International Journal of Computer Networks and Communications Security

VOL. 2, NO. 3, MARCH 2014, 119–124
Available online at: www.ijcncs.org
ISSN 2308-9830

eLoBa Chaotic Keystream Cipher: Implementation, Analysis and Performance

RUI MIGUEL SILVA[1], RUI GUSTAVO CRESPO[2]

[1] Polytechnic Institute of Beja (IPBeja)/Lab UbiNET/INESC-ID, Department of Engineering, Beja, Portugal

[2] Technical University of Lisbon (UTL)/INESC-ID, Department of Electrical and Computer Engineering, Lisbon, Portugal

E-mail: [1]ruisilva@acm.org, [2](Posthumous Co-Author)

ABSTRACT

Many computer-based security services are required by modern society, such as cipher/decipher, authentication and integrity. Security services are implemented by cryptographic systems, such as symmetric (block and keystream) and asymmetric. We focus our attention to a special sort of keystream cryptographic systems, based on chaotic systems. In our paper, we describe eLoBa architecture, based on Lorenz chaotic system. We also analyze one important security issue of keystream cryptographic systems, the cycle length. Finally, we compare eLoBa against several cryptographic systems, for its performance on a authentication service implementation on a resource constraint smart-card.

Keywords: *Chaotic Cipher, Smart-Card Authentication, Keystream Cipher, Security and Privacy, Cryptography.*

1 INTRODUCTION

Cryptography for resource constrained devices is a challenge with increasing interest due to the rise of Internet of Thing, the applications of Wireless Sensor Networks, or the massive usage of smart-cards, among other areas. However, most of the cryptographic algorithms demand for computational resources that could not be available in such devices. One important cryptographic service is Authentication and chaos cryptography is low resource consumer. In this paper we merge this two facts and evaluate the results in real world smart-card implementation.

2 CHAOTIC KEYSTREAM CIPHER

In this section we briefly present chaotic ciphers motivation and the architecture of a specific one targeted for resources constrained devices, named eLoBa [1].

2.1 Chaotic Ciphers

Since 1890, when the French mathematician and physician Henri Poincaré had make reference to chaotic behavior about the movement of “n free bodies”, as the solar system, that Chaos Theory have evolved and been subject to several applications. The basis of Chaos Theory is that small disturbances in the initial conditions (of a chaotic system), leads to an evolution with exponential divergent grow. This kind of behavior is an expected property of cryptographic systems. Considering the use of a chaotic system as the base for a cipher and using its initial conditions as key, a small disturbance in the initial conditions could be the change of a single bit in the key, would lead to a completely different cryptogram. This is an acceptable thought and indeed gives rise to several researches around the possible and secure use of chaotic systems as base for cryptographic systems.

However there are a structural difference between chaotic systems and cryptographic systems, the first ones have a continuous essence meanwhile the second ones have discrete a discrete essence. So, there are two approaches for the use of chaotic systems as base for cryptographic systems: using chaotic systems as continuous systems; or using discretized chaotic systems, which could be called "rounded chaos".

The use of continuous chaotic systems as base for construction of cryptographic systems begins in 1990 [2] with the use of the analogical signal of the chaotic system to support for data transmission. This approach has been used in several works, like [3][4][5][6][7][8][9][10].

The use of discretized chaotic systems as base for construction of cryptographic system could be divided in two classes of approaches: one that uses chaotic systems as pseudo-random generators; another that uses chaotic systems as increment and decrement systems for cryptographic operations. The second class of this approach started in 1990 [11] was cracked in 1991 [12], being improved in following works as [13][14] or [15], however its acceptance was never to much relevant.

