Lightweight Stream Ciphers based on Chaos for Time and Energy Constrained IoT Applications

Ons Jallouli, Maryline Chetto, Safwan El

Lightweight Stream Ciphers based on Chaos for Time and Energy Constrained IoT Applications

Ons Jallouli
Ibm France
17 Avenue de l'Europe
92275 Bois-Colombes, France
onsjallouli90@gmail.com

Maryline Chetto
Nantes Université, École Centrale Nantes
CNRS, LS2N, UMR 6004
F-44000 Nantes, France
maryline.chetto@univ-nantes.fr

Safwan El Assad
Nantes Université
CNRS, IETR, UMR 6164
F-44000 Nantes, France
safwan.elassad@univ-nantes.fr

***Abstract*—The design of efficient and secure cryptographic algorithms is a fundamental problem of cryptography. Due to the tight cost and constrained resources devices such as Radio-Frequency IDentification (RFID), wireless sensors, smart cards, health-care devices, lightweight cryptography has received a great deal of attention. Recent research mainly focused on designing optimized cryptographic algorithms which trade offs between security performance, time consuming, energy consumption and cost. In this paper, we present two chaotic stream ciphers based on chaos and we report the results of a comparative performance evaluation study. Compared to other crypto-systems of the literature, we demonstrate that our designed stream ciphers are suitable for practical secure applications of the Internet of Things (IoT) in a constrained resource environment.**

***Index Terms*—Security, lightweight cryptography, energy constrained devices, chaotic stream ciphers, Internet of Things.**

I. Introduction

The concept of IoT is becoming an increasingly growing topic, due to huge advancements in wireless networking technology and standardization of communication protocols [1] [2]. The core idea of this concept lies in the presence of everyday physical objects known as things which are connected to the internet. Interconnection between things is made possible by technologies such as RFID, Wireless Sensor Networking (WSN), cloud servicing, machine-to-machine interfacing (M2M), etc.

Secure data transmission is a very significant issue because confidential and proprietary information have to be transmitted, especially in healthcare applications. Unfortunately, existing cryptographic techniques developed for enterprise and desktop computing might not satisfy embedded application requirements as they can be too slow, huge and very power consuming [3].

Smart devices of the IoT, including sensors, are inherently resource constrained with regard to memory, communication band-width, processing power and energy [4]. The most widespread energy sources are typically batteries, renewable energy sources of the environment, or both. A wireless device should operate for several years even for decades. Energy consumption is consequently a central performance factor since it directly impacts lifetime of the device. Hence, a challenging topic concerns the design of efficient and lightweight cryptographic techniques to guarantee secure data transmission in the IoT. Such techniques should fit the low energy, computation and memory capabilities of cyber-physical systems and provide an optimized security/cost/performance trade-off [5]. This is why the new field of Light Weight Cryptography (LWC) is emerging.

Many encryption algorithms are nowadays used for information security [6] [7] [8] [9]. All lightweight encryption algorithms are based on Symmetric encryption, due to their low-cost computation compared to Asymmetric encryption [10]. One of the main categorization methods for encryption relies on the form of the input data they operate on. Two types exist: Block Cipher and Stream Cipher. Block ciphers perform confusion and diffusion operations on N-bit block of plain-text data using the secret key and generate N-bits block of cipher-text. Stream ciphers are based on One Time Pad (OTP) principle. This is achieved through a random keystream with same size as plaintext then XORing both of them to generate the ciphertext. Strength of the stream cipher depends on robustness of the generated keystream. They are suited to low cost devices since having low memory and computation requirements.

The four main characteristics that differentiate one stream cipher from another are: ability to secure the protected data, speed i.e. computational complexity, energy consumption and memory required in doing so. This paper aims to show a performance comparison based on these four characteristics between two lightweight chaotic stream ciphers that we recently designed [11], [12].

