)

Restriction that Rabin system has is that as a result, four possible solutions are defined. If the message sent is a text of known language then it should be easy to determine the correct one. On the other hand, if plaintext is a stream of undefined random bits or bytes, then there is no way to choose the correct value. (Oppliger, 2005, 347-352.)

In case of usage of digital signatures, it is never the whole message that is used for verification. As the straightforward use of the schemes require splitting the message into short blocks and so creating a number of signed blocks in the same document. To solve this problem only a small digital image is used for obtaining a signature. If such image, called hash, is confirmed to be genuine, the whole document is considered signed. (Oppliger, 2005, 347-352; Dent & Mitchell, 2004, 93-95.)

Though Rabin system does solve problem of RSA of being as secure as primer mathematical problem, this case factoring, it is absolutely insecure against a chosen-ciphertext attack. The way to solve this issue is using hashing function. Also Rabin himself suggested one more way for defending from chosen-ciphertext attack: appending a random string to a plaintext before using hash function and creating a signature. (Oppliger, 2005, 347-352.)

4.4. The El Gamal Digital Signature

El Gamal digital signature system is named after its inventor, Tahir El Gamal and based on the scheme for generating public and private key introduced in Diffie-Hellman method. Important difference, introduced by El Gamal is that unlike Diffie-Hellman method, it can be used not only for encryption but also for creating and verifying digital signatures. (Ryabko & Fionov, 2004, 33; Delfs & Knebl, 2007, 70.)

Algorithm requires a large prime number *p*, and he primitive element *α* modulo *p*. Important condition is that (p-1) factoring includes at least one large prime factor. Same as in two algorithms mentioned above, the secret key *a* is chosen and public key is generated from it. Formula for calculating public key is $k_a = \alpha^a$. A's signature is the pair of numbers *r* and *s* that meet the requirement $(\alpha^M) = {k_a}^r r^s$. (Delfs & Knebl, 2007, 70-72; Moldovyan & Moldovyan, 2006, 45-47.)

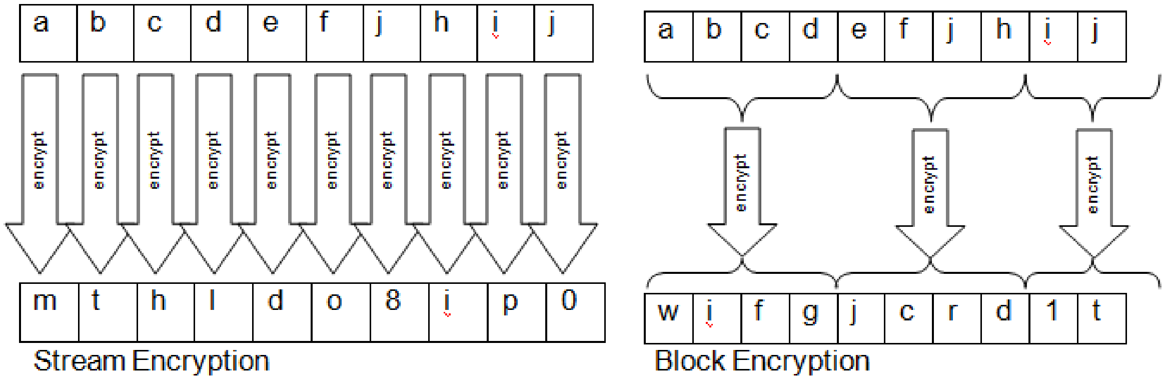
Finding the pair of numbers r and s without knowing the private key k_a is a complex task and only the owner of secret key can generate the correct signature. Computing the private key from the public one would require solving a complex discrete logarithm problem, as it is introduced in Diffie-Hellman method. Modulus that is to be used to make sure that system is secure should be at least 1024 bits long. Furthermore, primes with special properties, as those for which discrete logarithms are known to be fast and efficiently found out, should be avoided. (Delfs & Knebl, 2007, 70-72; Moldovyan & Moldovyan, 2006, 45-47.)

One more important peculiarity of El Gamal algorithm that for every message secret key k_a should be different and values used during generation should be destroyed. The reason for it is that if the k_a value used previously is recovered, then the private key can also be calculated. (Silverman et all, 2008, 68-72; Moldovyan & Moldovyan, 2006, 45-47.)

5 SYMMETRIC CIPHERS

On the other side of encryption types from public key systems stand symmetric ciphers. It is also referred as secret-key system or conventional algorithms. Unlike public-key systems, key used for encryption can be calculated from decryption key and vice versa. Moreover, for most algorithms, both keys are just the same. So, these algorithms require both receiver to agree upon decryption key before the communication started. So, safety of the communication lays completely on secrecy of the key, not on the details of the algorithm. (Schneier, 1996, 247-248.)

All symmetric algorithms can be divided into two large categories. Those that operate on monograph of a plaintext at a time are called stream algorithms or stream ciphers. Those that operate groups of bits (or bytes) are called block algorithms or block ciphers. Both are presented on Graph 7. Currently typical length of a block processed is 128 bits. (Schneier, 1996, 247-248.)



GRAPH 7. Block and stream encryption. (Schneier, 1996, 247-248.)

As well as many public key algorithms were broken and are not used anymore, several of the symmetric encryption systems are not used in practice for different reasons, for example if they are theoretically relevant but could only be implemented and safely used in small and typically closed communities.(Oppliger, 2005, 26-28.)

Today there are two most known symmetric encryption systems: DES (Data Encryption Standard) and AES (Advanced encryption standard), both of them are block ciphers. Stream ciphers are usually based on pseudorandom cipher digit stream and so should be at least of the same length as plaintext and so of low convenience, feasibility and use. (Oppliger, 2005, 26-28.)

