Xiankleber C. BENJAMIM[1], Felipe O. S. GAMA[1], Rodrigo S. SEMENTE[2],Elmer R. L. VILLAREAL[2], Andrés O. SALAZAR[1]

Universidade Federal do Rio Grande do Norte (1), Universidade Federal Rural do Semi-Árido (2)

doi:10.15199/48.2018.08.21

Development and Analysis of a Cryptographic Algorithm for Genetically Optimized Wireless Sensor

Abstract. *This article proposes a methodology for obtaining a cryptographic algorithm, optimized for wireless sensor networks, through genetic algorithm. With the objective of increasing the level of security, computational efficiency and highlighting the energy consumption, considering that the autonomy of the wireless sensor devices is directly influenced by this factor. In aptitude function of genetic algorithm, were used metrics of algorithm runtime, maximum deviation and irregular, space occupied in memory and correlation coefficient (a new proposed metric), in order to find a safe and fast algorithm. The results obtained through computational simulations show the efficiency of the proposed methodology, in terms of processing time, coefficient of correlation and occupation of memory.*

Streszczenie. *W tym artykule zaproponowano metodologię uzyskiwania algorytmu kryptograficznego, zoptymalizowanego dla bezprzewodowych sieci czujników, za pomocą algorytmu genetycznego. W celu zwiększenia poziomu bezpieczeństwa, wydajności obliczeniowej i podkreślenia zużycia energii, biorąc pod uwagę fakt, że ten czynnik ma bezpośredni wpływ na autonomię bezprzewodowych czujników. W funkcji uzdatniania algorytmu genetycznego wykorzystano metryki czasu pracy algorytmu, maksymalnego odchylenia i nieregularności, miejsca zajmowanego w pamięci i współczynnika korelacji (nowa proponowana metryka), aby znaleźć bezpieczny i szybki algorytm. Wyniki uzyskane za pomocą symulacji obliczeniowych pokazują efektywność proponowanej metodologii, pod względem czasu przetwarzania, współczynnika korelacji i zajęcia pamięci.* ***(Opracowanie i analiza kryptograficznego algorytmu dla zoptymalizowanego genetycznie czujnika bezprzewodowego)***

Keywords: Cryptographic Algorithm, Genetic Programming, Wireless Sensor Networks
Słowa kluczowe: Algorytm kryptograficzny, Programowanie genetyczne, Sieci czujników bezprzewodowych

Introduction

The wireless sensor networks consist of nodes interconnected by wireless channels with sensing, control and actuation capacity. The data obtained and transmitted by the wireless sensor nodes in an industrial environment, especially when the sensor nodes are installed outdoors, must be confidential. For this purpose, are available in the literature various techniques that aim to maintain the confidentiality of the information transmitted. Among them, there is an approach that employs the increase of data security transmitted in wireless sensor networks, it is the use of cryptography techniques, according to [1].

Cryptographic security is responsible for keeping the information confidential, preventing the system from being invaded. By allowing the inclusion of false information in the system, an invasion can lead to even greater failures in the entire monitoring and control process, according to [1].

The most widely used standard in wireless sensor networks is the IEEE 802.15.4 [2], created in parallel with the Network Protocol ZigBee [3]. The standard IEEE 802.15.4 defines the physical layer and the Medium Acces Control (MAC) layer, as well as the recommended encryption technique for system security, the Advanced Encryption Standard (AES) [4].

The AES-128 is an efficient technique in terms of data processing and security. There are some reports of security breaches in partial versions of the algorithm, which does not mean that it is unsafe. However, there are studies indicating that it can be broken in a timely manner in the coming decades [5, 6].

It is important to emphasize that the energy efficiency of the nodes of the wireless sensor networks is more critical than in designed systems in which it does not need an autonomy of months and even years. In this context, electronic devices of low power and complexity are used, such as 8-bit and low memory microprocessors [7, 8].

There are papers in the literature that demonstrate that it is possible to create cryptographic algorithms with acceptable performance and security characteristics using genetic programming (GP) [9, 10, 11]. However, none of these papers presented results focusing on devices with only 8-bit processing capacity, taking into account both energy efficiency, occupied memory space and security, or a more robust comparative study, including cryptographic algorithms in effect.

Taking all these aspects into account and seeking to improve the security efficiency of wireless sensor networks, new algorithms can be developed for this purpose [12]. However, this work proposes the use of genetic algorithms with the objective of obtaining an optimized cryptographic algorithm for the wireless sensor networks, based on the Maximum and Irregular Deviation Ratio (MIDR), correlation coefficient, occupied memory and the execution time of the algorithm.

