Applied Mathematical Sciences, Vol. 4, 2010, no. 20, 967 - 976

Algorithm for Strengthening Some Cryptographic Systems

Víctor Manuel Silva, G. Rolando Flores C., Itzamá López Y. and Carlos Rentería M.[1]

Centro de Innovación y Desarrollo Tecnológico
en Cómputo-IPN, CIC-IPN, ESFM-IPN
Instituto Politécnico Nacional
07340 México, D.F., MEXICO
vsilvag@ipn.mx, renteri@esfm.ipn.mx

Abstract

In the present work a field is built on the set of permutations, using the construction of a bijection, I_m, which goes from the set $N_m = \{n \in \mathbb{N} \,|0 \leq n \leq m! - 1\}$ to the set $\Pi_m = \{\pi|\, \pi$ is a permutation of array $0, 1, 2, \ldots, m-1\}$. During this work, it is shown that this bijection defines an isomorphism $I_m : N_m \longrightarrow \Pi_m$; also, two binary operations $—\oplus$ and $\odot—$ on the set Π_m are defined, such that a field can be built on a subset of Π_m. On the other side, it is also possible to define a unary operation on Π_m whose result is the inverse of a permutation, denoted as $\neg\pi$. Finally, an application of this tool to cryptography is presented, specifically to symmetric cryptosystems DES and Triple DES [1], with a modification: the inverse permutation at the end of the ciphering process is not applied; only the initial permutation is applied, but using a variable permutation. It can be seen that, as a result, both cryptosystems are strengthened against brute-force attacks, as well as against differential and linear cryptanalysis in the case of DES [2],[3].

Mathematics Subject Classification: 94A60, 11T71

Keywords: Cryptography JV theorem, DES, Triple-DES

1 Development

Given two non-negative numbers n, m such that $0 \leq n \leq m! - 1$ and using Euclid's Division algorithm [4], number n can be written in a unique manner

[1]Partially supported by, IPN, CONACYT,COFFA-IPN and SNI-SEP, México.

as follows:

$$n = C_0(m-1)! + C_1(m-2)! + C_2(m-3)! + \ldots + C_{m-2}1! \tag{1}$$

Notice that $n \in N_m = \{n \in \mathbb{N} \,|0 \leq n \leq m! - 1\}$ and $(m-1)!, (m-2)!, \ldots, 1!$ are considered to be fixed.

Also, it is simple to prove that

$$0 \leq C_i < (m-i), \text{ with } 0 \leq i \leq (m-2) \tag{2}$$

Now, once the values of $C_0, C_1, \ldots, C_{m-2}$ are computed, the following algorithm can be built:

Step 0 An increasing array is defined in the following way: $X[0] = 0, X[1] = 1, X[2] = 2, \ldots, X[m-1] = m-1$.

Step 1 Using expression 2 we have that $C_0 < m$; then $X[C_0]$ is one of the elements of the array from step 0. $X[C_0]$ is eliminated from that array and a new array is started from $X[0]$ to $X[m-2]$.

Step 2 Again, according to expression 2 we have that $C_1 < m-1$; then $X[C_1]$ is one of the elements of the array from step 1. In the same fashion as in the former step, $X[C_1]$ is eliminated from the array from step 1 and a new array is started from $X[0]$ to $X[m-3]$.

Step m-1 If this is done repeatedly, at the end we will have the following array: $X[C_{m-2}]$ and $X[0]$.

Finally, the result of the eliminated numbers $X[C_0], X[C_1], \ldots, X[C_{m-2}]$ and $X[0]$ is a permutation of array $0, 1, 2, \ldots, m-1$. Then, it can be said that any $n \in N_m$ can be associated to a permutation in $m-1$ steps. At this point a question arises: given two different numbers from the set N_m, will they have associated two different permutations? The answer to this question is given by the JV theorem, which is presented below. Notice that this theorem and its proof were taken from [5], where a more ample discussion is given.