5.1. Data Encryption Standard

The Data Encryption Standard, which was originally defined in FIPS46, 1977 was first standard and previously mostly widely used symmetric-key encryption algorithm. It was used as basis for secure and authentic communications not only by militaries and governments but also banks and commerce applications. (Delfs & Knebl, 2007, 16.)

As already mentioned, until till the second half of 20th century, cryptography was not a science and no common standards existed. Several small companies were specializing on producing and selling cryptographic equipment but primarily to overseas governments. It was usually all different and not available for interoperation. (Silverman et al, 2008, 485-487.)

US National Bureau of Standards (NBS) invited interested parties to submit proposals for data encryption standard in 1973. A proposal made IBM engineers in 1975, was accepted and became “The data Encryption Standard”. It was the first time that US National Security Agency (NSA) evaluated algorithm was made available for public. It was mentioned later, off the record, that NSA did not realize that NBS going to make it public and that anyone would be able to write a software otherwise they would never agree to it. (Churchhouse, 2004, 183-185; Schneier, 1996, 375-378.)

After acceptance, the standard went a long way of discussion and was only accepted as a private-sector standard in 1981 under name Data Encryption Algorithm (DEA). Afterwards, also a standard for network encryption that uses

DES was published and a standard for DEA mode of operation. (Churchhouse, 2004, 183-185; Schneier, 1996, 375-378.)

As a part of the standard, implementations of DES are validated. Until 1994, only hardware and firmware implementations were validated; software implementations were prohibited. (Churchhouse, 2004, 183-185; Schneier, 1996, 375-378.)

DES standard was withdrawn in 2005 and completely replaced by published in 2001 Advanced Encryption Standard. Later, in 2006 and 2008 the algorithm was broken by brute force method in 9 and 1 day respectively. (SciEngines, 2009.)

5.1.1. Details of DES encryption

DES algorithm is originally designed to encipher blocks of 64 bit data under control of 64-bit Key. This size was commonly used for over 25 years. However, this value was chosen as a compromise between security, which requires the greatest possible value and implementation convenience which becomes more complex with raise of bits. Later with development of computers and microelectronics, 128 bit input block became the standard. (Churchhouse, 2004, 183-185; Schneier, 1996, 378-388.)

Parties who plan to communicate using DES agree on the secret key (*K*). All other information about system is public. Finally, for secret key, 64-bits key is built of user-chosen seven 8-bit characters for which DES later adjoins a further 8 parity check bits to get required 64-bit secret key. As it is a symmetric encryption algorithm, the same key is used for encryption and decryption; they are only used in reverse order. (Churchhouse, 2004, 183-185; Schneier, 1996, 378-388.)

5.2. Advanced Encryption Standard

With development of computer technology and Internet, it became clear that DES is no longer providing the necessary level of security. Key and block size have become too small to resist attempts to break it. So, after more than 20 years, a search for successor, advanced encryption standard, was started by NITS. (Delfs & Knebl, 2007, 19.)

5.2.1. History of development of AES

In January 1997 NIST started an open selection for a new standard for encryption. Several requirements were defined for the standard. Firstly, algorithm should be of symmetrical block type, supporting block size of at least 128 bits and of key sizes of 128, 192 and 256 bits. Then, algorithm should be published and open for use in any kind of products which means it could not be patented. Finally, it should be designed both for hardware and software realization. (Delfs & Knebl, 2007, 19-24; Zenzin & Ivanov, 2002, 47-75.)

Selection process was divided into three rounds. Out of 21 proposed for first, only 15 were selected. Then chosen candidates were evaluated by public discussion. After that, in the second round five were left: MARS from IBM, RC6 from RSA, Rijndael by Rijmen and Daemen, Serpet by Anderson, Biham and Knudsen and Twofish from Counterpane. To decide on the final solution, three international conferences were held and in October 2000 NIST decided on Rijndael to be the AES. (Delfs & Knebl, 2007, 19-24; Zenzin & Ivanov, 2002, 47-75.)

The only difference that was introduced for the originally proposed algorithm was that AES fixed the block lengths to as requirements were set. Rijndael itself supports additional key lengths of 160 and 224 bits and block sizes with same lengths. (Delfs & Knebl, 2007, 19-24; Zenzin & Ivanov, 2002, 47-75.)

5.2.2. Details of AES encryption

Similarly to DES, AES encrypts and decrypts plaintext by repeating the same basic operation several times. Depending on the type of the key, it can be 10, 12 or 14 rounds, which are called iterations. Basic operations are using 128 bit blocks which are broken into 16 bytes. Then, as each byte consists of 8 bits, each of it is treated as a separate element. AES is a byte-oriented algorithm, where input and output are considered as one-dimensional arrays each of 8-bit-bytes. Encryption by Rijndael algorithm starts from initial key round, then the round function is applied defined number of times and followed by final round with slightly modified round function. (Delfs & Knebl, 2007, 19-24; Zenzin & Ivanov, 2002, 47-75.)

AES and Rijndael use new architecture, called s-box (square matrix serving for multiplicative inverse for a given number). They have byte-oriented structure, perfect for 8-bit processing and all iterations are operations in finite fields which allows effective implementation on different platforms. (Delfs & Knebl, 2007, 19-24; Zenzin & Ivanov, 2002, 47-75.)