To demonstrate the feasibility of using the GP technique in the development of cryptographic algorithms, two classical encryption algorithms Advanced Encription Standart, XXTEA, and the CRYSEED algorithm presented in the paper [12]. The results presented in this work demonstrate that the use of GA can create efficient cryptographic algorithms, being a significant advance for the diffusion of artificial intelligence techniques.

Cryptographic Algorithms

This section presents two cryptographic algorithms which were used in this work to compare performance of the proposed algorithm.

The Advanced Encryption Standard (AES), is the chosen algorithm by the US government for the protection of data traffickedby digital means [13]. AES can use keys of 128, 192 and 256 bits.

The Algorithm considers groups of 16 bytes that form one state, which in other words is simply a 4×4 array with each element being a byte. This algorithm can be divided into four basic operations:

- AddRoundKey: Each byte of a state is combined with a byte of the key through the XOR (OR Exclusive) bit by bit.
- SubBytes: A nonlinear substitution where each byte is replaced by another according to a table lookup.
- ShiftRows: The bytes of each line undergo a shift cyclic by a certain amount of steps.
- MixColumns: Each column of state $a(x)$ is combined according to a known matrix $c(x)\ 4 \times 4$, which multiplied by state columns returns a new column for the resulting

state $b(x)$. The operation is reversible, simply by multiplying the columns of state $b(x)$ by a matrix $d(x)$, the result will be state $a(x)$.

Given these four operations, the algorithm can be divided into four stages, where the third step is usually repeated 20 times:

1. Key expansion based on key scheduling;
2. Start round: AddRoundKey;
3. Rounds:
   - SubBytes;
   - ShiftRows;
   - MixColumns;
   - AddRoundKey
4. Final round (without column mixing):
   - SubBytes;
   - ShiftRows;
   - MixColumns;

In the work [14] he developed the block encryption, Tiny Encryption Algorithm (TEA), which was notable for its simplicity and few lines of code, but that soon presented a security flaw that made the maximum size of its equal key 126 bits, making it so easy to break it such as a hash table or cryptographic scatter, tables substitutions created by spreading functions.

In 1997 the TEA was improved with an extension, thus generating Block TEA, or as it is better known the XTEA (eXtended TEA), which keys with a size of 128 bits [15]. Due to other security issues, the algorithm was restructured a second time, creating the XXTEA [16].

The XTEA is based on the classical Feistel network, varying only the Feistel function, as can be observed in Figure 1. XOR and modular sum operations are represented, respectively, by the square and the circle in Figure 1. The left and right shift operations are also used, but with a fixed number of offsets, 4 bits and 5 bits respectively.

XXTEA is an unbalanced Feistel network, whose input block is divided into N parts, each part being called X_R. So, the encryption of each die depends both his predecessor and successor neighbors , as observed in Figure 2, which presents the Feistel function that depends on both the X_{R-1} and X_{R+1}. Also used is K_R, which applies a left-hand cyclic shift of 2 bits each round.

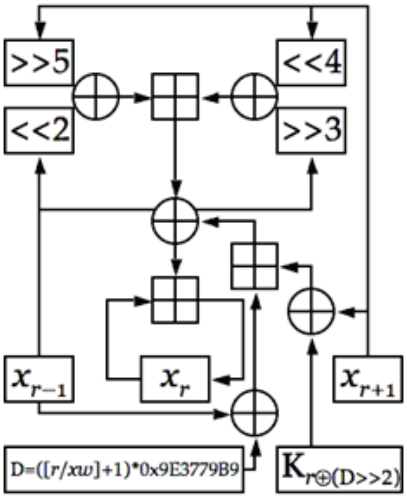


Fig. 1. XXTEA Flowchart. Source [18]

A fault in the XXTEA has already been found, with key breaking through differential cryptanalysis, performing a number of queries of only 2^{59} [19]. This method is a little difficult to apply in practice in an RSSF, since you have to choose the correct plain text for comparison. The data generated by a node of the RSSF are obtained with each measurement, they are not data that can be known by an attacker, being difficult to carry out this attack. Even so, this break can mean the existence of other faults, which may indicate the non-recommendation of its use, even in the Wireless sensor network.