Theorem 1.1 (JV Theorem) *Given the sets* $N_m = \{n \in \mathbb{N} \,|0 \leq n \leq m! - 1\}$ *and* $\Pi_m = \{\pi \,|\pi$ *is a permutation of array* $0, 1, 2, \ldots, m-1\}$*, the algorithm described*
above defines a bijection from the set N_m *to the set* Π_m*.*

Proof.

The fact that function I_m is one-to-one will be proved by *reductio ad absurdum*.

Let us assume that for $n_1 \neq n_2$ with $n_1, n_2 \in N_m \Longrightarrow I_m(n_1) = I_m(n_2)$. From expression 1 we know that n_1, n_2 can be written as:

$$\begin{aligned} n_1 &= C_{0,1}(m-1)! + C_{1,1}(m-2)! + C_{2,1}(m-3)! + \ldots . + C_{m-2,1}1! \text{ and} \\ n_2 &= C_{0,2}(m-1)! + C_{1,2}(m-2)! + C_{2,2}(m-3)! + \ldots . + C_{m-2,2}1! \end{aligned}$$

Now, $I_m(n_1) = I_m(n_2)$ means that: $C_{0,1} = C_{0,2}, C_{1,1} = C_{1,2}, \ldots , C_{m-2,1} = C_{m-2,2}$. Thus $n_1 = n_2$, which contradicts the initial hypothesis. Then, we can conclude that if $n_1 \neq n_2$ with $n_1, n_2 \in N_m \Longrightarrow I_m(n_1) \neq I_m(n_2)$. The latter proves that function I_m is one-to-one.

The proof that I_m is onto is quite simple, given that the number of elements in both sets N_m and Π_m are equal. ■

In order to build a field on the set Π_m we need to define two operations, i.e. *addition* and *product*. However, we need also to build the inverse function of I_m: $(I_m)^{-1}$. This means that for a given permutation $I_m(n)$, we need to give the algorithm which calculates the value of n. The following algorithm does exactly such an operation.

Step 0 Start with an ordered array; that is, $0, 1, 2, \ldots , m-1$.

Step 1 Take the first element of permutation $I_m(n)$, say π_1, and locate the position of value π_1 in the array $0, 1, 2, \ldots , m-1$, counting from 0. Let us denote this position as C_0 and eliminate the value π_1 from the array obtained in step 0. The resulting array is denoted as $A\,(C_0)$. In fact, C_0 is the coefficient of $(m-1)!$; that is, $C_0\,(m-1)!$.

Step 2 Take the second element of permutation $I_m(n)$, say π_2, and locate the position of value π_2 in the array $A\,(C_0)$, counting from 0, denoted as C_1. Then eliminate said value from the array $A\,(C_0)$ resulting the array $A\,(C_0, C_1)$. The number C_1 is the coefficient of $(m-2)!$, and the two first summands in expression 1 would be: $C_0\,(m-1)! + C_1\,(m-2)!$.

It is not difficult to reach the conclusion that after $m-1$ steps, the value of n which we are looking for has the following expression:

$$n = C_0(m-1)! + C_1(m-2)! + C_2(m-3)! + \ldots + C_{m-2}1!$$

2 Definition of the Operations $\oplus$ and $\odot$, and Building the Field

Two algorithms were described in the former section. The first algorithm defines a bijective function which goes from the set N_m on the natural numbers

to the set Π_m of permutations, while the second algorithm defines the inverse function of the first one.

On the other hand, if we denote that functions as I_m and $(I_m)^{-1}$, respectively, then the operations $\oplus$ and $\odot$ are defined as follows:

Definition 2.1 *The addition of two elements $\pi_1, \pi_2 \in \Pi_m$ is expressed as:*

$$\pi_1 \oplus \pi_2 = I_m(((I_m)^{-1}(\pi_1) + (I_m)^{-1}(\pi_2)) \bmod m!)$$

Definition 2.2 *The product of two elements $\pi_1, \pi_2 \in \Pi_m$ is defined as:*

$$\pi_1 \odot \pi_2 = I_m(((I_m)^{-1}(\pi_1) * (I_m)^{-1}(\pi_2)) \bmod m!)$$

Using the latter two operations, we shall see that function I_m defines an isomorphism [4], which goes from the set N_m to the set of permutations Π_m.