5.3. Stream type encryption

As already mentioned, unlike block type, stream type encryption algorithm does not make enciphering of the file by block of certain size but byte-by-byte. Though, the main principal difference of stream encryption is that this algorithm for each portion of plaintext to encrypt new key of the same size is used. For block cipher, for each portion same key is used. In other worlds, in block encryption, result is dependent on the position of the block in the text. On the other hand, stream encryption does not have this dependency. Also, stream ciphers allow much faster time of encryption than block algorithm. As speed of output is the same or comparable to output, stream encryption is widely used for fast transfer of significant amount of data such as digital video or sound. (Zenzin & Ivanov, 2002, 23-26.)

5.3.1. Synchronous stream encryption

Stream ciphers are divided into two types – synchronous stream encryption and self-synchronizing one. Keystream for synchronous stream cipher is generated independently from the plaintext stream. A keystream generator spits out keystream bits one by one. This leads to requirement for two key generators – for encryption and decryption – to be synchronized. If one of them skips a cycle or if a ciphertext bit gets lost, then after the error all the ciphertext gets decrypted incorrectly. In such a case, both sides have to re-synchronize their keystream generators before they can proceed. This is not as easy solution as it seems as for security of the encryption, no part of the keystream should be repeated so resetting to the earlier state before error is not possible. (Schneier, 1996, 291-292.)

On the other hand, if a bit is garbled during transmission in synchronous cipher, then only that one bit will be decrypted incorrectly. All other preceding and subsequent bits will be left unaffected. So errors do not get multiplicities; as many bits were affected during encryption, so would be for decryption. (Schneier, 1996, 291-292.)

In case of synchronous encryption, keystream length should be at least same as plaintext. As encryption is implemented in finite-state machines, the sequence will be eventually repeated. Keystream generators for such encryption are called periodic. The only encryption algorithm that uses non-periodic keystream generator is one-time pad. (Schneier, 1996, 291-292.)

For self-synchronizing stream encryption, input sequence for encryption is defined in accordance with *N* preceding elements. Complexity is in output stream which is calculated from internal state and generating a keystream. This mode has advantage of automatically re-synchronizing after *n* bits so even if an error appeared, after defined length it is eliminated. But this also means existence of error propagation. If a bit gets garbled, whole amount of *n* subsequent bits will be decrypted incorrectly. (Schneier, 1996, 291-292.)

5.4. Symmetric and public key systems differences

Both symmetric and public key systems are used in contemporary cryptography though they have own advantages and disadvantages. As symmetrical algorithm systems do not require computing of decryption key they are faster comparing to public-key systems. Also, key length for required for a system to be considered secure is different for public-key and secret-key algorithms. According to Bruce Schneier, equivalent to 56-bits long secret key would be a 384-bit long public key. Resistance for brute force attack of 128-bit symmetric key would be same with 2304-bit public key. (Schneier, 1996, 247-248.)

Though facts mentioned above do not definitely mean that symmetric algorithm is more secure or advanced. One of the weakest parts of symmetric encryption is still the need to have a secure channel to deliver and keep the secret key for both encryption and decryption. Two algorithms are just different ways to solve different problems. Public-key cryptography is suitable for tasks that symmetric cryptography is not. It is more suitable for key management, digital signatures and different types of protocol. For encryption itself and long-term storage of information, symmetric cryptography is a better solution. (Schneier, 1996, 247-248.)

6 ONE-TIME PAD

In 1917 an encryption scheme that in theory is absolutely secure was invented by Major Joseph Mauborgne and AT&T's Gilbert Vernam. After his name, the algorithm is sometimes referred to as Vernam's cipher. It is a special case of threshold scheme and was called one-time pad. By definition, one-time pad uses a large not repeating keystream of truly random symbols. Each key symbol should be used exactly once. As symmetrical type of encryption, algorithm requires both sender and receiver having exact keystream for encryption and decryption. (Schneier, 1996, 39-42; Dorichenko & Yashenko, 1994, 48-49; Ryabko & Fionov, 2004, 120.)

To encrypt a plaintext *p*, key stream *k* is required. Encryption and decryption are given by bitwise XOR operation of *p* with *k*. As a result, given ciphertext without secury key is equally likely to correspond to any possible plaintext of the equal size. So algorithm does not give any information about plaintext. (Schneier, 1996, 39-42; Dorichenko & Yashenko, 1994, 48-49; Ryabko & Fionov, 2004, 120.)

For the ensuring of absolute security of one-time pad algorithm requirements are than key is truly random; length of the key and plaintext are equal; and that the key is used only once and destroyed after usage. These requirements make use of the algorithm extremely expensive and not feasible. One of the disadvantages is equality of key length and final length of transferred message. Also, both parties are required to have the same key. This means that for providing it, an absolutely secure way for transferring the key should exist. This leads to situation that though in theory absolutely secure algorithm exists, it is almost not used in practice. It is known that Vernam's cipher was used on governmental hot line between Moscow and Washington and the keys were transported by trusted courier. (Schneier, 1996, 39-42; Dorichenko & Yashenko, 1994, 48-49; Ryabko & Fionov, 2004, 120.)

In addition, Vernam one-time pad does ensure confidentiality but cannot secure messages from modifications. If any bit of ciphered text is changed and decrypted result makes sense, the receiver will not notice the change. Nevertheless, stream ciphers are used nowadays but with certain changes. One of them is instead of generation of truly random sequence for key pseudo random number generators are used. (Schneier, 1996, 39-42; Dorichenko & Yashenko, 1994, 48-49; Ryabko & Fionov, 2004, 120, Dent & Mitchell, 2004, 53-54.)