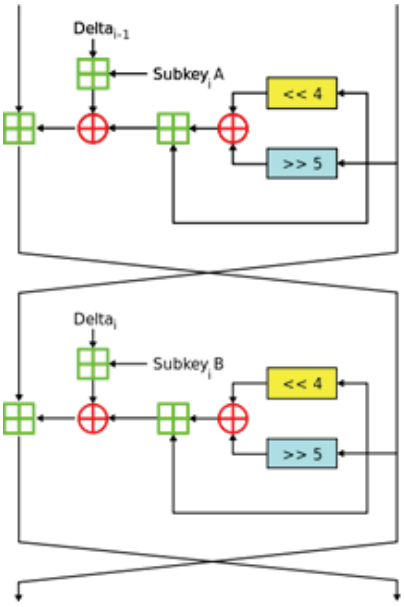


Fig. 2. XXTEA Feistel function. Source [17]

Genetic Programming

Based on natural evolution, several computational program solution methods have been developed, such as genetic algorithm, evolutionary programming, evolutionary strategies and genetic programming. The genetic algorithm allows the creation of programs to solve a problem, created automatically by the computer using a procedure inspired by evolution [20]. This program is usually represented by a tree structure, where each node and a function, where it receives an input from the son and returns an output to the parent, and the root node returns the final output of the program.

Genetic programming can be represented by a tree structure, where each node is a function, which receives an input from each child and returns an output to the parent, and where the root node returns the final output of the program. Thus PG follows the following steps:

- Initial population: it is the initial step of the genetic algorithm is the creation of an initial population of individuals. In this work the initial population is created in a random way.
- Fitness Function : it is the evaluation of the performance of each individual of the population, measuring the how much the result of that species is better compared to the others.
- Selection: it select the individuals with better aptitude from the previous population to generate the population. These individuals may pass through reproduction, crossover or mutation.
- Reproduction: choose the individuals with better aptitude from the previous population to generate the next population. These individuals may pass through reproduction, crossover or mutation.

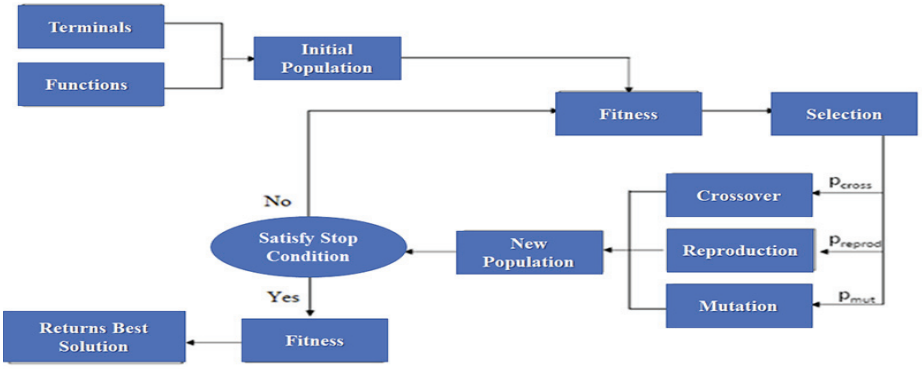


Fig. 3. Diagram of execution of a genetic programming.

- Crossover: each gene can be exchanged between two individuals, according to a uniform probability, generating two son.
- Mutation: is small probability of one or more random genes being exchanged for any other, generating a new individual slightly modified.
- New population: from the processes of reproduction, crossover and mutation a new population is generated.
- Satisfy Stop Condition: stop criteria are checked, which can contain: maximum number of iterations, so that the program can stop (mandatory condition); rate of population, stopping the program if it is not able to vary the population further.
  - If the stopping criterion is satisfied, the aptitude test must be performed, returning the best individual of the current population.
  - If it is not satisfied you should return to the second step.

The Figure 3 presents the order of execution of these steps, presented the probabilities of crossover P_{cross}, probability of reproduction P_{reprod} and probability of mutation P_{mut}. Some genetic programming do not use the reproduction step, since the crossover is similar in some respects to the first one, changing genes between two individuals [20].

Implementation

In this work, a C-library (Lil-GP) was used, making it possible to create algorithms that work in 8-bit microcontrollers, as proposed in this work. The use of the C language is also important, since it is used to program in the main microcontrollers. With the objective of embarking the genetic programming in the microcontroller, modifications were made. The following are the changes in the library:

- Adaptation so that all data operated by the cryptographic algorithms were 8-bits, through the type unsigned char. Although the DATATYPE variable type allows you to change most of the library, some parts, such as in preview and printing, needed to be adjusted in code.
- Parameter creation limiting the minimum number of nodes of the trees generated. This was created to prevent the generated programming from being trivial, such as simple assignments or denials, and to fit only performance parameters, not security parameters.
- Presentation of new fitness variables to better visualize the values of each plots of more complex fitness functions.
- Creations of variable weights to allow automatic execution of several genetic algorithms, each with different weight in the fitness functions.

