Encryption processes for scripts and the development of algorithms for stored data (subject review)

AlaaThamer Mahmood[1]
Technical Instructors Training Institute, Middle Technical University,Baghdad, Iraq[1]
alaa.thamer@mtu.edu.iq
RaedKamil Naser[2]
Administration Directorate, Ministry of Defense, Baghdad, Iraq[2]
ayhar_2013@yahoo.com

Abstract
Before tackling the security of systems, it is good to know a little about the different techniques available to ensure the security of communications. The science studying secret codes is called cryptography, and the science studying the weaknesses of these codes is cryptanalysis. They are both intimately linked, of course, and expertise in one implies a good knowledge of the other. This principle remains general to all security problems that you may encounter, computer or not; you cannot be a good security expert without potentially being a good thief. I emphasize the potentially, the goal is not to be a literal thief, but rather to master the techniques and the way of thinking in a legal framework.
Cryptography therefore relates, in a broad sense, to information masking techniques. It is a very old discipline, which has been historically linked to the diplomatic and military fields for which the secrecy of correspondence is crucial. Today, everyone uses some form of cryptography, usually without knowing it,your bank card, email account, Amazon order page and even your login password are based on so-called cryptographic primitives, i.e. mathematical functions optimized for masking some information.

Keywords:cryptographic, Mono-alphabetical,

Introduction:

Encryption or encryption is the technical term of the “secret code”, and consists of transforming a message to make it unreadable by a third party.We are talking about plain text and encrypted text, and their English equivalents complaintxt and ciphertext.

Cryptanalysis seeks to break encryption algorithms and cryptographic primitives.Encrypt does not exist in French, it is an Anglicism. However, it has become commonplace and I often make this mistake myself.

1. **Mono-Alphabetical ciphers**

Historically, one of the oldest known encryption methods is called the Caesar code. It is said that Julius Caesar used these algorithms to encrypt his communications during wars, but in reality, he used this number for his personal communications! How Caesar's cipher works is extremely simple. Just take the usual alphabet, and shift the letters of $k \leq 25$ to get the corresponding encrypted alphabet. Thus, for $k = 3$, we obtain the equivalence $A \rightarrow D$, $B \rightarrow E$ ... $Z \rightarrow C$. We speak of mono-alphabetical encryption, because each letter corresponds to one and only one substitution. The encrypted letter D in our previous example corresponds, in the encrypted message, only to A. It is therefore possible to use only one substitution alphabet [1]:

1.1. Mono-Alphabetical figures

One can easily express a Caesar figure in mathematical form. For this, we associate each letter of the alphabet in the usual lexicographic order with a number from 0 to 25, and we use the principles of integer arithmetic modulo $N = 26$. In the example we have given previously, from a letter of the alphabet x, we obtain the encrypted letter c by the operation:

$$a = b + x \bmod N \qquad N = 26$$

In this equation, the variable x corresponds to plain text, the variable c to encrypted text and the variable k to the encryption key. It is this key that indicates how to decrypt the message. If we reverse the previous equation, we have:

$$a = b + (N - x) \bmod N \qquad N = 26$$

A Caesar cipher therefore has 25 possible keys. This number is extremely low, and provided that a cryptanalyst knows that the transmitter has used a Caesar cipher, it would not take him very long to obtain the offset k used by an exhaustive test[2].

By keeping the principle of the Caesar cipher, it is possible to define a more secure form of mono-alphabetic cipher, by performing a random permutation of the alphabet. As before, we encrypt a message by replacing the letter encountered by the equivalent in the permuted alphabet. This type of encryption is therefore called substitution encryption.[3]

The operation therefore resembles the figure of Caesar, but the number of possible keys increases enormously. In practice - and I encourage you to understand where this result comes from, it is a simple combinatorial reasoning - this number is $26! = 400 \times 1024$. There is therefore little chance of being able to break this code by proceeding

brutally and testing each key. However, in practice, it is very simple to encrypt these codes. We're actually betrayed by language, but let's take an example:

JHFUD JGJG JFJJOL NRIRUJ PWQSD FR F JFJJRFKK JGJKFRKJ VEURBG

First of all, in this text, we see appearing artifacts helping deciphering. The spaces in the first place help us to cut out the words, and will facilitate the discovery of the substitution alphabet.[4].

