Journal of Mathematical Sciences: Advances and Applications
Volume 1, Number 3, 2008, Pages 563-577

AN MCA-BASED ALGORITHM TO ENCRYPT TEXT DATA

A. MARTÍN DEL REY

Department of Applied Mathematics
E.P.S de Ávila
Universidad de Salamanca
C/ Hornos Caleros 50
05003-Ávila, Spain
e-mail: delrey@usal.es

Abstract

In this work, a new cryptographic protocol to encrypt text data is proposed. It is based on the use of a particular type of discrete dynamical system called reversible memory cellular automata. It is shown to be secure against the most important cryptanalytic attacks such as brute-force attack, key sensitivity attack, ciphertext-only attack, known-plaintext attack and chosen-plaintext attack.

1. Introduction

As is well-known, confidentiality is mandatory for majority of network applications. Achieving information security in our electronic society requires a vast array of technical means, which are provided through cryptography.

2000 Mathematics Subject Classification: 68Q80, 94A60, 68P25.

Keywords and phrases: cryptography, memory cellular automata, information processing, secret-key cryptosystems.

This work has been supported by Ministerio de Ciencia e Innovación (Spain) under grant MTM2008-02773.

Received October 21, 2008

Specifically, cryptography is the study of mathematical techniques related to aspects of information security, that is, its main goal is to provide security for communications and data storage systems. Consequently, the four basic objectives of cryptography are: Confidentiality, data integrity, entity authentication, and data origin authentication.

This paper deals with confidentiality. Roughly speaking, our main goal is to provide an efficient mathematical algorithm to enable two people to communicate over an insecure channel in such a way that an opponent cannot understand what is being said. To get this objective, the original message to be sent (plaintext) must be modified, according to an algorithm (cryptosystem) and some parameters called keys, to obtain the encrypted message (ciphertext). If the keys are only known for the sender and the receiver of the message, the cryptosystem is called symmetric or secret-key cryptosystem, and its security mainly relies on keeping secret the key used. On the other hand, if the key to encrypt the message (public key) is publicly known, whereas the key to decrypt the message (private key) is only known by the receiver, the cryptosystem is called *asymmetric* or *public-key cryptosystem* (see, for example, [9, 13]).

In order to design a robust cryptosystem, one should study if there exist any weaknesses in the protocol. The techniques used to compromise cryptosystems are referred to as *cryptanalytic attacks*. Specifically, the main objectives of such techniques are to obtain the keys of a cryptosystem or the plaintext from the ciphertext. The most important attacks are brute-force attacks, ciphertext-only attack, known-plaintext attack and chosen-plaintext attack.

In this work we are interested in the use of very simple models of computation, called *reversible memory cellular automata*, in order to design a secret-key cryptosystem for 128 bitlength block messages. Basically, memory cellular automata are delay discrete dynamical systems formed by a finite number of identical objects called *cells*, which are endowed with a state that changes at every discrete step of time according to a deterministic rule, whose variables are the states of a set of cells at previous time steps (see, for example, [1]).

The use of cellular automata to design cryptographic protocols goes back to middle eighties when Wolfram proposed the cellular automaton with rule number 30 as a pseudorandom bit generator for cryptographic purposes (see, [14, 15]). Since then, many cryptosystems based on cellular automata have been proposed (see, for example, [2, 3, 4, 5, 6, 7, 8, 10, 11, 12]).

The rest of the paper is organized as follows: In Section 2, the basic theory about memory cellular automata is introduced; in Section 3 the algorithm to encrypt text data is presented; an example is shown in Section 4 and the security analysis of the protocol is given in Section 5. Finally, the conclusions and further work are introduced in Section 6.

2. Overview of Cellular Automata

A one-dimensional cellular automaton (CA for short) is a special class of discrete dynamical system, which is formed by a finite one-dimensional array of n identical objects called *cells*: $\langle 1 \rangle, ..., \langle n \rangle$. Each one of them can assume a state from a finite state set S. If the number of possible states is k, then $S = \mathbb{Z}_k$. The state of the i-th cell at time t is denoted by $s_i^{(t)}$.