The rest of this paper is organized as follows. In the next section, an overview of the related works are given. And we present the two recently designed chaotic stream ciphers. In section III, we report the results of a performance evaluation study that deals with speed, energy consumption and memory

This work is partially supported by a French grant from the Isite NExT (Project entitled "Optimisation et Gestion en Temps Réel de la Consommation Energétique d'un Système Cobotique") within the New Partnerships framework.

requirement. Section IV concludes the paper.

II. Related works

In the last years, many lightweight encryption algorithms have been proposed. In [5] and [13], a selection of recently published LWC hardware and software implementations are described and compared. Chip size, number of clock cycles and/or energy consumption are considered for an evaluation in hardware implementation. Memory requirements (ROM and RAM) and number of clock cycles are quantified for software implementation. Lightweight encryption algorithms use less than 32 KB ROM and 8 KB RAM [14].

The traditional stream ciphers, namely Rivest Cipher 4 (RC4) [15], A5/1 and E0 [16] suffer from serious security vulnerabilities and should not be used in new generation applications. AES in CTR mode currently appears as the only secure and widespread solution for stream encryption [17]. This fact has inspired researchers in their efforts to propose new stream ciphers with better performance.

The eSTREAM project [18] which held from 2004 to 2008, under the Information Societies Technology (IST) Program gave rise to new fast and secure stream ciphers. Two profiles of ciphers for software and hardware implementations were defined. Profile 1 consists of ciphers with high throughput in software that are faster than the 128-bits AES-CTR. Finalist ciphers include Rabbit, HC-128, Salsa 20/12 and SOSEMANUK. Profile 2 consists of ciphers that are more compact than the 80-bits AES in hardware. The finalists include MICKEY 2.0, Grain and Trivium. These ciphers were found to be secure against known attacks through cryptanalysis. Grain and Trivium are the most notable ciphers for hardware implementation whereas Salsa and Rabbit are suitable for compact embedded software implementation.

In the last years, many chaos-based stream ciphers have been proposed [19] [20] [21].Tremendous interest in using chaotic functions in cryptography is due to their intrinsic properties such as ergodicity, randomness and high sensitivity to initial conditions and parameters [22]. We recently designed two novel stream ciphers based on chaotic systems [11] [12]. The principle of a stream cipher is given in Figure 1.
The plain text P_i is XORed with the keystream generated by a pseudo chaotic number generator (PCNG) in order to obtain the cipher text C_i Inputs are a secret shared key K and an initial vector IV. As the PCNG is deterministic, the same keystream can be generated in decryption. Then, to recover the original plaintext P_i, we XOR the same keystream with the cipher text C_i . We will describe these two PCNGs. Performance security study shows that they are secure and they resist against known attacks [11] [12].

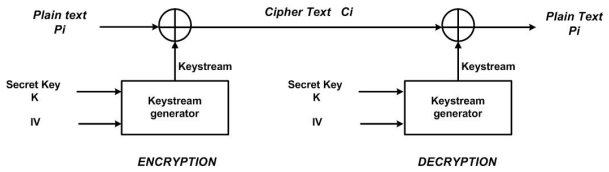

Fig. 1. General structure of a stream cipher.

We now briefly present two new PCNGS respectively Coupling-Multiplexing PCNG (CM-PCNG) and Coupling-Swap PCNG (CS-PCNG). Each one has secret key K and initial vector IV as inputs, and keystream as output. Each one contains four functional blocks, namely IV-setup, Key-setup, Internal State and Output function as described in what follows:

- IV-setup function generates sub-IVs from the the initial vector IV, that will be used in the key-setup,
- Key-setup function computes initial values of the chaotic maps,
- Internal State function performs chaotic techniques (coupling/coupling and swap techniques) for producing future samples of the output function,
- Output function produces the output sequence.

The two functions Internal State and Output differentiate one stream cipher from the other one. CM-PCNG is based on coupling and multiplexing techniques whereas CS-PCNG uses coupling and swap techniques.

The architecture of CM-PCNG is given in Fig.2. Three weakly coupled chaotic maps, namely PWLCM, Skew Tent and Logistic and a multiplexing chaotic technique are used.