1.2. Poly-Alphabetical Encryption

We see that H is a single letter, which means that if the text is in French it corresponds to a ory. The letter M precedes an apostrophe in two cases, and would therefore correspond to l, d, n, m or j. By doing so, the cryptographer can get a part of the key and play hangman with the rest by making hypotheses.[5]

It is therefore good practice to remove the punctuation first, and replace the spaces with a rare letter in the language of the message, such as w in French. The encrypted message will already seem much more difficult to crack, but not impossible. Mono-alphabetic ciphers reveal, whatever happens, the statistics of the language, that is to say the frequency of appearance of letters, diagrams, trigrams or even words.

HLKDJ HFUEBHD HDFJUWO SJDIUF NFDJDFJ JKDJD QIEU JFJERJ DJFIW AODHD JDFIEBG NJFGUVBE JFGPEJUR JDIEB JEIEBH JDIEWB JDRINWE JEJUBE JEDJNE KJEJBWE LFIWJ KJWHVBWE KWIDRNB JFJIEB KWBE KFJE WIWB JFBHB KPEB MNSAW JFUWE NHDBV OWBBH JWKIB JKDBWJI POIQWE NBCXH NSDFG JKHDUEB.

In the previous text, the letter C is clearly the most common, with 36 appearances over 225 letters. We can therefore infer that it is probably the letter e, the most frequent in French at 16%, or space if the cryptographer has chosen to replace spaces with letters. We can also be interested in the distribution of digraphs, groups of two letters. Thus, in this example, the XC and CX diagrams are the most common, with 6 occurrences each! If I had to crack this text, I would infer that these two diagrams are ET and TE, two of the most frequent in English. If time allows, you will have the opportunity in TP to perform a substitution encryption cryptanalysis. If not, I encourage you to go see the references for this course, or to read The Golden Beetle, where the author has his hero decipher an old message indicating the location of a treasure, noting the details of his reasoning .[6]

2.ProposedPoly-alphabetical ciphers

Frequency analysis therefore makes it possible to crack any encryption by substitution, subject to having access to enough encrypted text to see the appearance of distributions of letters and usable diagrams to overcome this problem.

2.1 Poly-Alphabeticalciphers

So-called poly-alphabetical ciphers. In mono-alphabetical encryption, the letters in the encrypted text always correspond to one and only one letter in the plain text. Thus, by an arbitrary technique, if we decided that the letter C corresponded to the letter e, it was the same for all the letters C of the message. Poly-alphabetical ciphers remove this restriction, which makes the cryptographic system sensitive to frequency attacks. In practice, each letter of the plain text will be encrypted by a substitution alphabet specific to that letter. The alphabet will be determined for each letter by the cryptographic system, and all the alphabets will therefore form the key to the message.

The operation is very simple, which contributed to its popularity until the beginning of the 20th century. The interlocutors choose a word which will serve as the key for the message, for example AALI. This key is repeated above the plain message, and provides each letter with an offset corresponding to the index of the letter in the alphabet. The letters are then encrypted by a Caesar code using this offset.

Key	A 12	A 9	L 15	I 14	A 12	A 9	L 15	I 14	... ...
Offset	A	B	Y	F	O	U	I	G	...
Clear	Q	N	S	J	F	T	Z	D	...

Although this figure is based on the simple substitution of Caesar, his cryptanalysis is much more complex! We see indeed that the letter e of the clear message was associated in the encrypted message with the letters M and R, and that the letter X corresponds to both m and p. Poly-alphabetical ciphers thus mask the frequency distribution of the language in encrypted messages, and make cryptanalysis more difficult. The later section in this section presents the method used to break Vigenère's cipher, but there are much more resilient poly-alphabetic ciphers. Thus, the Enigma machine used for communications from Germany during the Second World War held in check the sum of the combined brains of England and the United States until 1943, and it was basically a poly encryption - alphabetical. Like the Vigenère figure, the weakness of these methods lies in the choice of substitution alphabets. Where Vigenère used a Caesar, Enigma used a complex mechanical process which ultimately presented a statistical bias. By dint of recovering encrypted messages, the cryptanalysts of the Allies gradually managed to reconstruct an equivalent of the machine and to determine the alphabets used. It is very likely that this advance played a big role in their victory, the Germans ignoring that their communications were now intercepted by the United States.[6]