The CA evolves deterministically in discrete time steps, changing the states of all cells according to a local transition function, $f : S^m \to S$. The updated state of each cell depends on the m variables of f, which are the states at previous time steps of a set of cells, including the cell itself, and called its *neighborhood*. The set of indices of the CA is the ordered finite subset $V \subset \mathbb{Z}, |V| = m$, such that for the i-th cell, its neighborhood, V_i, is the set of m cells given by $V_i = \{\langle i + \alpha \rangle : \alpha \in V\}$.

In this work, r-th order symmetric neighborhoods are considered; that is $V = \{-r, -r + 1, ..., -1, 0, 1, ..., r - 1, r\}$. As a consequence, the evolution of the state of the cell $\langle i \rangle$ is given by

$$s_i^{(t+1)} = f(s_{i-r}^{(t)}, ..., s_i^{(t)}, ..., s_{i+r}^{(t)}), \tag{1}$$

or, equivalently, $s_i^{(t+1)} = f(V_i^{(t)})$, where $V_i^{(t)}$ stands for the states of the

neighbour cells of $\langle i \rangle$ at time t. The vector $C^{(t)} = (s_1^{(t)}, ..., s_n^{(t)}) \in \mathbb{Z}_k^n$, is called the *configuration* at time t of the CA. The set of all configurations of a CA is denoted by $\mathcal{C}$, and consequently $|\mathcal{C}| = k^n$. The configuration $C^{(0)}$ is called the *initial configuration* of the CA and the sequence $\{C^{(0)}, ..., C^{(t)}\}$ is called the t-th *order evolution* of the CA (starting from $C^{(0)}$).

As the number of cells is finite, boundary conditions must be considered in order to assure the well-defined dynamics of the CA. In this paper, periodic boundary conditions are taken: If $i \equiv j(\mathrm{mod}\, n)$, then $s_i^{(t)} = s_j^{(t)}$.

Moreover, the global function of the CA is a linear map, $\Phi : \mathcal{C} \to \mathcal{C}$, that yields the configuration at the next time step during the evolution of the cellular automaton, that is, $\Phi(C^{(t)}) = C^{(t+1)}$. If Φ is bijective then the CA is called *reversible* (RCA for short) and the evolution backwards is possible by means of the inverse CA whose global transition function is Φ^{-1}.

The standard paradigm for cellular automata considers that the state of every cell at time $t+1$ depends on the state of its neighbor cells at time t. Nevertheless, one can consider cellular automata for which the state of every cell at time $t+1$ not only depends on the states of the neighbor cells at time t, but also on their states at previous time steps: $t-1$, $t-2$, etc. This is the main feature of memory cellular automata, MCA for short (see [1]). Specifically, a k-th order MCA is defined by a global transition function given by

$$\Phi : \mathcal{C} \times \overset{k}{\cdots} \mathcal{C} \to \mathcal{C},\ \Phi(C^{(t)}, C^{(t-1)}, ..., C^{(t-k+1)}) = C^{(t+1)}. \tag{2}$$

3. The Cryptosystem

In this section, the proposed secret-key cryptosystem to encrypt text data is introduced.

Basically, it consists of a 128-bit block cipher based on the use of memory cellular automata. Its secret key is a pair (K, K'), where the size of each subkey is of 128 bits (the total bitlength of the key is 256). The first subkey, K, is involved in the MCA, whereas the second subkey, K', is obtained from the protocol, and, as a consequence, it can be considered as a session key.

The proposed cryptographic algorithm is formed by three phases: The setup phase, where the sender chooses the memory cellular automata and the first subkey to be used in the protocol, the encryption phase, where the sender encrypts the plaintext using the protocol and the second subkey is obtained, and finally, the decryption phase, where the receiver decrypts the ciphertext.