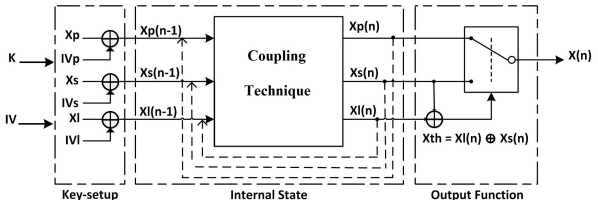

Fig. 2. Structure of CM-PCNG.

The Internal State function is governed by the following equations:

$$Xp(n) = \sigma_1 \times Fp[Xp(n-1)] + \varepsilon_{12} \times Fs[Xs(n-1)] + \varepsilon_{13} \times Fl[Xl(n-1)]$$

$$Xs(n) = \varepsilon_{21} \times Fp[Xp(n-1)] + \sigma_2 \times Fs[Xs(n-1)] + \varepsilon_{23} \times Fl[Xl(n-1)]$$

$$Xl(n) = \varepsilon_{31} \times Fp[Xp(n-1)] + \varepsilon_{32} \times Fs[Xs(n-1)] + \sigma_3 \times Fl[Xl(n-1)]$$

where $\sigma_1 = 2^N - \varepsilon_{12} - \varepsilon_{13}$; $\sigma_2 = 2^N - \varepsilon_{21} - \varepsilon_{23}$; $\sigma_3 = 2^N - \varepsilon_{31} - \varepsilon_{32}$.

ε_{ij} are the weakly coupling parameters, ranging from 1 to 2^k and k$\leq$5. And $Fp[Xp(n-1)]$, $Fs[Xs(n-1)]$ and $Fl[Xl(n-1)]$ are the discrete functions of the chaotic maps PWLCM, Skew Tent and Logistic respectively [23].

The resulting multiplexed samples of sequence $X(n)$ are controlled by chaotic sample $Xth(n)$ and threshold T, as shown in Fig.2, and defined as follows:

$$X(n) = \begin{cases} Xp(n), & \text{if } 0 < Xth(n) < T \\ Xs(n), & \text{otherwise} \end{cases} \quad (1)$$

with $Xth(n) = Xl(n) \oplus Xs(n)$.

The architecture of CS-PCNG is presented in Fig.3. CS-PCNG contains two chaotic maps, namely PWLCM and SkewTent, and includes coupling and swap chaotic techniques.

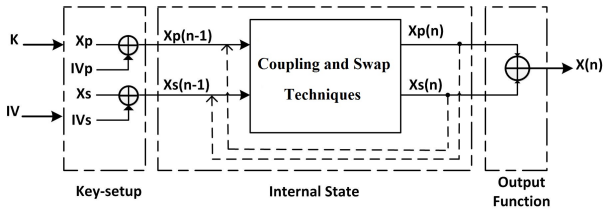

Fig. 3. Structure of CS-PCNG.

The following equations govern CS-PCNG:

$$\begin{cases} Xp(n) = \sigma_1 \times Fp[Xs(n-1)] + \varepsilon_{12} \times Fs[Xp(n-1)] \\ Xs(n) = \varepsilon_{21} \times Fp[Xs(n-1)] + \sigma_2 \times Fs[Xp(n-1)] \end{cases} \quad (2)$$

with $\sigma_1 = 2^N - \varepsilon_{11}$; $\sigma_2 = 2^N - \varepsilon_{22}$.
$Fp[Xp(n-1)]$ and $Fs[Xs(n-1)]$ are the respective discrete functions of the chaotic Pwlcm and Skew Tent maps.

Output samples $X(n)$ are calculated throughout samples $Xp(n)$ and $Xs(n)$ as follows:

$$X(n) = Xp(n) \oplus Xs(n). \quad (3)$$

We studied the security performance of the two proposed stream ciphers called Coupling-Multiplexing Stream Cipher (CM-SC) and Coupling-Swap Stream Cipher (CS-SC) that use CM-PCNG [11] and CS-PCNG [12] respectively as keystream generator. Both stream ciphers CM-SC and CS-SC are secure and efficient and could be involved in applications where data security presents a big concern. Moreover, we note that stream cipher CM-SC offers better security characteristics than stream cipher CS-SC.