2.2 Modern figure and symmetric cryptography

The acronym NDNF is repeated in the message, as well as the acronyms GJ and YCM. These acronyms are separated by 54, 24 and 12 letters respectively. As a poly-alphabetical encryption would mask this kind of statistical bias with a high probability, repetitions like these on such a short message are due to the fact that the same (di-tri-quadri-) grams of plain text have been encrypted with the same part of the key! The latter is therefore necessarily a common divisor of 54, 24 and 12. We can therefore test the keys of size 3, 4, 6 (which is the size of the key I used, CRYPTO) until obtaining a cryptanalysis sensible.[7]

There is a somewhat less artisanal method of carrying out this analysis. We use a statistical metric called coincidence index, which is defined as follows, with fi the frequency of each letter in the encrypted message: [8]

Without going into details, this metric will indicate the shift between our text and a text where all the symbols are equiprobable, where in this case we will have IC ≈ 1. Each language will have a specific coincidence index, plotted on the table opposite. This metric allows, among other things, for a mono-alphabetical encryption, to know the language used in plain text.[9]

Language	**IC**
Eng	2,84
Fre	3,13
Ger	3,16
Spa	2,83

We use the coincidence index to crack Vigenère's figure as follows. We assume a key size (2, 3,...) and for each size we calculate the index of coincidence of the message from the letters spaced by the supposed size of the key. If there is a coincidence index close to 1, then the key size assumption is not correct. Otherwise, it's likely that you've encountered the correct size, or a multiple of that size. The following table gives an example of this use, where the key is of size 5.[10]. On a short message size, we can therefore identify the size of the key and carry out a frequency analysis which will quickly give the offsets used for the codes of Caesar.

Size	index
1	1,12
2	1,19
3	1,05
4	1,17
5	**1,82**

6	0,99
7	1,00
8	1,05
9	1,16
10	**2,07**

SKFJF URYEE OPWER JGIFR JFGKK JJOEB JREOWP JGBKUI NGFJFR ALPIWE JFINHR KIFHK KIFGHK IUWYRI KVGBKL KFNRO ORENHLK KFBHK JGFHIK JGIOEI JGLNRE KJGLJR .

2.3 Modern ciphers and symmetric cryptography

The Second World War brought cryptography into the industrial and scientific fields. In particular, all the mathematicians of the time were recruited by the armies to serve as cryptanalysts, and this generated a large number of advances in the field. In particular, Claude Shannon stated in a 1948 paper two basic principles for the design of cryptographic primitives:

- The algorithm should help confuse the key with the message. Ideally, this means that each symbol of the encrypted message depends on the clear message and all the symbols of the key. This dependency must also appear random for an attacker

-The algorithm must broadcast the statistical properties of the clear message. Contrary to the principle of confusion, which dealt with the relationship between the encrypted message and the key, dissemination implies that each letter of the encrypted message must depend on a maximum of letters of the clear message, in order to "hide" its statistical distribution.

You are now able to understand the interest behind these two maxims. The first relates to encryption methods like that of Vigenère, where the relationship between the key and the encrypted message is too deterministic and allows cryptanalysis. The second aims to discourage approaches based on the frequency analysis of the encrypted message. We can illustrate these two principles in pictures

In this case, each pixel block is encrypted independently using a substitution similar to poly-alphabetic encryption. We can clearly see the original image appear in the second image, even if it is scrambled! The confusion is good, but the diffusion is clearly not, and the algorithm must be further developed to find an image close to random noise.

3.Methodology:

Modern encryption methods use these two principles. In particular, the current standard in this area, the AES (for Advanced Encryption System) works as follows.

The message is divided into blocks of 16 symbols, and each block passes through a transformation formed by 4 simple operations.