Theorem 2.3 *The function $I_m : N_m \longrightarrow \Pi_m$ defines an isomorphism with operations $\oplus$, $\odot$. This means that the following properties are true:*

1. $I_m(a + b \bmod m!) = I_m(a) \oplus I_m(b)$
2. $I_m(a * b \bmod m!) = I_m(a) \odot I_m(b)$ *where* $a, b \in N_m$

Proof.
Let us begin with item 1, that is, $I_m(a + b \bmod m!) = I_m(a) \oplus I_m(b)$.

$$I_m(a) \oplus I_m(b) = \pi_a \oplus \pi_b$$

where π_a, π_b are the permutations associated to numbers a, b, respectively. However, by definition 2.1 we have that

$$\pi_a \oplus \pi_b = I_m(((I_m)^{-1}(\pi_a) + (I_m)^{-1}(\pi_b)) \bmod m!)$$

It follows that

$$I_m(((I_m)^{-1}(\pi_a) + (I_m)^{-1}(\pi_b)) \bmod m!) = I_m(a + b \bmod m!)$$

which proves item 1.

For item 2 we will follow a similar strategy; that is, to begin from the right-hand side of the expression:

$$I_m(a) \odot I_m(b) = \pi_a \odot \pi_b$$

However, by definition 2.2 we have that

$$\pi_a \odot \pi_b = I_m(((I_m)^{-1}(\pi_a) * (I_m)^{-1}(\pi_b)) \bmod m!)$$

Finally, it is concluded that

$$I_m(((I_m)^{-1}(\pi_a) * (I_m)^{-1}(\pi_b)) \bmod m!) = I_m(a * b \bmod m!)$$

■

Now let us prove the following theorem.

Theorem 2.4 *Starting from the two operations formerly defined,* $\oplus$ *and* $\odot$*, it is possible to build a ring on the set* Π_m*.*

Proof.
The proof is very simple, given that a ring can be built on the N_m set by using the properties of modular arithmetic, and the fact that I_m is an isomorphism with image in the set Π_m. ■

Before addressing the problem of the multiplicative inverse, it is important to notice that any element different to 0 from the set N_m —say $a \in N_m$— has a multiplicative inverse mod $m!$, if and only if the maximum common divisor of a and $m!$ is 1 [6]. In this sense, if a subset $N_m^p \subset N_m$ is chosen such that all elements different to 0 in the subset have multiplicative inverse, it follows that a subset of prime size must be chosen: $|N_m^p| = p$.

This prime number p holds to the condition $1 \leq p \leq m! - 1$. Actually, there are some particular cases where $m! - 1$ is prime, such as when $m = 3$ or $m = 4$. Then we proceed in the following manner: for a given positive integer m, a prime number p is chosen such that $1 \leq p \leq m! - 1$. Then, by using function I_m we can find the image of set $N_m^p = \{0, 1, \ldots, p-1\}$ which is $\Pi_m^p = \{I_m(0), I_m(1), \ldots, I_m(p-1)\}$. Both operations $\oplus$ and $\odot$ on Π_m^p are defined as stated in definitions 2.1 and 2.2, except that instead of using mod $m!$, mod p is used.

Theorem 2.5 *It is possible to build a field on set* Π_m^p *by using operations* $\oplus$ *and* $\odot$*.*

Proof.