7 PSEUDO RANDOM NUMBER GENERATOR

The definition used in cryptography for random sequence is that binary sequence is considered to be random if the probability of the subsequent digit to be 0 is 0.5. For decimal digits, probability would be 0.1. For sequence of letters of English alphabet probability of particular letter would be 1/26. So, the probability of particular symbol should be equal to probability of any other symbol n sequence, no matter how many digits preceded. It is clear that truly random sequence cannot be produced by any mathematical formula, as knowledge of it and initial values would enable prediction of the next value with some certainty. (Churchhouse, 2004, 94-101.)

Naturally there are several ways to produce sequence of random values, starting from simple coin spin or throwing a dice (which on practice because of some regularity of the spin and the material can have some regularity) to complex ways as usage of noise amplifiers, which convert the noise into a signal to switch gate off and on and so be interpreted as sequence of 0 and 1. Yet any way of producing a keystream from any generator is to be tested and at some point it is a philosophical debate, where any of the techniques produce real number or not. (Churchhouse, 2004, 94-101; Schneier, 1996, 503-505.)

Pseudo-random sequences are based on different mathematical formulas as for example Fibonacci sequence, the mid-square method, linear congruential generators. As for this research main purpose is not ensuring absolute security of the plaintext but analysis of distribution of bytes after the encryption, pseudo-random number generator provided by means of C++ language is sufficient. (Churchhouse, 2004, 94-101; Schneier, 1996, 503-505.)

Currently, NIST maintains website which contains a list of approved pseudo-random number generators and also statistical tests and requirements for generators. FIPS 186-2, ANSI X9.31, ANSI X9.62-1998 and SP 800-90A are listed as approved. There are no “true” random number generators currently approved. For approval of the PRNG several tests can be applied, though there is no specific set of tests held for every pseudo random number generator. (NIST, 2015.)

Random number generators using an algorithm are known as Deterministic Random Bit Generators. Such generators use a pre-specified algorithm to produce a sequence of bits out of initial value. Value itself is determined by a so-called seed, string of bits with entropy sufficient to support security, defined from entropy input. Seed needs to be kept secret and, on condition of well-designed algorithm, output bits will be unpredictable, providing security strength of the produced sequence. (Barker & Kelsey, 2012.)

One solution for DRBG mechanism is non-invertible or one-way hash function. Security strength in this case is equal to security of the hash function used. NIST provides approved algorithms, seed length and period of generating bits in relevant document. The other solution is based on block ciphers. In this case, any of already approved block cipher algorithm can be used. Security strength varies according to algorithm and key length. (Barker & Kelsey, 2012.)

Finally, DRBG mechanisms can be based on number theoretic problems. In this case, security strength of the generator is same as security of the curve. As different mathematical problems can be used for such a mechanism, security level differs and NIST provides some of it. (Barker & Kelsey, 2012.)

8 AIM OF THE RESEARCH

Main task for the research was to check the entropy of the distribution of bytes for different types of files depending on times the file was encrypted. It was important to see the minimum rounds of encryption needed to have the bytes in files distributed uniformly. Also, it was important to find out whether uniformity rises with each new encryption round or if there is a certain limit when further encryption becomes meaningless. To test entropy, χ^2 test was chosen. The limit for the rounds was set to 15, so if value of χ^2 would keep decreasing up to the last round, a balance for speed and performance value should be chosen. Finally, it was important to find out how do encryption affect different types of files. If possible, suitable solution for any type of file number of rounds should be defined.

9 TOOLS USED IN RESEARCH

For the research, several tools were used. First, for statistical analysis, chi2 goodness of fit test was decided as method to check uniformity. Also, encryption software, based on one-time pad was developed. This software only encrypted byte information and then analysis was performed based on formulas.

9.1. CHI2 TEST

Chi-squared test of goodness of fit was introduced by Karl Pearson in his paper in 1900. Today his method is essential for statistics analysis. Important introduction in the research made by Pearson was accepting the null hypothesis and comparing the results not to standard deviation but to most probable one. (Plackett, 1983.)

Pearson's chi square test is used for verification if the data is specifically distributed. If the data follows a specified distribution, then H_0 hypothesis is confirmed. H_a hypothesis is defined if data does not follow it. Though the test is not suitable for small samples, it is commonly used for test of pseudo random number generators. In formula used for calculation of χ^2, O_i is observed value, E_i expected value and n number of values:

$$x^2 = \sum_{i=1}^{n} \frac{(O_i - E_i)^2}{E_i}$$

(Filliben & Heckert, 2012; Plackett, 1983.)