Upstream of the encryption, the key is extended into a certain number of sub-keys which will be added at each turn of the algorithm. During a turn, we perform a deterministic substitution in the manner of a mono-alphabetic cipher, then we diffuse the symbols inside the matrix formed by the block by inverting the lines and applying linear operations on the columns. The first and the last stages thus ensure confusion, and the two intermediate stages the diffusion. After about ten turns, we obtain an encrypted block. Each of these operations being invertible, it suffices to carry out the reverse path through the algorithm to decipher the message. An interlocutor having knowledge of the key can therefore proceed to its extension into sub-keys and obtain the clear text.

We speak in this case of symmetric encryption. The sender and the recipient use an encryption and decryption algorithm and an identical, shared key. Communication security is therefore ensured as long as an attacker does not know the key, according to Kerckhoffs' principle. The difficulty stems from sharing the key, which must be exchanged securely between the two parties. If a physical meeting can take place, there is no problem, but other techniques must be implemented in order to be able to communicate the key securely over the Internet, for example.

4.Asymmetric cryptography

This key exchange problem is fundamental to cryptography, and remained intractable until 1970 when Whitfield Diffie presented a revolutionary idea. The idea was to use two different keys; a public key used to encrypt the message, and a private key used to decrypt it. As long as it is not possible to guess the private key from the public key, the system is secure. The public key is freely accessible and only the holder of the private key can decrypt the message.Diffte and Hellman's good idea

Although Diffie presented his idea, it was not until 1977 that an asymmetric cryptographic system allowing the sending and receiving of messages appeared. However, Diffie and Hellman proposed a very intelligent algorithm to create a shared key between two interlocutors, thus solving one of the problems of symmetric cryptography. Like all asymmetric cryptography systems, their algorithm is based on one-way functions, which are easy to calculate in one direction but difficult in the other. The best known is undoubtedly the prime factorization problem; from 2 prime factors a and b, it is simple to calculate their product c = ab, but it is extremely difficult to find a and b by knowing only the result c of the operation, if a and b are sufficiently large.

The operation used by Diffie and Hellman is near. This is the problem of the discrete logarithm, which is the inverse of the discrete exponentiation:

$$A = g^a \quad \text{mod} p$$

As with prime factorization, it is extremely difficult for large values of p to discover the exponent used from the value of A, g and p. The algorithm is as follows. Alice and Bob seek to create secret value without ever exchanging it directly with each other.

A				B		
Security	generl	Calculated	Send	Calculated	general	Security
a *a* *a* *a ..s*	*p ..j p*, *j*et *A p*, *g* et *A* *p, j,A ..D*	$A = g^a \bmod p$ $c = B^a \bmod p$	p..j → *A*→ ←*B*	$B = g^b \bmod p$ $c = A^b \bmod p$	*p ..j* *p ,j,A..cp ,j,* *A..*c	*b* *b* *b* *b* et *s*

In less formal terms, we can give the following description:

1 .A has a whole secret, whatever. It chooses a large prime number p and a base gthat it sends to Bob, and it also sends the value A = j c mod p.

2 .B has a whole secret b, whatever. Having received g and p, it refers to Alice B = jc mod p.

3 .The two parties calculate s = Ba mod p = Ab mod p = g a.b mod p using their respective secrets, and are therefore in possession of a shared s value without ever having sent a and b over the network.

An attacker who only knows A, B, g and p cannot directly deduce the value of s. For this, he would need the value of a or b, which he can only obtain by solving the problem of the discrete logarithm on A or B!

The current reference for asymmetric cryptography systems is called RSA, named after its inventors Ronald Rivest, Adi Shamir and Leonard Adleman, in 1977. Operation is fairly simple, but requires notions of modular arithmetic 1. Take however, time to read this page, which gives you a vision of the mathematical subtlety present in modern cryptography. RSA must generate 2 keys, according to the following algorithm:

1. We choose 2 prime numbers p and q.
2. The first part of the public key is n = pq, which becomes the modulo in the arithmetic calculations that follow.
3. Then calculate A = (p - 1) (q – 1).
4. We then choose e <A such that e is prime with A (in practice it is simple, it suffices to choose a prime number e which is not a divisor of A). The number e forms the second part of the public key. We can therefore publish (n, e) in the eyes of all.