3.1. The setup phase

In this phase, the sender divides the plaintext, M, into k blocks of $n = 128$ bits, so that, $M = M_1 \| M_2 \| \cdots \| M_{k-1} \| M_k$, where $\|$ denotes the concatenation operator. Note that each block can be interpreted as a configuration of a CA of $n = 128$ cells with state set $S = \mathbb{Z}_2$ as follows: If

$$M_i = m_{i,1} \cdots m_{i,128}, \tag{3}$$

where $m_{i,j}$ is the j-th bit of the block M_i, then the configuration associated is

$$C = (m_{i,1}, ..., m_{i,128}) \in \mathbb{Z}_2^{128}. \tag{4}$$

Moreover, the sender selects a secret key K of 128-bitlength, and generates a sequence of $(2r+1)k$ bits, where r is the order of the neighborhood, by means of a pseudorandom bit generator (see [9]) taking K as the seed:

$$b_1^{(1)}, ..., b_{2r+1}^{(1)}, ..., b_1^{(k)}, ..., b_{2r+1}^{(k)}. \tag{5}$$

Subsequently, the sender chooses k MCA with $S = \mathbb{Z}_2$ and $n = 128$ cells. The j-th MCA, $1 \le j \le k$, is a $(j+1)$-th order MCA whose global

function is as follows:

$$C^{(t+1)} = \Phi_j(C^{(t)}, ..., C^{(t-j)}) = \Psi_j(C^{(t)}, ..., C^{(t-j+1)}) + C^{(t-j)}. \quad (6)$$

Moreover, the local transition function is given by the following expression:

$$s_i^{(t+1)} = f_1(V_i^{(t)}) + \cdots + f_j(V_i^{(t-j+1)}) + s_i^{(t-j)}, \pmod 2 \quad (7)$$

with $1 \le i \le 128$, where

$$f_m(V_i^{(t-m+1)}) = \sum_{t=-r}^{r} b_{r+1-l}^{(m)} s_{i+l}^{(t-m+1)} \pmod 2,\ 1 \le m \le j. \quad (8)$$

For the sake of simplicity, this $(j+1)$-th order MCA is denoted by $\mathcal{A}_j$, and, obviously, it is reversible (see [1]). Its inverse MCA, $\mathcal{A}_j^{-1}$, is defined by the following local transition function:

$$s_i^{(t+1)} = -f_j(V_i^{(t)}) - \cdots - f_1(V_i^{(t-j+1)}) + s_i^{(t-j)}, \pmod 2 \quad (9)$$

with $1 \le i \le n = 128$.

Finally, the sender chooses a sequence, $\{N_1, ..., N_k\}$, of k secret integer numbers such that $N_i > N_{i-1} + i$ for every i, $1 \le i \le k$, and $N_1 > 2$.

3.2. The encryption phase

In this phase, the sender encrypts the text $M = M_1 \| M_2 \| \cdots \| M_{k-1} \| M_k$ using the MCA defined in the setup phase.

The algorithm is composed by k steps. In the i-th step, the evolution of the $(i+1)$-th order MCA, $\mathcal{A}_i$, is computed starting from the i configurations obtained from the $(i-1)$-th step and a new configuration given for the i-th block M_i. Specifically, the steps are as follows:

Step 1. Let $C^{(0)}$, $C^{(1)}$, be the initial configurations of the 2-th order MCA, $\mathcal{A}_1$, such that $C^{(1)} = M_1$, and $C^{(0)}$ is a truly random sequence of bits which must be generated by a device without the intervention of the sender. Then by iterating N_1 times the MCA, $\mathcal{A}_1$, the configurations

$$C^{(N_1-1)}, C^{(N_1)} \tag{10}$$

are obtained. When the protocol is finished, the configuration $C^{(0)}$ can be destroyed.

Step 2. Let us consider the two configurations obtained in the previous step and set $C^{(N_1+1)} = M_2$. Iterating N_2 times the 3-th order MCA, $\mathcal{A}_2$, the following three configurations are obtained:

$$C^{(N_2-2)}, C^{(N_2-1)}, C^{(N_2)}, \tag{11}$$

…

Step i. Let us consider the i configurations obtained from the step $i-1$:

$$C^{(N_{i-1}-i+1)}, ..., C^{(N_{i-1})}, \tag{12}$$

and set $C^{(N_{i-1}+1)} = M_i$. Iterating N_i times the $(i+1)$-th order MCA, $\mathcal{A}_i$, the following $i+1$ configurations are obtained:

$$C^{(N_i-i)}, C^{(N_i-i+1)}, ..., C^{(N_i-1)}, C^{(N_i)}. \tag{13}$$

…

Step k. Let us consider the k configurations obtained from the step $k-1$:

$$C^{(N_{k-1}-k+1)}, C^{(N_{k-1}-k+2)}, ..., C^{(N_{k-1})}, \tag{14}$$

and set $C^{(N_{k-1}+1)} = M_k$. By simply computing N_k iterations of the $(k+1)$-th order MCA, $\mathcal{A}_k$, the following $k+1$ configurations are obtained:

$$C^{(N_k-k)},\ C^{(N_k-k+1)},\ ...,\ C^{(N_k-1)},\ C^{(N_k)}. \tag{15}$$

Consequently, the output of the MCA-based algorithm is a set of $k+1$ configurations, each one of size 128 bits, that is, $128(k+1)$ bits are obtained.

The ciphertext, $\overline{M}$, is formed by the concatenation of the configurations $C^{(N_k-k+1)},\ ...,\ C^{(N_k)}$:

$$\overline{M} = \overline{M}_1 \| \cdots \| \overline{M}_k, \tag{16}$$

where $\overline{M}_i$ stands for the configuration $C^{(N_k-k+i)}$, $1 \le i \le k$.

Moreover, the second subkey is given by $K' = C^{(N_k-k)}$. Note that it is a session key which depends on the plaintext to be cipher.

3.3. Decryption phase

In this phase, the receiver decrypts the ciphertext using the inverse MCA defined in the setup phase.

To decrypt a ciphertext, $\overline{M} = \overline{M}_1 \| \cdots \| \overline{M}_k$, the receiver must consider the k MCA, $\mathcal{A}_1^{-1},\ ...,\ \mathcal{A}_k^{-1}$, defined above. Then, taking into account the configurations

$$\widetilde{C}^{(0)} = \overline{M}_k,\ ...,\ \widetilde{C}^{(k-1)} = \overline{M}_1,\ \widetilde{C}^{(k)} = K', \tag{17}$$

and applying the following k-steps algorithm, the receiver can obtain the plaintext M. The algorithm to decrypt the message is as follows:

Step 1. Starting from the $k+1$ initial configurations given in (17), and by computing $N_k - N_{k-1} - 1$ iterations of $\mathcal{A}_k^{-1}$, the configurations

$$\widetilde{C}^{(N_k-N_{k-1}-1)} = C^{(N_{k-1}+1)},\ ...,\ \widetilde{C}^{(N_k-N_{k-1}+k-1)} = C^{(N_{k-1}-k+1)}, \tag{18}$$

are obtained. The configuration $\widetilde{C}^{(N_k-N_{k-1}-1)} = C^{(N_{k-1}+1)} = M_k$ gives the k-th block of the plaintext.

Step 2. Let us consider the k configurations obtained in the previous step:

$$\widetilde{C}^{(N_k - N_{k-1})}, \ldots, \widetilde{C}^{(N_k - N_{k-1} + k - 1)}. \tag{19}$$

If $N_{k-1} - N_{k-2} - 1$ iterations of the MCA $\mathcal{A}_{k-1}^{-1}$ are computed starting from them, the following k configurations are obtained:

$$\widetilde{C}^{(N_k - N_{k-2} - 1)}, \widetilde{C}^{(N_k - N_{k-2})}, \ldots, \widetilde{C}^{(N_k - N_{k-2} + k - 2)}. \tag{20}$$

Obviously, $\widetilde{C}^{(N_k - N_{k-2} - 1)} = M_{k-1}$, and the receiver must use the remaining $k - 1$ configurations, $\widetilde{C}^{(N_k - N_{k-2})}, \ldots, \widetilde{C}^{(N_k - N_{k-2} + k - 2)}$, in the next step.

…

Step k. Let us consider the configurations obtained from the $(k - 1)$-th step:

$$\widetilde{C}^{(N_k - N_1 - 1)}, \widetilde{C}^{(N_k - N_1)}, \widetilde{C}^{(N_k - N_1 + 1)}. \tag{21}$$

The first one is the second block of the plain text, M_2, whereas if the receiver computes $N_1 - 1$ iterations of $\mathcal{A}_1^{-1}$ using $\widetilde{C}^{(N_k - N_1)}, \widetilde{C}^{(N_k - N_1 + 1)}$ as the initial configurations, it yields $\widetilde{C}^{(N_k - 1)}, \widetilde{C}^{(N_k)}$. The last block of the plaintext to be obtained is $\widetilde{C}^{(N_k - 1)} = M_1$.