The first class of this approach, using chaotic systems as pseudo-random generators is the better succeeded one. It begins in 1989 [16] with some problems due to rounding operations, rising up the "short cycle problem", yet today to be taken into account and focused in the present paper. The list of works around the use of chaotic systems as pseudo-random generator for cryptography is very big, however we consider the following as the most important ones because they focus on some particular problems and solutions in the optimization of this approach, namely: (a) the use of the Lorenz attractor [17] as one of the most used chaotic systems in cryptography; (b) the use of different chaotic systems in the same cipher [18][19] to improve the unpredictability of the cipher evolution; (c) the use of orbit disturbance, or chaotic disturbance, [20] that forces the change in the normal evolution of the chaotic system orbit, improving its unpredictability; (d) the use of some of the proposals together like in [21], where the authors propose a system using both contributions (b) and (c); (e) the protection of the chaotic system seed, or initial conditions, [22] changing the values of the initial conditions of the chaotic system before using them as the cipher key; (f) the protection of the chaotic system state in the calculus of the cipher key [24] using just part of the total number of bits from the chaotic system state to calculate the cipher key; and finally (g) where the use of big dimension arithmetic's could improve the cipher construction due to the bigger space of results [24].

2.2 eLoBa Architecture

eLoBa was proposed in 2010 [1] with good properties for light cryptography as needed in resource constrained devices as for instance smart-cards. In this section we present the eLoBa architecture in a functional approach.

Figure 1 illustrates the eLoBa architecture, where the numbers represent bits.

There is an "Initialization SubSystem" which protects the cipher key from its use in the chaotic system initial conditions. This SubSystem is used for initialize the other three subsystems of eLoBa: the "Chaotic SubSystem"; the "Chaotic Disturbance SubSystem"; and the "Key Mix SubSystem". The initialization process of the "Chaotic Disturbance SubSystem" and the "Key Mix SubSystem" is performed by the "Initialization SubSystem" but through the "Chaotic SubSystem".

The Chaotic SubSystem contains the Lorenz system state variables, is responsible for its the iteration, for give input to the "Chaotic Disturbance SubSystem" each time the Lorenz system is iterated and is also yet responsible for input to the "Key Mix SubSystem" each time the Lorenz system is iterated.

The "Chaotic Disturbance SubSystem" receives the state variables of the Lorenz system from the "Chaotic SubSystem", change them through the use of rotations and LFSRs and then return the new state variables to the "Chaotic SubSystem" to be used in the next iteration. This behavior lead to change of orbit for each iteration of the Lorenz chaotic system.

The "Key Mix SubSystem" receives the three coordinates of the Lorenz system from the "Chaotic SubSystem" with 384 bits and use them to calculate two cipher keys to be used for the next two 128 bits "Data Inputs". This processing of the keys just use 256 bits from the total 384 of the three coordinates and besides that, none of the three coordinates are used completely for the 256 bits of the two keys.

The "Key Control Generation" just manages the iteration of the Lorenz system in the "Chaotic SubSystem" due to the fact that each iteration gives two cipher keys and so the Lorenz system just needs to be iterated once for two "Data Inputs" of the cipher.

For a complete understating of eLoBa, access to [1].

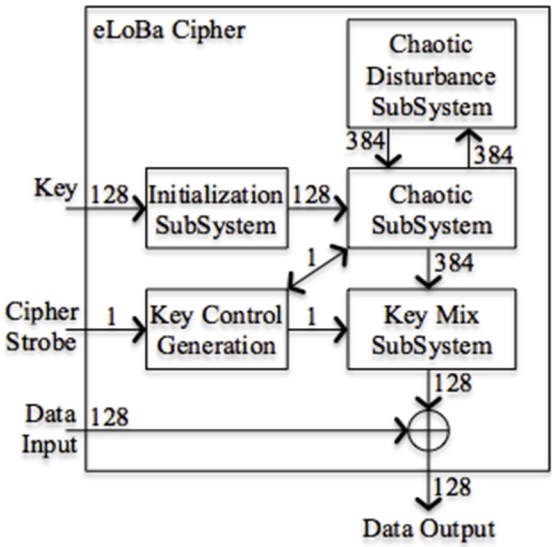

Fig. 1. eLoBa architecture

3 CYCLE LENGTH

In this section we analyze one of the cryptanalysis main topic of keystream cipher generators - the number of bits that are to be generated before the keystream output repeats.