According to theorem 2.4, the set Π_m^p forms a ring with operations $\oplus$ and $\odot$. It only remains to be proved that for any permutation $\pi \in \Pi_m^p$, and $\pi \neq \mathbf{0} = I_p(0)$. Then exists π^{-1} such that $\pi \odot \pi^{-1} = \pi^{-1} \odot \pi = \mathbf{1} = I_m(1)$. If $n \neq 0$ is the positive integer associated to π, it follows that exists n^{-1} such that $n * n^{-1} = n^{-1} * n = 1 \bmod p$ [6]. Thus, π^{-1} is the permutation associated to n^{-1} such that $\pi \odot \pi^{-1} = \pi^{-1} \odot \pi = 1$ ■

Now let us address another problem. Suppose that permutation π is a function that goes from the set $\{0, 1, 2, \ldots, m-1\}$ to itself; then the inverse permutation, denoted as $\neg\pi$, is that one which makes the following true:

$$\neg\pi \circ \pi = \mathbf{0}$$

Remark 2.6 *Notice that in the preceding expression, symbol $\circ$ represents function composition.*

Now, if we consider the unary operation $U(\pi) = \neg\pi$, it is not complicated to prove that it is not necessarily closed on set Π_m^p: given $\pi \in \Pi_m^p$, it is not necessary that $\neg\pi \in \Pi_m^p$. However, operation $U(\pi)$ is closed in Π_m since this latter set contains all permutations of array $0, 1, 2, \ldots, m-1$. The former property is relevant given that symmetric cryptosystems —such as DES or Triple-DES— start with a fixed permutation π [1],[7], and at the end of the cipher cycle permutation $\neg\pi$ is applied.

3 How the DES and Triple-DES Cryptosystems are Strengthened

It is well known that both the DES and Triple-DES cryptosystems begin with a fixed permutation [1]-[9]. In fact, it can be stated that in every cryptosystem of the Substitution Permutation Network kind, there is involved a permutation considered to be fixed. To this moment, no proposal has been made to allow these fixed permutation to be variable. These fixed permutations are applied by means of a table. For the DES and Triple-DES cryptosystems, an initial permutation IP is applied at the beginning of the cipher process, while at the end the inverse permutation $\neg(IP)$ is used. The IP permutation and the DES algorithm description are presented in [10]. It is noteworthy that the same permutation is used in the case of Triple-DES.

Now, by using the algorithm described in Section 1, it can be verified that the number associated to the IP permutation of the international norm is:

$$n_{IP} = 1145717915565593966585098047364889558943238666054$$
$$8771580773169226133877528403109435275327 9$$

In the present work, the cipher algorithms used do not apply permutation $\neg(IP)$ at the end of the cipher process. These algorithm shall be called, within this work, modified DES and Triple-DES. The former poses no problem, given that permutation IP does not increment the complexity of the DES and Triple-DES algorithms, since it is fixed. Actually, the initial block may be considered as clear text: L_0 and R_0 [8].

On the other hand, it is important to notice that a brute-force attack would not be feasible on a DES cryptosystem modified with a variable permutation. This is so because it would not be enough to know a block of clear text and its corresponding ciphered text, and try all 2^{56} keys (a process which currently is done [1]), since the initial permutation would not be known. Strictly speaking, a variable initial permutation would force a brute-force attack to try the 2^{56} possible binary keys for each permutation and find the right array at the output. It is clear that this would make this kind of cryptanalysis unfeasible, computationally speaking.

In this sense, one of the proposals of the current work is that the initial permutation IP applied at the beginning of the modified Triple-DES or DES, be variable.

Another relevant point is that by starting with a variable permutation we obtain a property similar to *whitening* [7], which the most recent encryption algorithms, such as AES [11], [12] posses.

In order to illustrate the latter, a n_{EP} different from n_{IP} is presented:

$$n_{EP} = 98976714136546768494654687987987977777777779789465465465$$
$$4132123156465498794865413515902 14$$

The permutation corresponding to such number, EP, is presented in table 1.