9.2. Encryption software

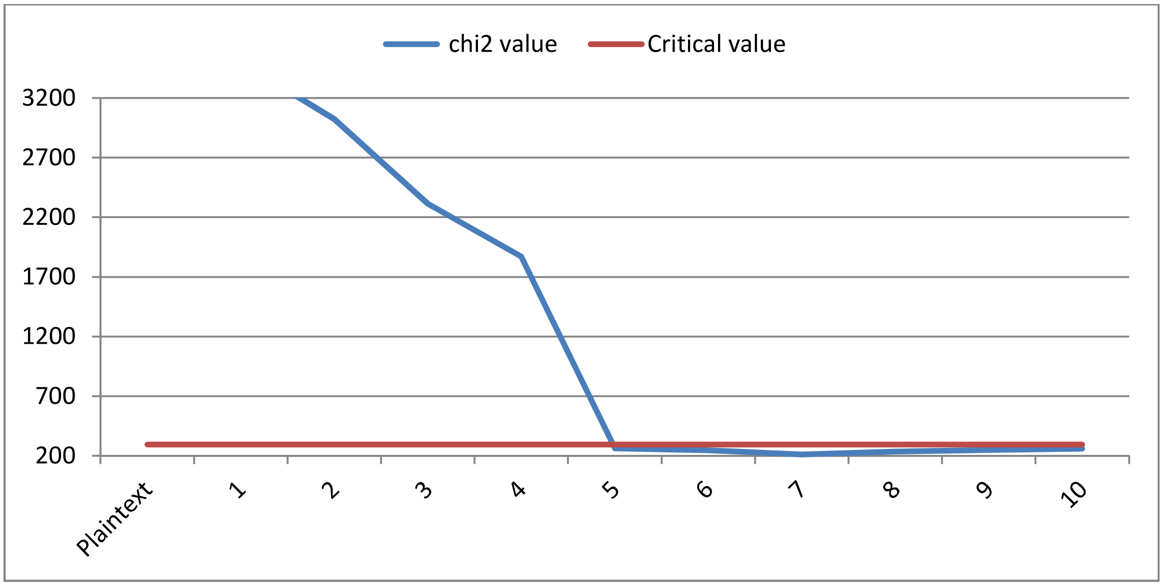
To encrypt files simple program was used. Stream of bytes is read from .txt file and encrypts it byte-by-byte using one-time pad algorithm. As the software can only read the stream from .txt file, additional software to read any type of file as stream bytes was used, so freeware Hex and Disc editor, HxD. After the file was presented as stream of bytes in HxD, the stream was put into test.txt file and software for encryption was used. As a result, two .txt files were generated in the same folder: encrypted.txt and unencrypted.txt. Afterwards, the files were read again as stream of bits and xi-squared value was calculated using formulas. Also, for easiness of calculations and representing, some Excel functions were used.

To encrypt file for the next round, file encypted.txt was renamed into test.txt and the software was run again. Each next round was run for previously encrypted stream of bytes. As software is intended only for encryption, it does not have user interface. Only an executable file exists, making console panel opened and closed right after the encryption is finished.

10 RESULTS

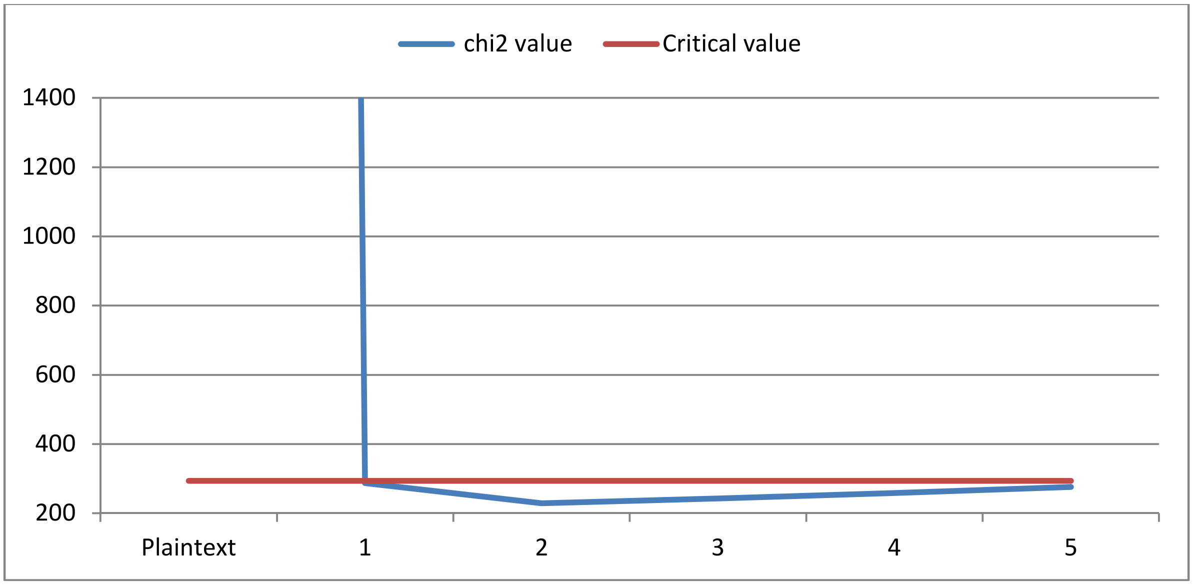
Plaintext and encrypted files need to be verified for uniformity of bytes distribution using χ^2 goodness of fit test. First of all, basic values for analysis are calculated: χ^2 critical value, degrees of freedom and confidentiality level. As encryption was performed byte-by-byte, 256 possible values of bytes exists. This means df = 255 degree of freedom. Additionally, confidentiality level for the research is assumed as α = 0.05 (95%). This means that critical value is **293,2**.

For each new round of encryption the result of previous round was used as a plaintext. The result of χ calculations was compared to critical value. To state that null hypothesis is not rejected χ^2 value needs to be less than 293.2. As a result, minimum obtained value is marked and also first round which gives result less than critical value.