4. An Example

For the sake of simplicity, let us consider the 1024-bitlength plaintext M, which can be divided into the following $k = 8$ blocks of 128 bits:

$$C^{(1)} \equiv f9d9e40e77681da9c864f13bcc64b933,$$

$$C^{(N_1+1)} \equiv 1a9ed3c271558b00865f21684a13c83,$$

$$C^{(N_2+1)} \equiv c6e1acf69a4c6d1fcb0a115fcdbfc796,$$

$$C^{(N_3+1)} \equiv 5d60497a02ae8c4ba2f71aec35377521,$$

$$C^{(N_4+1)} \equiv ccfee680c6d25f2b079570b499c2611b,$$

$$C^{(N_5+1)} \equiv b9cd827ab5405cea78c5e94b43e6ecc,$$

$$C^{(N_6+1)} \equiv fc587f2414399cb3c270f195f6e8c5c7,$$

$$C^{(N_7+1)} \equiv 32ec704baca0a0a0fa748eac2c5c8aa5. \tag{22}$$

Note that the blocks are given in terms of their hexadecimal codes. Also suppose that the random initial configuration is given by:

$$C^{(0)} \equiv 259d4c71449e7942e8df3eda3d7a03b0. \tag{23}$$

Moreover, set $r = 5$ as the radius of the symmetric neighborhoods,

$$N_1 = 78,\ N_2 = 129,\ N_3 = 175,\ N_4 = 243,$$

$$N_5 = 317,\ N_6 = 340,\ N_7 = 432,\ N_8 = 499, \tag{24}$$

and

$$K = 3ec48d3ff788d150c5133fce4533e4c0. \tag{25}$$

Then, using the MCA-based protocol introduced in Section 3, the ciphertext is given by the following eight configurations:

$$C^{(492)} \equiv 951204262404f740b8af61f111a8de8d,$$

$$C^{(493)} \equiv c6fe5fe4e419ba02e031d4102d1844e,$$

$$C^{(494)} \equiv 12ec43bb1cdfccbfd48e9adfb9954f02,$$

$$C^{(495)} \equiv cca7aca9d8f77810978d48ead1b413f7,$$

$$C^{(496)} \equiv a55f9528d694ba475d3afafd65263734,$$

$$C^{(497)} \equiv 2a6b1f04121cdda9688d6f84df275b0f,$$

$$C^{(498)} \equiv 79b320a2bae3f303fb3f969f0ab9337,$$

$$C^{(499)} \equiv 7c14e17c8c7d2b4d91faec79b622b603. \tag{26}$$

Furthermore, the subkey K' is

$$C^{(491)} \equiv K' = b71c9233f91b39ed89a04f25507520da. \tag{27}$$

To recover the plaintext from this ciphertext and the subkey K', it is enough to follow the steps pointed out in Subsection 3.3.

5. Security Analysis

Five cryptanalytic attacks for evaluating the security of the cryptosystem proposed are studied: brute-force attack, key sensitivity test, ciphertext-only attack, known-plaintext attack and chosen-plaintext attack. Each of them assumes that the cryptanalyst has the complete knowledge of the cryptosystem used but no information about the secret key.

As the bitlength of the secret key is of 256 bits, then the order of the key space is 1.15792×10^{77}, and consequently, it makes the brute-force attack infeasible.