Because the analytical solution of Lorenz differential equations is unknown, we follow a probabilistic strategy for the identification of eLoBa keystream output cycle length. Starting from one specific state, the chaotic trajectory follows a ∞ shaped curve, also known as "Lorenz butterfly". The disturbance sub-system moves the next state to another one in any position of the 3-dimension plane. We assume the distribution of the new position follows an uniform distribution.

For the eLoBa keystream generator with *D* bits for the variable dimension and holding a uniform distribution for the next state, the probability for repeating a previous value after the execution of *N* rounds is given by equation (1). The equation gives the accumulated probability for the first *(N-1)* rounds do not reach a visited state $\prod_{i=1}^{(N-1)}(1 - i.2^{-3D})$, and the N^{th} round reaches a visited state, $N.2^{-3D}$.

$$\Pr(N) = N.2^{-3D} \prod_{i=1}^{(N-1)}(1 - i.2^{-3D}) \qquad (1)$$

We now take a closer look to the sequence of products $\prod_{i=1}^{(N-1)}(1 + \alpha.i)$ in equation (1), where α is equal to -2^{-3D}. The expansion of equation (1) is a *(N-1)* degree polynomial, with coefficients presented in table 1. For example, for *N = 5*, the polynomial is $1 + 10\alpha + 35\alpha^2 + 50\alpha^3 + 24\alpha^4$.

The coefficients C_N^i are evaluated by the set of equations 2.

$$C_N^i = \begin{cases} 0, & if\ i \geq N \\ 1, & if\ i = N \\ C_{(N-1)}^i + (N-1)C_{(N-1)}^{(i-1)}, & if\ 0 < i < N \end{cases} \qquad (2)$$

For example, $C_5^2 = C_4^2 + (5-1).C_4^1 = 11 + 4.6 = 35$.

Table 1: $\prod_{i=1}^{(N-1)}(1 + \alpha.i)$

	0	1	2	3	4	5	6
N=1	1						
N=2	1	1					
N=3	1	3	2				
N=4	1	6	11	6			
N=5	1	10	35	50	24		
N=6	1	15	85	225	274	120	
N=7	1	21	175	735	409	1764	720

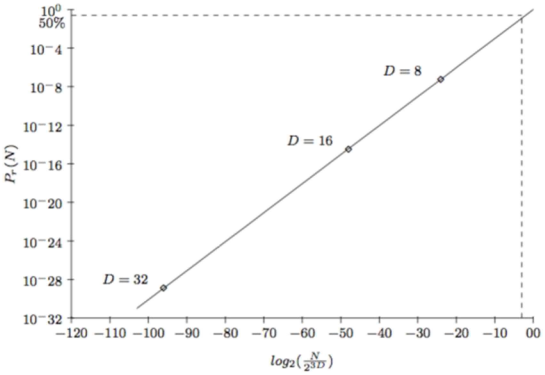

Fig. 2. Pr(N) values

Figure 2 depicts the values of equation (1), wth the horizontal axis normalized to $log_2\left(\frac{N}{2^{3D}}\right)$. For example, the probability of generating one cycle in the keystream output is $2.38 * 10^{-7}$ when the number of rounds satisfies equality $2^{\frac{N}{2^{3D}}} = 2^{-22}$: for 8 bits dimension the number of rounds N is 2^2, for 16 bits dimension *N* is 2^{26}, and for 128 bits dimension *N* is $2^{36\ 2}$. The diamond depicts the probability of generating one cycle when *N* = 1, for dimensions *D* equal to 8, 16 and 32.