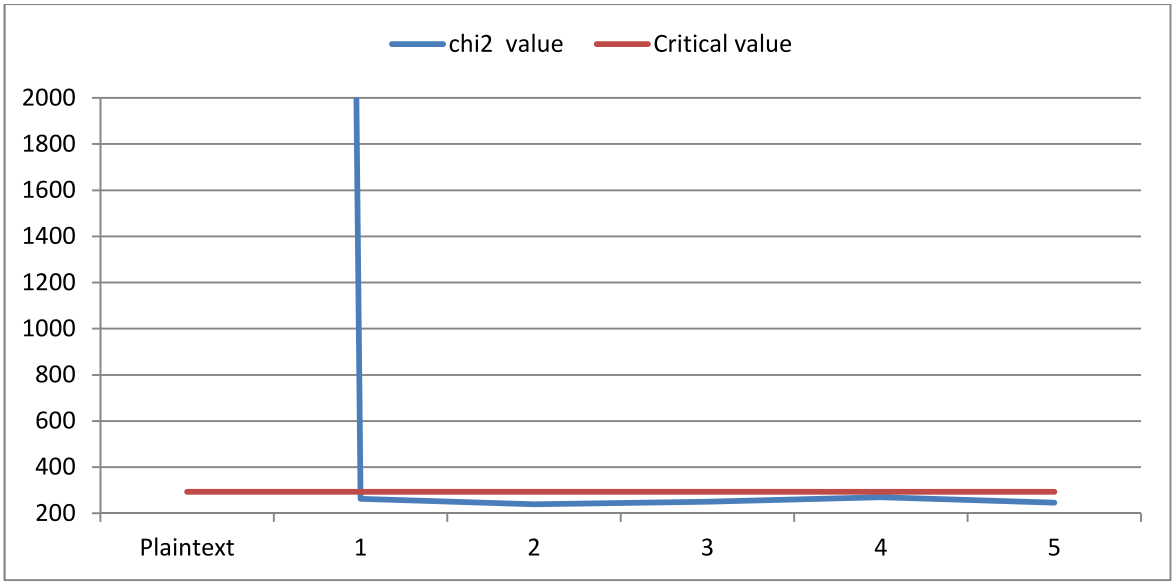
GRAPH 8. Encryption of .mp3

As seen from the Graph 8 above for digital sound file, .mp3 format 10 round of encryption was performed. The red line on the graph shows critical value 293.2 (α = 0.05, df = 255). Plaintext χ^2 for this file is 1358601. First round of encryption gives the result of distribution far from uniform but it gives a significant influence lowering value of χ^2 to 3500. As seen, minimum required rounds of encryption is 5 that gives satisfying result. It is possible to continue encryption but only to 7th round. After that due to use of pseudorandom number generator of simple type the value starts to rise again.



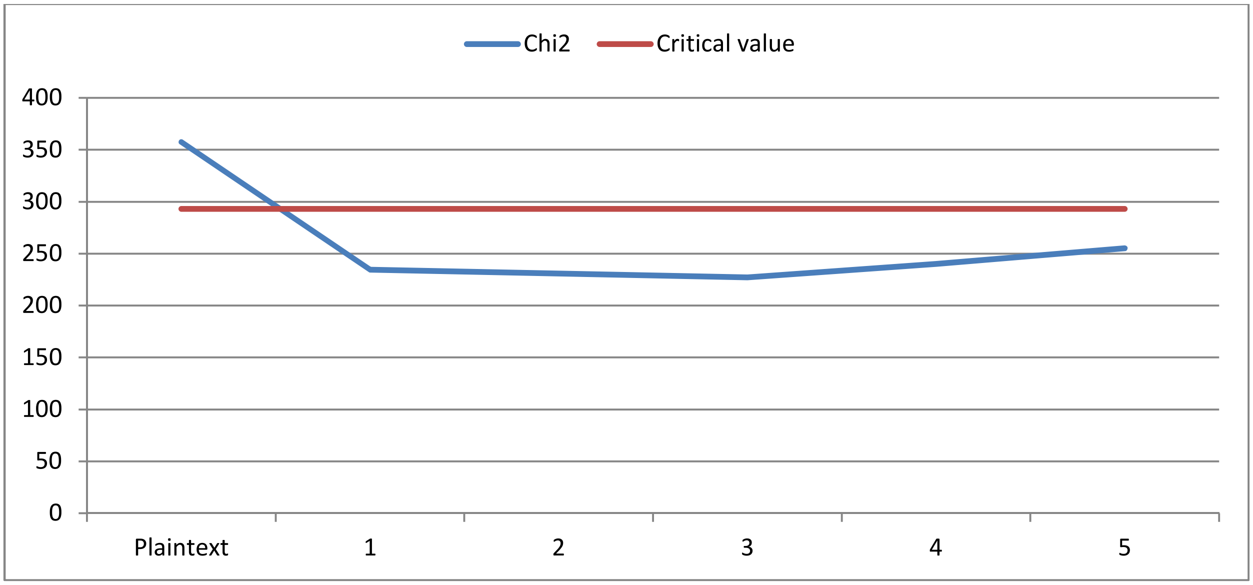
GRAPH 9. Encryption of .jpeg

Unlike sound file, .jpeg, presented on Graph 9, required only one round of encryption to have the bytes distributed uniformly. Already on the second round it is seen that the value is below the red line showing critical value of 293.2. Also, comparing to .mp3, plaintext χ^2 value was less, only 367490.8. In this case after several rounds of encryption performed, phenomena of graduate growth of χ^2 value was observed as well.