III. Performance evaluation

Design of LWC always gives rise to a compromise between security, cost and performance. Such implementations involve computationally intensive operations that require to be executed timely and energy efficiently under limited memory and power capabilities. We evaluate the two Stream ciphers CM-SC and CS-SC in terms of computation time, energy consumption and memory space requirement.

TABLE I
COMPUTATION PERFORMANCE MEASUREMENTS.

Stream cipher	Average encryption time (μs)	ET (Mbits/s)	NCpB
CM-SC	2403.04	624.2	31.78
CS-SC	1059.5	1415.76	14
Rabbit	855.45	1753.46	11.31
HC-128	1330	1127.81	17.59
AES-CTR	-	-	21.2

A. Experimentation setup

Experiments were performed on a personal computer built around Intel Core (TM) i5 @2.60 GHZ with 15.6 GB under Ubuntu 14.04 Trusty Linux operating system. Algorithms were coded in C language and applied to Lena image with size $(256 \times 256 \times 3)$. Algorithms were executed for 100 different keys.

B. Encryption time measurement

In the design of a lightweight cryptosystem, the computation performance of the encryption algorithm is an important factor to exhibit the performance of a lightweight cryptosystem. We calculate the average encryption time in micro second (μs), the encryption throughput in Mega bit par second BR(Mbit/s), and the number of cycles needed to encrypt one byte (NCpB), given as follow:

$$ET = \frac{Image\ size(MByte)}{Encryption\ time(s)} \quad (4)$$

$$NCpB = \frac{CPUspeed(Hz)}{ET(Byte/s)} \quad (5)$$

The computation performance measurements are reported in table I. Results show that CS-SC algorithm has better computing performance than CM-SC algorithm. It needs about 57% less number of cycles to encrypt one byte. Also, these computing performances are comparable or better than some lightweight stream ciphers of the literature.

C. Energy measurement

Energy and power consumption are critical metrics in the design of LWC algorithms. Traditionally, the energy consumption of an application is measured using counters. Recently, Intel introduced a new approach for measuring the energy consumption of on-core hardware components. Every Intel SandyBridge chip includes a new interface called "Running Average Power Limit" (RAPL) [24].
RAPL provides a set of counters which reports energy and power consumption information. RAPL is not an analog power meter, but rather uses a software power model. This one estimates energy usage through hardware performance counters and I/O models [25]. To obtain these results, the

TABLE II
ENERGY CONSUMPTION (J).

Stream cipher	CM-SC	CS-SC	Rabbit	HC-128
PKG energy(J)	0.078613	0.022672	0.013855	0.038768
PP0 (J)	0.036316	0.010297	0.006104	0.020316
PP1 (J)	0.007568	0.000112	0.000150	0.000030
DRAM (J)	0.012939	0.002669	0.001648	0.003806

TABLE III
POWER CONSUMPTION IN MILLIWATT (MW).

Stream cipher	CM-SC	CS-SC	Rabbit	HC-128
Power Estimation (mW)	3.4	2.9	2.7	3.18

user needs a Model Specific Register (MSR) which updates every millisecond approximately. The Linux "MSR driver" permits to access these registers. RAPL gives various energy measurements in Joules such as the energy consumption of the total processor package (referred to as Package (PKG)), the total combined energy used by all cores (Power-Plane 0 (PP0) which includes all processor caches) and the energy readings for the DRAM interface. Also, some versions of SandyBridge chips report power usage due to the on-board GPU (Power-Plane 1 (PP1)).

The RAPL interface is used here to determine the energy consumed by the encryption function. Table II gives the different average values of energy consumption for the two stream ciphers CM-SC and CS-SC which are compared to that of Rabbit and HC-128.

We also report a power estimation through the PowerTop Linux tool which permits to diagnose issues with power consumption and power management. This tool helps to point out the power inefficiencies of a program. It shows how well the different hardware power-saving features are used and report software components that are preventing optimal usage [26] [27]. It also returns a power estimation for each device. Power measurements are given in table III.