The probability 50% of generating one cycle in the keystream output is reached when $N = 2^{3D-1} + 1$. For eLoBa, with *D* = 128, the cycle length is $2^{38\ 3} + 1 \approx 10^{116}$: this number is much larger than the number of estimated hydrogen atoms in the Universe, $4 * 10^{79}$.

4 SERVICE PERFORMANCE OF SMART-CARD AUTHENTICATION

We analyze the performance of eLoBa authentication service against other cypher systems on a constrained device, a Smart-card [27]. Performance of authentication mechanisms was measured with a Gemalto TOP GX4 smart card. It is a Java Card 2.2.1 and Global Platform 2.1.1 compliant, with 68KB available memory. Dedicated coprocessor supports cryptographic algorithms of AES - 128,192 and 256 bits - and RSA up to 2048 bits.

4.1 Architecture

Authentication service follows a challenge-response strategy, and its architecture is depicted in figure 3.

The 128 bit challenge is divided as follow: (1) the upper 123 bits are X-ored with eLoBa key; (2) the fifth bit selects the stream output; (3) the lowest four bits are added to 0x10 to define the number of rounds that the chaotic subsystem must execute.

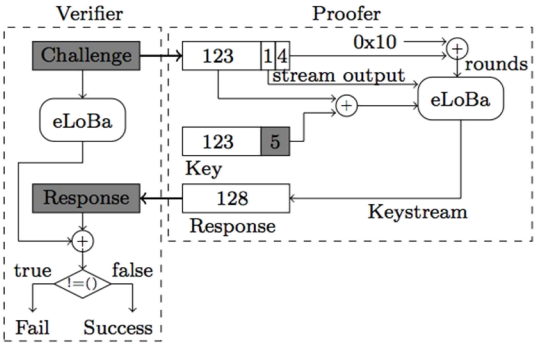

Fig. 3. eLoBa authentication

4.2 Performance Metrics

We implemented eLoBa, and other cipher systems in Java language [30]. All values presented in this article refer to the total time required to carry out an authentication protocol. Total time includes four overhead functions: (a) Smart card connection - 1.71ms; (b) Smart card disconnection - 0.20ms; (c) Applet selection - 3.21ms; (d) Data transmission and reception for an APDU with different lengths are depicted in table 2.

Table 2: Time spent for a data APDU

Data len (bits)	128	512	1024	2048
Time (ms)	6.98	10.99	17.62	30.01

Time spent to send and receive an empty data APDU is 2.98ms. 2048 bit data size overpasses APDU 255 Byte limit, which was overcome by using P1 field. In this section we compare execution times for any random key, as function of key size.

4.3 Coprocessor Efficiency

We implemented two versions of the protocols. In coprocessor version, the computations are performed as much as possible on the cryptographic coprocessor, whereas in the optimized version all computations are performed on the virtual machine of the Java card.

The most computationally expensive operation is modular exponentiation, followed by modular multiplication. The former is directly available through the RSA cipher in the Java Card Cryptographic API. For modular exponentiation, coprocessors implement the CRT - Chinese Remainder Theorem optimization, which requires the previous knowledge of *N* prime factors.

Concerning modular multiplication, this operation cannot be directly executed in the cryptographic coprocessor, as the Java Card API does not provide support for such. A possible workaround is the squaring algorithm [26], implemented by coprocessor. However, this method has its limitations. For instance, the RSA cipher on the card operates only within valid bitlengths for RSA keys and only suports modulus from values starting with 512 bits. This prevents us from using this technique with short modulus values, which is the case of eLoBa that works in 128-bit arithmetic.

Without coprocessor support, we implemented modular multiplication with Barrett reduction [25].

4.3.1 Performance comparison

Figure 4 depicts the execution times for two kinds of authentication protocols: (a) challenge-response based on symmetric cryptosystem AES [28], based on public-key cryptosystem RSA [32], and based on keystream cryptosystem eLoBa; (b) zero- knowledge protocols (ZKP) FFS [29], and GQ [31]. Both protocols were implemented with coprocessor

support (white marks) and programmed only (black marks). Execution times for FFS and GQ programmed protocols are similar.