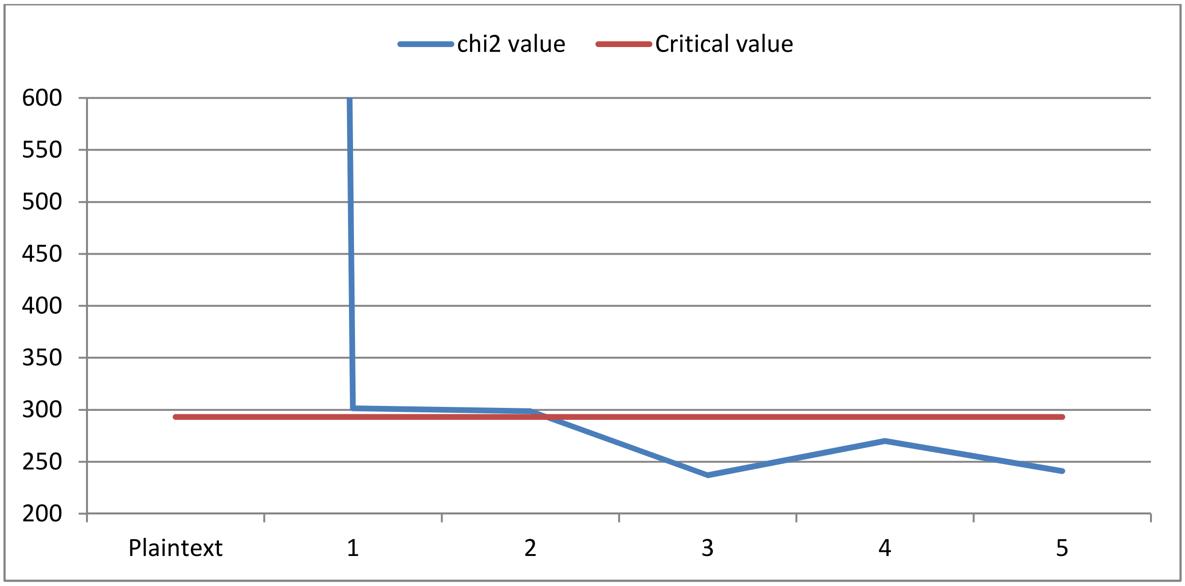
GRAPH 10. Encryption of .mp4

The case of the .mp4 file is quite similar to the .jpeg. For plaintext it has higher χ^2 value which equals 1181257.8. Though after first round of encryption, bytes are distributed uniformly and, similar as already observed for image file, reaches its χ^2 minimum after second round of encryption. Since then the value almost follows the red line's critical value 293.2.



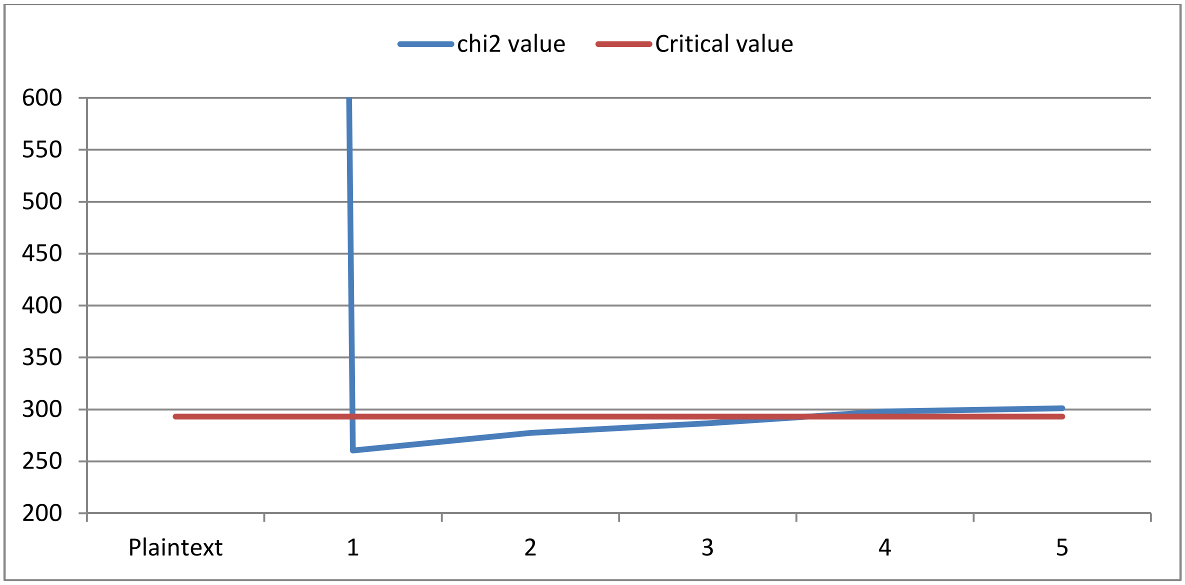
GRAPH 11. Encryption of .pdf

Graph 11 shows similar situation when only one round of encryption is required was observed for the .pdf file. Already χ^2 for plaintext is close to critical value, being 357.6 and first round of encryption gives the result 234.466. It is seen that the results for the first five rounds are lower than the critical value 293.2, they are marked red.



GRAPH 12. Encryption of .doc – plaintext

Finally, statistical analysis for text files was performed, it is presented in Graphs 12 and 13. Results in this case are different for the type of the text contained in the file. Plaintext χ^2 value for sample text is quite low, only 25194, whereas for the monograph text it is over 121333.



GRAPH 13. Encryption of .doc – monograph

Even the first encryption round gives quite good results for uniform distribution. Minimum requirement for sample text is 3 rounds – though even the second round is quite close to it, for one-symbol file even 1 round is sufficient. For the monograph file the χ^2 value after third round is already higher than critical value.

11 CONCLUSIONS

For the thesis different types of encryption were studied and statistical analysis for several types of files was performed. Different types of files have different structure and at the beginning it was assumed that some of them will require up to 15 encryption rounds for uniform distribution of bytes. In this research shows that media files – digital picture and video, as well as document file .pdf require only one round of encryption with one-time pad to have bytes distributed uniformly.

Different case is observed for the .mp3 file. The structure of the .mp3 file contains information additional text notes about the file – artist, track number. As seen from the research, simple text requires most number of rounds for encryption. Sound file .mp3 required more rounds of encryption than other media files due to contained text description.