Moreover, the proposed cryptosystem also passes the key sensitivity test: Let us consider the plain-text M and the secret key (K, K') mentioned in the above example. If we modified only one bit in the secret subkey K, a new subkey, Q, is obtained:

$$Q = 3ec48d3ff788d151c5133fce4533e4c0. \tag{28}$$

If the plaintext is ciphered using the new subkey Q, the following ciphertext is obtained:

$$C^{(492)} \equiv 9e2857c25587e242f9ec9d958a955c1e,$$

$$C^{(493)} \equiv 686f554d619886dccaa6a813d7d0cc4c,$$

$$C^{(494)} \equiv 7867fa4a32ec97c8f0ed59b51ee5cdfb,$$

$$C^{(495)} \equiv 1014a4c3993b0b852271b1eb2c1bbb1a,$$

$$C^{(496)} \equiv 8f74ed1dc5b9ed6c24b92e95737477ca,$$

$$C^{(497)} \equiv a605539d01a841226e0fbbf8c715f52a,$$

$$C^{(498)} \equiv e94defcd5254cc7365b8d08c097a45ed,$$

$$C^{(499)} \equiv f278f3fc53e8d83474e18d67f74646db. \tag{29}$$

Note that the discrepancies (that is, the number of different bits) between this ciphertext (obtained from the modified key Q) and the correct cryptogram is of 49.12% of total bits. Moreover, the corresponding subkey Q' obtained is very different from K'.

Furthermore, if we try to recover the plaintext using the secret key (Q, K'), and the original ciphertext obtained from (K, K'), the recovered plaintext is the following:

$$C^{(1)} \equiv 6608fb20899d96397dce91ebf628aa1,$$

$$C^{(N_1+1)} \equiv ce5374d1e6515569634acaf80a427ac9,$$

$$C^{(N_2+1)} \equiv 9784ef72f0243bccffe92362236a012d,$$

$$C^{(N_3+1)} \equiv 28e3b6070a65a2513a9b727bf5470b4f,$$

$$C^{(N_4+1)} \equiv 4b150581d33a1f96cc07b3e64f91551a,$$

$$C^{(N_5+1)} \equiv c988daa09f9b7fb228009073bc0d1741,$$

$$C^{(N_6+1)} \equiv b14c25ebd7969f4874f12330892cc440,$$

$$C^{(N_7+1)} \equiv 642c3489b5b09d8ad3a8fc9c1d7e2f1a. \tag{30}$$

The results state that the percentage of different bits between the two plaintexts is 49.71%.

Finally, if we try to recover the plaintext using the same subkey K, and a slightly different subkey K',

$$b71c9233f91b39ed89a44f25507520da, \tag{31}$$

the plaintext obtained is:

$$C^{(1)} \equiv 658d472eb4b37a189219fa27ddadc3eb,$$

$$C^{(N_1+1)} \equiv ce41f8ddc6ce56a5dd4f6d1a1162901b,$$

$$C^{(N_2+1)} \equiv abf2b0691dc81e1f26e9d18746a72983,$$

$$C^{(N_3+1)} \equiv a0ae83282dcdd598536b9e57cbbc714a,$$

$$C^{(N_4+1)} \equiv a4374fd3f48a074fd9cbace10db485bd,$$

$$C^{(N_5+1)} \equiv 80836b0a12a668dc7b734110c471e751,$$

$$C^{(N_6+1)} \equiv dcc0372ad4afbb73f714341df0bee50d,$$

$$C^{(N_7+1)} \equiv afcc65b15911ee4440c12d1544a3be23. \quad (32)$$

As a simple calculus shows, the percentage of different bits between the two plaintext is 49.32%.

In the ciphertext-only attack, the secret key must be determined solely from an intercepted ciphertext:

$$C^{(N_k-k+1)}, \ldots, C^{(N_k)}. \quad (33)$$

Consequently, both subkeys, K and K', are unknown for the cryptanalyst. In this case, to obtain only the k-th block of the plaintext, $M_k = C^{(N_{k-1}+1)} = \widetilde{C}^{(N_k-N_{k-1}-1)}$, a non-linear system of 128 equations with $128 + (2r + 1)$ unknown variables: $b_1^{(k)}, \ldots, b_{2r+1}^{(k)}$, and the bits of K', must be solved, which is impossible taking into account the conditions of the cryptosystem.

In the known-plaintext attack, the secret key (K, K') must be obtained starting from a pair of a plaintext and its corresponding ciphertext. The difficulty of this attack is the same as the previous one, and consequently, the cryptosystem is also secure against known-plaintext attack.