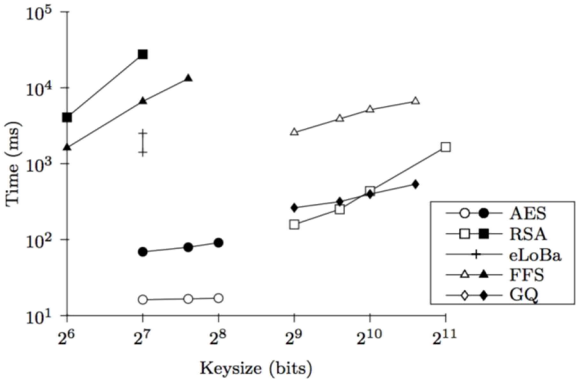

Fig. 4. Comparison of authentication protocols

AES calculations are much faster than RSA calculations, even without support of dedicated coprocessors. Concerning only to performance, GQ protocol stands similar to RSA executed by a dedicated coprocessor. Without coprocessor, RSA and ZKP authentication on smart card is only feasible on short key lengths, which is unsafe.

eLoBa keystream depicts an excellent performance, about one-tenth of AES, and does not require dedicated coprocessors.

5 ACKNOWLEDGEMENTS

Authentication with eLoBa presents good performance when compared with other alternatives, does not require coprocessors and has 128 bits key, which proves its eligibility for authentication on resource constrained devices like smart-cards.

6 CONCLUSION

This work was partially supported by FCT (INESC-ID multiannual funding) through the PIDDAC Program funds. The two authors, Rui Miguel Silva and Rui Gustavo Crespo, initiated this work, however destiny ends the life of Rui Crespo before the end of this work. Here it stays in its honor.

7 REFERENCES

[1] Rui Miguel Silva, Rui Gustavo Crespo, Mário Serafim Nunes, "Enhanced Chaotic Stream Cipher for WSNs", The Fifth International Conference on Availability, Reliability and Security (AReS 2010), Krakow, Poland, February, 2010, pp. 210-215.

[2] L. Pecorra and T. Carrol, “Synchronization in chaotic systems”, Physical review letters 64 (8), 1990, 821–824.

[3] T. Carrol and L. Pecorra, “Synchronization in chaotic circuits”, IEEE Transations in Circuits and Systems (38), 1991, pp. 453–456.

[4] N. Philip and K. Joseph, “Chaos for stream cipher”, Recent Advances in Computing and Communications (ADCOM 2000), 2000, pp. 35–42.

[5] A. Oppenheim, G. Wornell, S. Isabelle, and K. Cuomo, “Signal processing in the context of chaotic signals”, IEEE International Conference on Acoustics, Speech and Signal Processing (4), 1992, pp. 117–120.

[6] L. Kocarev, K. Halle, K. Eckert, L. Chua, and U. Parlitz, “Experimental demonstration of secure communications via chaotic synchronization”, International Journal on Bifurcation and Chaos (2/3), 1992, 709–713.

[7] K. Cuomo and A. Oppenheim, “Synchronization of lorenz-based chaotic circuits with applications to communications”, IEEE Transations on Circuits and Systems - II, (40), 1993, pp. 635–642.

[8] P. Celka, “Synchronization of chaotic systems through parameter adaptation”, International Symposium on Circuits and Systems (1), 1995, pp. 692–695.

[9] S. Papadimitriu, A. Bezerianos, and T. Buontis, “Secure communication with chaotic systems of differential equations”, IEEE Transactions on Computers (46), 1997, p. 27.

[10] Q. Memon, “Synchronized chaos for network security”, Computer Communications (6), 2003, 498–505.

[11] T. Habutsu, Y. Nishio, I. Sasase, and S. Mori, “A secret key cryptosystem using a chaotic map”, The transactions of the IEICE (E73), 1990, pp. 1041–1044.