Results shown in Tables II and III indicate that CS-SC algorithm has less energy and power consumption compared to CM-SC and HC-128 algorithms. Indeed, CS-SC consumes about 30% of energy consumed by the CM-SC algorithm, and 60% of that of the HC-128 algorithm. However, CS-SC consumes about two times more energy than Rabbit.

D. Memory assessment

Whatever the complexity of the cryptographic primitives in terms of computational overhead and memory usage, the hardware resources available must be able to minimize their impact on the execution time of the secured applications. However, embedded devices often have inherent limitations in terms of memory space. Hence, it would be necessary to analyse how these primitives perform over highly-constrained devices.

TABLE IV
CODE SIZE AND RAM CONSUMPTION (BYTES).

Stream cipher	Code size (bytes)	RAM consumption (bytes)
CM-SC	7240	660
CS-SC	6562	564
Trivium	5764	1516
Snow	12861	1741
Rabbit	1714	216
HC-128	23100	4556

We calculate the requirements of the designed stream ciphers in terms of RAM consumption and code size. We use the FELICS framework which is an open source benchmarking framework. Its goal is to evaluate the performance of software implementation of lightweight cryptographic primitives for embedded devices [28].

The code size measures the amount of data that is stored in the Flash memory of the target device. The RAM consumption includes the stack requirements and data requirement. The former presents the maximum value of RAM used to store local variables. The later forms represents the static RAM, given by the size of the constants stored in target device RAM (such as data to encrypt, key, initial vectors...). Table IV clarifies the code size and RAM consumption measurements of the four tested algorithms using FELICS framework. The code size and RAM consumption values, for Rabbit and HC-128 stream ciphers, are given by [14].

The results show that RAM and ROM requirements of the two designed stream ciphers CM-SC and CS-SC, are less than 8 KB and 32 KB respectively. Therefore, they are suitable for resource-constrained devices as those encountered in the IoT.

IV. CONCLUSION

In this paper, we have provided a quantitative evaluation of computation performance, energy and power consumption and memory size requirement of lightweight stream cipher algorithms including CM-SC and CS-SC. These algorithms are partly the contributions of a PhD work prepared at the university of Nantes [11] [12]. We may conclude that the stream cipher CM-SC offers better security characteristics than the stream cipher CS-SC.
We have conducted experiments that demonstrate that our chaos-based stream ciphers can be efficiently implemented on energy and time constrained resources devices of the internet of things where security is a big concern.

REFERENCES

[1] M. A. Feki, F. Kawsar, M. Boussard, and L. Trappeniers, "The internet of things: the next technological revolution," *Computer*, vol. 46, no. 2, pp. 24–25, 2013.

[2] L. Ericsson, "More than 50 billion connected devices," *White Paper*, 2011.

[3] P. Koopman, "Embedded system security," *Computer*, vol. 37, no. 7, pp. 95–97, 2004.

[4] T. Good and M. Benaissa, "A low-frequency rfid to challenge security and privacy concerns," in *Mobile Adhoc and Sensor Systems, 2009. MASS'09. IEEE 6th International Conference on*. IEEE, 2009, pp. 856–863.

[5] T. Eisenbarth and S. Kumar, "A survey of lightweight-cryptography implementations," *IEEE Design & Test of Computers*, vol. 24, no. 6, 2007.

[6] G. Singh, "A study of encryption algorithms (rsa, des, 3des and aes) for information security," *International Journal of Computer Applications*, vol. 67, no. 19, 2013.

[7] J. D. Golić, "Stream cipher encryption of random access files," *Information processing letters*, vol. 69, no. 3, pp. 145–148, 1999.

[8] P. Sarkar, "Tweakable enciphering schemes using only the encryption function of a block cipher," *Information Processing Letters*, vol. 111, no. 19, pp. 945–955, 2011.

[9] P. Szalachowski, B. Ksiezopolski, and Z. Kotulski, "Cmac, ccm and gcm/gmac: Advanced modes of operation of symmetric block ciphers in wireless sensor networks," *Information Processing Letters*, vol. 110, no. 7, pp. 247–251, 2010.