Finally, in the chosen-plaintext attack, the cryptanalyst is able to choose a plaintext and obtain its corresponding ciphertext. In this way the better attack consists of taking as a plaintext the following: $M = (0, \overset{(128\cdot k)}{\ldots}, 0)$; consequently $C^{(N_{i-1}+1)} = (0, \overset{(128)}{\ldots}, 0)$. Nevertheless,

as the configuration $C^{(0)}$ is randomly computed, the cryptanalyst must solve k non-linear systems of 128 equations with $128 + (2r + 1)k$ unknown. As the integer numbers N_1, ..., N_k, remain unknown to the cryptanalyst, it is impossible for him/her to know the number of terms of such non-linear systems. Consequently, the protocol is secure against chosen-plaintext attack.

6. Conclusions and Further Work

In this work a novel secret-key cryptosystem for text data has been proposed. It is based on the use of memory cellular automata. Specifically, the text to be encrypted is divided into k blocks of 128-bitlength, and it is encrypted by means of a random initial configuration and k reversible memory cellular automata involving a 128-bitlength key. As a consequence its corresponding ciphertext (of the same size) is obtained jointly with another 128-bitlength subkey, necessary to recover the original plaintext.

The protocol is shown to be secure against the most important cryptanalytic attacks such as brute-force attack, ciphertext-only attack, known-plaintext attack and chosen-plaintext attack. It is also shown to pass the key sensitivity test.

Further work aimed at designing a similar cryptosystem in which only the 128 bits of the first subkey, K, determine the local transition functions, instead of the pseudorandom sequence of bits generated using these bits as a seed.

Moreover, the proposed scheme can be used to design an algorithm to encrypt images, by using two-dimensional memory cellular automata.

References

[1] R. Alonso-Sanz and M. Martín, One-dimensional cellular automata with memory: Patterns from a single site seed, Internat. J. Bifur. Chaos 12 (2002), 205-226.

[2] G. Álvarez, A. Hernández Encinas, L. Hernández Encinas and A. Martín del Rey, A secure scheme to share secret color images, Comput. Phys. Comm. 173 (2005), 9-16.

[3] F. Bao, Cryptanalysis of a partially known cellular automata cryptosystem, IEEE Trans. Comput. 53 (2004), 1493-1497.

[4] A. Fúster-Sabater and D. de la Guía-Martínez, Cellular automata applications to the linearization of stream cipher generators, Proceedings of ACRI 2004, Lect. Notes Comput. Sci. 3305 (2004), 612-621.

[5] P. Guan, Cellular automaton public-key cryptosystem, Complex Systems. 1 (1987), 51-57.

[6] H. A. Gutowitz, Cryptography with dynamical systems, Proceedings of the NATO Advanced Study Institute: Cellular Automata and Cooperative Systems (1993), pp. 237-274.

[7] A. Martín del Rey, J. Pereira Mateus and G. Rodríguez Sánchez, A secret sharing scheme based on cellular automata, Appl. Math. Comput. 170 (2005), 1356-1364.

[8] W. Meier and O. Staffelbach, Analysis of pseudorandom sequences generated by cellular automata, Advances in Cryptology: Euro Crypt'91 Proceedings, Lect. Notes Comput. Sci. 547 (1991), 186-189.

[9] A. Menezes, P. van Oorschot and S. Vanstone, Handbook of Applied Cryptography, CRC Press, Boca Raton, 1997.

[10] S. Nandi, B. K. Kar and P. P. Chaudhuri, Theory and applications of cellular automata in cryptography, IEEE Trans. Comput. 43 (1994), 1346-1357.

[11] M. Seredynski and P. Bouvry, Block encryption using reversible cellular automata, Proceedings of ACRI 2004, Lect. Notes Comput. Sci. 3305 (2004), 785-792.

[12] M. Seredynski, P. Bouvry and A. Zomaya, Cellular automata computations and secret key cryptography, Parallel Comput. 30 (2004), 753-766.

[13] D. R. Stinson, Cryptography: Theory and Practice, Second Edition, Chapman & Hall/CRC, Boca Raton, 2002.

[14] S. Wolfram, Random sequence generation by cellular automata, Adv. Appl. Math. 7 (1986), 123-169.

[15] S. Wolfram, Cryptography with cellular automata, Advances in Cryptology: Crypto'85 Proceedings, Lect. Notes Comput. Sci. 218 (1986), 429-432.

■