[12] E. Biham, “Cryptanalysis of the chaotic-map cryptosystem suggested at eurocrypt’91”, Advances in Cryptology (EUROCRYPT’91) (0547), 1991, pp. 532–534.

[13] H. Gutowitz, “Cryptography with dynamical systems”, Cellular automata and Cooperative Phenomena, 1993, pp. 237–274.

[14] Z. Kotulski and J. Szczepanski, “Discrete Chaotic Cryptography (DCC) - New Method for Secure Communication”, 1997.

[15] N. Masuda and K. Aihara, “Cryptosystems with Discretized Chaotic Maps”, IEEE Transactions on Circuits and Systems - I:

Fundamental Theory and Applications (49), 2002, pp. 28–40.

[16] R. Matthews, “On the derivation of a ”chaotic” encryption algorithm”, Cryptology (13/1), 1989, 29–42.

[17] J. Carrol, J. Verhagen, and P. Wong, “Chaos in cryptography: The escape from the strange attractor”, Cryptology (16/1), 1992, 52–72.

[18] V. Protopopescu, R. Santoro, and J. Tollover, “Fast and secure encryption - decryption method based on chaotic dynamics”, US Patent No. 5479513, 1995.

[19] N. Philip and K. Joseph, “Chaos for stream cipher”, Recent Advances in Computing and Communications (ADCOM2000), 2000, pp. 35–42.

[20] T. Sang, R. Wang, and Y. Yan, “Perturbation-based algorithm to expand cycle length of chaotic key stream”, Electronic Letters (34/9), 1998, 873–874.

[21] S. Li, X. Mou, and Y. Cai, “Pseudo-random bit generator based on couple chaotic systems and its applications in stream-cipher cryptography”, Second International Conference on Cryptology in India (INDOCRYPT ’01), 2001, pp. 316–329.

[22] N. Pareek, V. Patidar, and K. Sud, “Discrete chaotic cryptography using external key”, Physics Letters (A309), 2003, pp.75–82.

[23] L. Barash and L. Shchur, “Periodic orbits of the ensemble of sinai-arnold cat maps and pseudorandom number generation”, Physical Review (E73), 2006, pp. 1–18.

[24] D. Wheeler and R. Matthews, “Supercomputer investigations of a chaotic encryption algorithm”, Cryptologia (15/2), 1991, pp. 140–152.

[25] P. Barret. “Implementing the rivest shamir and adleman public key encryption algorithm on a standard digital signal processor”, Advances in cryptology-Crypto’86, 1987, pp. 311–323.

[26] L. Batina, J.-H. Hoepman, B. Jacobs, W. Mostowski, and P. Vullers, “Developing efficient blinded attribute certificates on smart cards via pairings”, 9th IFIP WG 8.8/11.1 CARDIS 2010, 2010, pp. 209–222.

[27] Zhiqun Chen, “Java Card Technology for Smart Cards: Architecture and Programmer’s Guide”, Prentice-Hall, 2000.

[28] J. Daemen and V. Rijmen, “The Design of Rijndael: AES - The Advanced Encryption Standard”, Springer Verlag, 2001.

[29] U. Feige, A. Fiat, and A. Shamir, “Zero-knowledge proofs of identity”, Journal of Cryptology (1/2), 1988, pp. 77-94.

[30] James Gosling, Bill Joy, Guy Steele, and Gilad Bracha, “Java Language Specification”, Addison-Wesley, 3 edition, 2005.

[31] L.C. Guillou and J.-J. Quisquater, “A practical zero-knowledge protocol fitted to security microprocessor minimizing both transmission and memory”, EUROCRYPT’88, 1988, pp. 123-128.

[32] R.L. Rivest, A. Shamir, and L. Adleman, “A method for obtaining digital signatures and public-key cryptosystems”, Communications of the ACM (21/2), 1978, pp. 120-126.