Let us now discuss how the complexity of the modified DES and Triple-DES cryptosystems is increased when proposing a variable initial permutation. Given that the modified DES and Triple-DES cryptosystems begin their cipher process with strings of 64 bits in length, it follows that there are 2^{64} different possibilities of input and 2^{64} different possibilities of output [13]. However, if these output possibilities are seen as an array, there are $(2^{64})!$ different output arrays.

In this point some notation specifications will be done. Let the encryption processes of modified DES and Triple DES be denoted as $e(X)_{K,P}$. In this

EP							
49	59	7	12	28	16	43	34
45	41	36	10	17	27	4	25
23	22	63	55	57	44	46	40
6	24	5	42	37	20	47	11
19	50	53	21	8	2	33	13
39	29	56	61	54	15	9	18
31	1	35	60	30	58	26	51
32	62	3	48	52	38	0	14

Table 1: Example permutation EP

case, X is a 64 bits string representing the plain text; K is the key, which can be a 64 or 128 bits string depending on which algorithm is being used. P is a permutation of the positions of a string 64 bits long. Also, $e(X)_{K,P}$ has associated an output array for some K,P given, which is of this form:

$$e(00\ldots0)_{K,P} = Y_0, \ldots, e(11\ldots1)_{K,P} = Y_{2^{64}-1}$$

It is not complicated to show that the modified DES and Triple-DES algorithms define a one-to-one function, given that each round is a one-to-one function. The latter is easily proved by *reductio ad absurdum*. Thus, the modified DES and Triple-DES algorithms can be seen as the composition of either 16 or 48 one-to-one functions, respectively.

Now, in order to know whether each of the $64! \cong 2^{300}$ possible permutations has a different output array —which would prove complexity to be al least 2^{300}—, the following theorem is presented.

Theorem 3.1 *Given two permutations $P_1 \neq P_2$ on the positions of 64 bits strings, and the fact that the key K of either 64 or 128 bits of length is considered to be fixed, then functions $e(X)_{K,P_1}$, $e(X)_{K,P_2}$ define different arrays; that is, they cipher in a different manner.*

Proof. If $P_1 \neq P_2$, then there exists at least one 64 bits string, say X^0, such that $P_1(X^0) \neq P_2(X^0)$. This is true, since if $P_1 \neq P_2$ it means that there are at least 2 positions where P_1 and P_2 are different.

Suppose there are $i_1, i_2, \ldots, i_k$ different positions with $2 \leq k \leq 64$. Now, if a 64 bits long string is built by making $x_{i_1} = 0, x_{i_2} = 1, \ldots, x_{i_k} = 0$ or $x_{i_k} = 1$, depending on whether k is odd or even. Then, the resulting string is:

$$C_1 = x_1, \ldots, x_{i_1-1}, 0, x_{i_1+1}, \ldots, x_{i_2-1}, 1, x_{i_2+1}, \ldots, x_{i_k-1}, 0 \text{ or } 1, x_{i_k+1}, \ldots, x_{64}$$

Since $P_1 \neq P_2$ in positions $i_1, i_2, \ldots, i_k$, than values $x_{i_1} = 0, x_{i_2} = 1, \ldots, x_{i_k} = 0$ or $x_{i_k} = 1$ are ordered in a different manner depending on P_1 or P_2 being applied to string C_1.

Now, let us denote $P_1(X^0) = TP_1, P_2(X^0) = TP_2$ and $e^*(X)_K$ as the encryption process followed after applying the initial permutation P; that is, $e^*(X)_K$ has as input arguments either TP_1 or TP_2. Also, since $e^*(X)_K$ is built by means of succesive one-to-one functions, it follows that $e^*(TP_1)_K \neq e^*(TP_2)_K$; remember that K is fixed. Therefore, it is concluded that the arrays associated to $e(X)_{K,P_1}$ and $e(X)_{K,P_2}$ are different, since there is at least one position different on both arrays; this means that they cipher differently. ■

Notice that $2^{300} \ll (2^{64})!$, and so it can be said that all permutations act as different keys, since they will give different output arrays. In this sense, it can be said that there are at least 2^{300} possible keys, which is greatly superior to 2^{56} for DES and 2^{112} for Triple-DES.