[10] S. Charlwood and P. James-Roxby, "Evaluation of the xc6200-series architecture for cryptographic applications," in *International Workshop on Field Programmable Logic and Applications*. Springer, 1998, pp. 218–227.

[11] O. Jallouli, S. El Assad, M. Chetto, and R. Lozi, "Design and analysis of two stream ciphers based on chaotic coupling and multiplexing techniques," *Multimedia tools and applications*, 2017.

[12] O. Jallouli, S. El Assad, and M. Chetto, "Robust chaos-based stream-cipher for secure public communication channels," in *International Conference on Internet Technology and Secured Transactions*, 2016.

[13] C. Paar, A. Poschmann, and M. Robshaw, "New designs in lightweight symmetric encryption," in *RFID Security*. Springer, 2008, pp. 349–371.

[14] C. Manifavas, G. Hatzivasilis, K. Fysarakis, and K. Rantos, "Lightweight cryptography for embedded systems–a comparative analysis," in *Data Privacy Management and Autonomous Spontaneous Security*. Springer, 2014, pp. 333–349.

[15] G. Paul and S. Maitra, *RC4 stream cipher and its variants*. CRC press, 2011.

[16] M. D. Galanis, P. Kitsos, G. Kostopoulos, N. Sklavos, O. Koufopavlou, and C. E. Goutis, "Comparison of the hardware architectures and fpga implementations of stream ciphers," in *Electronics, Circuits and Systems, 2004. ICECS 2004. Proceedings of the 2004 11th IEEE International Conference on*. IEEE, 2004, pp. 571–574.

[17] C. Manifavas, G. Hatzivasilis, K. Fysarakis, and Y. Papaefstathiou, "A survey of lightweight stream ciphers for embedded systems," *Security and Communication Networks*, 2015.

[18] M. Robshaw, "The estream project," in *New Stream Cipher Designs*. Springer, 2008, pp. 1–6.

[19] S. Lian, J. Sun, J. Wang, and Z. Wang, "A chaotic stream cipher and the usage in video protection," *Chaos, Solitons & Fractals*, vol. 34, no. 3, pp. 851–859, 2007.

[20] P.-H. Lee, S.-C. Pei, and Y.-Y. Chen, "Generating chaotic stream ciphers using chaotic systems," *Chinese Journal of physics*, vol. 41, no. 6, pp. 559–581, 2003.

[21] L. Shujun, M. Xuanqin, and C. Yuanlong, "Pseudo-random bit generator based on couple chaotic systems and its applications in stream-cipher cryptography," in *International Conference on Cryptology in India*. Springer, 2001, pp. 316–329.

[22] L. Kocarev and S. Lian, *Chaos-based cryptography: Theory, algorithms and applications*. Springer, 2011, vol. 354.

[23] O. Jallouli, S. El Assad, M. A. Taha, M. Chetto, R. Lozi, and D. Caragata, "An efficient pseudo chaotic number generator based on coupling and multiplexing techniques," in *International Conference on Emerging Security Information, Systems and Technologies (SECURWARE 2016)*, 2016, pp. paper–30 040.

[24] I. Corportation, "Ia-32 intel architecture software developer's manual," *Intel Corportation*, 2001.

[25] E. Rotem, A. Naveh, A. Ananthakrishnan, E. Weissmann, and D. Rajwan, "Power-management architecture of the intel microarchitecture code-named sandy bridge," *Ieee micro*, vol. 32, no. 2, pp. 20–27, 2012.

[26] (2016) Powertop — 01.org. [Online]. Available: https://01.org/powertop

[27] P. Larsson, "Energy-efficient software guidelines," *Intel Software Solutions Group, Tech. Rep*, 2011.

[28] D. Dinu, A. Biryukov, J. Großschädl, D. Khovratovich, Y. Corre, and L. Perrin, "Felics–fair evaluation of lightweight cryptographic systems," in *NIST Workshop on Lightweight Cryptography*, 2015.