Text file containing sample text required more rounds of encryption than media files. A sample containing only the monograph unexpectedly needed only one round to have bytes distributed uniformly. On the other hand, file containing monograph was the one that reached its minimum χ^2 also on the very first round.

It should be noticed that for one time pad of high importance is pseudo random number generator. For this research quite primitive one was used but it showed good encryption level. If pseudo random number generator is closer to generating truly random numbers the results should differ.

For confidentiality level set to 95% the results for minimum encryption rounds is as follows: 1 round with value 287.031 for .jpeg, 1 round for value 263.111 for mp4, 5 rounds for value 262.22 for .mp3, 3 rounds for value 237.199 .doc plaintext and 1 round for value 260.605 .doc monograph, single round for value 234.466 for .pdf file. So the maximum needed rounds are five though 1 round is enough for most file types. For the most uniform distribution – and more secure different results are observed: 228.458 after 2 rounds for .jpeg file, 238.908 after

2 rounds for .mp4, 209.835 after 7 rounds for .mp3, 237.199 after 3 rounds for .doc plaintext, 230.605 after 1 round for .doc monograph, 234.466 after 1 round for .pdf.

As seen from the above results, for most of cases 2 rounds would be a balance between higher level of uniform distribution of bytes and fewer number of encryption rounds. Also, it is seen from the research that χ^2 value of the plaintext does not have significant impact on number of rounds required for encryption. Unencrypted plaintext with high χ^2 value can require only one encryption round as well as file with plaintext χ^2 close t critical value.

REFERENCES

Barker, E., Kelsey, J. 2012. NIST Special publication. Computer Security. Recommendation for Random Number Generation Using Deterministic Random Bit Generators. National Institute os Standards and Technology, U.S.

Churchhouse, R. F. 2004. Codes and Ciphers. Julius Caesar, the Enigma and the Internet. Cambridge: Cambridge University press

Delfs, H., Knebl, H. 2007. Introduction to Cryptography. Principles and Applications. Second edition. Berlin, Heidelberg: Springer

Dent, A.W., Mitchell, C.J. 2005. User's Guide to Cryptography and Standards. Boston, Norwood: Artech House, Inc.

Dorichenko, S.A, Yashenko V.V. 1994. 25 etyudov o shifrah. Moscow: TEIS

Filliben, J.J, Heckert, A. 2012. Engineering Statistics Handbook. Available: http://www.itl.nist.gov/div898/handbook/eda/section3/eda35f.htm Accessed 15 February 2015

Kahn, D. 1996. The Codebreakers: The story of secret writing. Abridged version. New York: The New American Library.

Kerckhoffs, A. 1883. "La cryptographie militaire" Journal des sciences militaires, vol. IX, pp.

Lek, K, Rajapakse, N. 2012. Cryptography, Steganography and Data Security: Cryptography: Protocols, Design and Applications. New York: Nova Science Publishers.

Moldovyan, A., Moldovyan N. 2006. Innovative Cryptography. Second Edition. Boston: Course Technology/ Cengage Learning

National Institute of Standarts and Technlogy. Computer Security division. Computer Security Resource Center. Random Number Generator. 2015. Available: http://csrc.nist.gov/groups/ST/toolkit/random_number.html
Accessed: 20 March 2015

Oppliger, R. 2005. Contemporary Cryptography. Norwood: Artech House, Inc.

Ryabko, B.J., Fionov, A.N. 2004. Osnovy sovremennoy kryptografii dlya specialistov v informacionnyh technologiyah. Moscow: Nauchnyj mir.

Schneier, B. 1996. Applied Cryptography: Protocols, Algorithms, and Source Code in C. Second Edition. New York: Wiley Computer Publishing, John Wiley & Sons, Inc.

SciEngines press-release. 2009. Break DES in less than a single day. Available: http://www.sciengines.com/company/news-a-events/74-des-in-1-day.html
Accessed: 20 February 2015

Shannon, C.E. 1949. Communication Theory Of Secrecy Systems. Bell System Technical Journal, vol. 28-4

Silverman, J.H., Pipher, J., Hoffstein, J. 2008. Introduction to Mathematical Cryptography. Berlin, Heidelberg: Springer

Zenzin, O.S., Ivanov, M.A. 2002. Standart kryptograficheskoj zashity – AES. Konechnye polya. Moscow: Kuditz-obraz.

APPENDICES

Appendix 1

In C:\ create folder “test”.

Put file test.txt, containing stream of bytes in the folder.

Run “Aes.exe”. Files “encrypted_test.txt” and “unencrypted_test.txt” will appear in C:\test.

To encrypt additional round, rename “encrypted_test.txt” into test and start “Aes.exe” file again.

Software should be run as many times as many rounds are required.

As files are re-written, results need to be saved in different storage additionally.

Appendix 2

Round of encryption	χ^2 value	Information
1	3500.314	
2	3024.074	
3	2314.198	
4	1871.165	
5	262.228	<293.2
6	244.994	
7	209.835	Minimum value
8	234.209	
9	248.391	
10	260.292	

TABLE 1. Encryption of .mp3

Round of encryption	χ^2 value	Information
1	287.031	<293.2
2	228.458	Minimum value
3	242.695	
4	258.708	
5	276.095	

TABLE 2. Encryption of .jpeg

Round of encryption	χ^2 value	Information
1	263.111	<293.2
2	238.908	Minimum value
3	250.924	
4	269.603	
5	245.827	

TABLE 3. Encryption of .mp4

Round of encryption	χ^2 for sample text	χ^2 for one-symbol text
1	301.324	260.605
2	298.676	277.377
3	237.199	286.540
4	270.002	298.044
5	241.043	301.052

TABLE 4. Encryption of .doc