5. Find d = e − 1 mod A. It is a modular inverse, so we must return to the definition of inverse in a ring: de = 1 mod A.
6. The pair (n, d) forms the private key of the cryptosystem.

Security comes from the fact that to form the private key, we must calculate A. For an attacker knowing only N the only way to get there is to factor N to discover q and p, a very difficult problem if p and q are large (a 2048-bit key is made up of more than 600 decimal digits).

To encrypt any message M, we cut it into pieces. These blocks become the symbols, or the alphabet, of the message, and are assigned a number m such that m <n. To encrypt a block, we calculate:

$$c = me \bmod n$$

Decryption is carried out with the private key in a relatively simple way:

$$m = cd \bmod n$$

It should be noted that although the principle is not complicated for a computer, when the numbers are so large the calculation time is still high. On average, public key encryption takes 1,000 times longer than secret key encryption. Instead, public key ciphers will be used for message signing, encryption of highly sensitive information (bank transactions) or the exchange of secret keys. For example, by logging into your bank's website, you create a secret encryption key that is sent to the bank, encrypted using the bank's public key. Then you can exchange messages by simply encrypting using the shared key, which makes communication smoother. It is not immediately clear why RSA works. To do this, we need two arithmetic results:

1. Fermat's small theorem) Let p be a prime number and have any number, prime with p. We then have a (p − 1) = 1 mod p.
2. Chinese remainder theorem) Let p and q be two prime numbers between them, and (a, b) two whole numbers less than p and q. If a = b mod p and a = b mod q then a = b mod (pq).

We want to prove that:

cd = (me) d = m mod n

We can verify that, by Fermat's little theorem:

(ne) d = ned = n • ned −1 = n • nh (p − 1) (q − 1) = n • 1h (q − 1) = n mod p

Similarly:

(me) d = med = m • med −1 = m • mh (p − 1) (q − 1) = m • 1h (p − 1) = m mod q

We apply the theorem of Chinese remains to obtain:

(me) d = m mod pq⇔ (me) d = m mod n

Example of using RSA

We take simple values, for example p = 2 and q = 5. In this case, A = 4 and we need an exponent e which is prime with 4 and less than 4. Not too much choice, we take e = 3 We are looking for d such that de = 1 mod A, which gives us for example d = 7. We would also have 3 but in practice we often have d / = e

Since we have n = 5 × 2 = 10, so our alphabet must contain less than 10 letters. Choose by:

example {F, M, B, C, W, J, E, V, Z} and let's encrypt the word LUTIN (whose representation will be (9,8,5,4,3) :

Step:

Text clair	F	M	B	C	W
Equivalents	10	9	6	5	3
m^e	864	622	495	398	7
m^e mod$n$	10	3	4	7	9

Encrypted text

Cd	4782969	128	2187	279936
2097152				
Cd mod n	9	8	7	6
2				

Text equivalent

6. Hash and signature functions

Our (short) presentation of cryptographic primitives ends with hash functions. In its simplest form, a hash function is a mathematical function that returns a value of a fixed and predetermined size. For example, for a message x of N letters, where N is an arbitrary integer, the hash function h (x) will always

return a value represented by B bits of data, called a hash. The size of the hash depends on the function, but is often 128 or 256 bits.

This function is however not chosen completely at random, and must respect a few rules to be useful in a cryptographic system:

-For a given hash value y = h (x), it must be difficult to find x (resistance to pre-Pictures).

-It must be difficult to find x1 and x2 such that h (x1) = h (x2) (resistance to collisions).

-For a given x, it must be difficult to find xf such that h (x) = h (xf) (resistance to pre-seconds Pictures).

Obviously, a hash function must be deterministic, and return the same value h (x) = h (xf) = y for x = xf.