4 Conclusions

After the analysis of the preceding sections, it can be observed that an immediate application of isomorphism $I_m : N_m \longrightarrow \Pi_m$ is to consider permutations as keys in symmetrical cryptosystems of the Substitution Permutation Network kind; more specifically, its application to the DES and Triple-DES cryptosystems —with some minor modifications—was shown. The latter allows a computational complexity increase in said cryptosystems, such as in the case of the modifed Triple-DES which goes from 2^{112} to 2^{300}. On the other hand, in order to execute the differential and linear attacks on the DES cryptosystem, it is necessary to reach the boxes [2],[3]. However, if the initial permutation is variable, these kind of procedures cannot take place in the first round.

In general, it can be said that by using permutations as keys, it is possible to increase the computational complexity of any Substitution Permutation Network cryptosystem, against a brute-force attack, and differential and linear cryptanalysis.

On the other hand, it is noteworthy to remark that operating permutations as numbers is convenient, given that asymmetrical criptosystems such as RSA [14] cipher numbers, thus enabling this result to work on a Public Key Infrastructure (PKI) schema [15].

Also, it was shown how to build a field on the set of permutations Π_m^p, where $p < m!$ is a prime number.

Acknowledgements. The authors would like to thanks the Instituto Politécnico Nacional (Secretaría Académica, COFAA, SIP, CIDETEC, and CIC), Also, this work was partially supported by CONACYT and SNI-SEP, México.

References

[1] Grabbe J. Orlin, 2003, "Data Encryption Standard: The DES algorithm illustrated", Laissez faire City time, vol. 2, no 28.

[2] Biham E. and Shamir A., 1993, "Differential cryptanalysis of the full 16-round DES", Lecturer Notes in Computer Science.

[3] Matsui M, 1994, "Linear Cryptanalysis for DES cipher", Lecture Notes in Computer Science.

[4] Herstein I.N., 1986, Álgebra Abstracta, Grupo Editorial Iberoamérica, pp. 22 y 11.

[5] Silva García V.M. et al, 2007, "Bijective Function with Domain in N and Image in the Set of Permutations: An Application to Cryptography", International Journal of Computer Science and Network Security, Vol. 7 No 4, pp. 117.

[6] Koblitz M., 1987, A Course in Number Theory and Cryptography, Springer-Verlag, pp. 53-80, New York Inc.

[7] Douglas R. Stinson, 2006, CRYPTOGRAPHY: Theory and practice, CHAPMAN & HALL/ CRC Press, second edition, pp. 73-116.

[8] Douglas R. Stinson, 1995, CRYPTOGRAPHY: Theory and practice, CRC Press, first edition, pp. 89-98.

[9] Sorking A., 1980, LUCIFER: A cryptographic algorithm, Cryptología 8, pp 22-35.

[10] FIPS 46-2, 1977, Data Encryption Standard Federal Information Processing Standards Publication 46-2.

[11] J. Daemen and V. Rijmen, 1999, AES Proposal : Rijndael, AES algorithm Submission, FIPS 197

[12] FIPS 197, 2001, Advanced Encryption Standard Federal Information Processing Standards Publication 197.

[13] Rosen K., 2003, Discrete Mathematics and its Applications, Mc. Graw Hill, fifth edition.

[14] R.L. Rivest, A. Shamir and L. Adleman, 1978, A method for obtaining digital signatures and Public Key cryptosystems, Communications of the ACM, pp. 120-126.

[15] Adams, C., Lloyd, S., 2002, Understanding PKI: Concepts, Standards, and Deployment Considerations, Addison-Wesley Longman Publishing Co., Inc. Boston, MA, USA. ISBN: 0672323915.

Received: October, 2009