The basic use of hash functions is found in network protocol error detection algorithms, such as the CRC field of the Ethernet protocol. By calculating the hash of the transmitted packet, the sender allows the recipient to verify by calculating the hash in turn that the message has not been modified. However, a probability of non-detection remains, of course. As the domain of definition of h is larger than its domain of arrival, the function is not injective! As a result, the message may have been modified so that a collision occurs. The properties mentioned aboverather relates to the resistance of these functions to a malicious attack. When reading them, you should realize that the main use of hash functions is going to be for signing digital documents.

Conclusion:

The algorithm is fairly simple if you understand RSA. We assume that the sender has a public key (n, e) and a private key (n, d), and wishes to send a message m:

1 .He starts by calculating H = h (m), then he calculates a signature with his private key sig = Hd mod n.

2 .The message with its signature is sent to the recipient, who therefore receives (m, sig) and knows (n, e), the sender's public key.

3 .The recipient calculates H f = seat mod n, and verifies that h (m) = H f to accept the message. The transmitter therefore uses symmetry 2 of the RSA encryption algorithm to encrypt the hash of the massage using his private key. Using the properties of the discrete exponentiation, the recipient can retrieve the original hash with the public key. If the message has been modified, then the hashes will not match, and the same applies if a new signature has been

generated with a public key different from that of the sender. This operation is summarized in the diagram below.

Securing information systems is a discipline in its own right, which requires extensive knowledge of system vulnerabilities and associated protection measures. It is obvious that at the end of this course you will not be security experts, but this introduction should give you the keys necessary to understand the problems, avoid them at your level and be able to communicate with "real" experts on the projects you will carry out in your professional life. It also happens that users are the real big flaw in all systems, and that IT security education is becoming central in higher education.

There are several ways to deal with information system security, and of course a number of good practices. It is nevertheless essential to consider security as a global approach; there is no point in having an armored door to a warehouse with fragile and accessible windows, and you are never more secure than the weakest link in your communication chain.

Reference:

[1] RakeshAgrawal , Jerry Kiernan , RamakrishnanSrikant , YirongXu, Hippocratic databases, Proceedings of the 28th international conference on Very Large Data Bases, 2002, pp.143-154.

[2] Ernesto Damiani, S. De CapitaniVimercati, SushilJajodia, Stefano Paraboschi, and PierangelaSamarati, Balancing confidentiality and efficiency in untrusted relational dbms, Proceedings of the 10th ACM conference on Computer and communications security, ACM, 2003, pp. 93-102.

[3] Luc Bouganim and Philippe Pucheral, Chip-secured data access: confidential data on untrusted servers, Proceedings of the 28th international conference on Very Large Data Bases. 2020, pp. 131-142.

[4] HakanHacigumus, Balakrishna R. Iyer, and SharadMehrotra, Effcient execution of aggregation queries over encrypted relational databases, DASFAA, 2019, pp. 125-136.

[5] Sun S. Chung and GultekinOzsoyoglu, Anti-tamper databases: Processing aggregate queries over encrypted databases, Proceedings of the 22nd International Conference on Data Engineering Workshops, Washington, 2006, pp. 98-107.

[6] RakeshAgrawal, Jerry Kiernan, RamakrishnanSrikant, and YirongXu, Order preserving encryption for numeric data, Proceedings of the 2004 ACM SIGMOD international conference on Management of data, ACM, 2018, pp. 563-574.

[7] TingjianGe and S. Zdonik, Fast, secure encryption for indexing in a columnorienteddbms, IEEE 23rd International Conference on data engineering, 2007, pp. 676-685.

[8] Feifei Li, MariosHadjieleftheriou, George Kollios, and Leonid Reyzin, Dynamic authenticated index structures for outsourced databases, Proceedings of the 2006 ACM SIGMOD international conference on Management of data, ACM, 2020, pp. 121-132.

[9] HweeHwa Pang, Jilian Zhang, and KyriakosMouratidis, Scalable Verification for Outsourced Dynamic Databases, Proceedings of the 35th international conference on Very Large Data Bases. 2019, pp. 802-813.

[10] Min Xie, Haixun Wang, Jian Yin, and XiaofengMeng, Integrity auditing of outsourced data, Proceedings of the 33rd international conference on Very large data bases, 2017, pp. 782-793.