In this work we used the coefficient of correlation in Fitness Function of genetic programming, where this metric is a measure of similarity between two variables [21, 22, 23]. If the two variables are an image and its encryption, then they are in perfect correlation (that is, coefficient of correlation equals one) if they are highly dependent (identical). In this situation, the encryption process failed to hide the details of the original image. If the correlation coefficient is zero, then the original image and the encrypted image are totally different, that is, the encrypted image has no recognizable feature and is highly uncorrelated from the original image. If the coefficient of correlation equals -1, this means that the encrypted image is the negative of the original image. The parameters used in the genetic programming were: a population of 200 individuals, 8192 test samples with 100 generations. Two weights were used for variable balancing of the fitness formula, varied ten times for each, thus creating the equivalent of 100 individual tests. A total of four test batteries were made, totaling 400 samples.

The evaluation function used in this phase follows as defined by:

$$(1) Fit = \left(\frac{T_N}{100} + \left(\Delta R \frac{MIDR}{MIDR_B} + \Delta C \frac{CC_S}{CC} \right) \right),$$

where ΔR and ΔC are the variation weights of the MIDR and coefficient of correlation parameters, CC_S is the lowest correlation coefficient between the algorithms studied for each image, and $MIDR_B$ is the Maximum and Irregular Deviation Ratio of best individual. This variation has as main objective to find the relation between the two metrics, a balance of difficult balance. It was noticed that small variations (with difference of 0.1 between the weights) did not present significant changes in the results, so variations were used in the order of 10^{-N}, with N being the execution index of each of the genetic programming individually.

Results Obtained

In this section presents algorithm obtained through computational simulation. In the development of the algorithm the Maximum and Irregular Deviation Ratio, and Correlation Coefficient were considered, the algorithm obtained through the proposed methodology of this work was named CRYSEED2. The execution time was used for the fitness function in genetic programming. The calculation of the occupied memory was calculated after the final results, because a reliable calculation of the occupied memory through the structures of trees created by genetic programming is not feasible. To compensate for this problem, we used a 50-node number limitation on the tree. Another factor that also helps to compen-

sate for this limitation is the use of runtime optimization due to the relationship of small algorithms to runtime. Also in this section a performance analysis is performed of the algorithm obtained.

The sets of functions used in this work were based on the functions used in the work [24]. Randomly generated constants were also added. The evaluation function used in this phase is given by:

$$Fit = \left(0,2T_N + \frac{0,4DM}{DM_M} + \frac{0,4DI_M}{DI}\right). \quad (2)$$

After 20 interactions, the PG converged to an algorithm with a tree function $(add(shift\ a_{10}\ a_{12})(xor\ a_5\ a_2))$. In Algorithms 1 and 2 they present the pseudocodes of the encryptor and decryptor CRYSEED2, respectively.

Algorithm 1 Encryption Pseudocode CRYSEED2

for $t = 0$ to number of rounds **do**
  for $j = 0$ to 16 **do**
    $v_1(j) = v_2(j) \wedge$ round key
  end for
  for $k = 0$ to 16 **do**
    for $j = 0$ to 16 **do**
      $v_2(j) = v_1((k+j)\%8)$
    end for
    $T_1 = v_2(10) \ll v_2(12)$
    $T_2 = (v_2(5) \wedge v_2(2))$
    $v_1(k) = (T_1 + T_2) \wedge v_2(0)$
  end for
  for $j = 0$ to 8 **do**
    $v_2(j) = v_1(j)$
  end for
end for

Algorithm 2 Pseudocode of decryptor CRYSEED2

for $t = 0$ to number of rounds **do**
  for $j = 0$ to 16 **do**
    $v_1(j) = v_2(j) \wedge$ round key
  end for
  for $i = 1$ to 16 **do**
    for $j = 0$ to 16 **do**
      $v_2(j) = v_1((k+j)\%8)$
    end for
    $T_1 = v_2(10) \gg v_2(12)$
    $T_2 = (v_2(5) \wedge v_2(2))$
    $v_1(k) = (T_1 - T_2) \wedge v_2(0)$
  end for
  for $j = 0$ to 8 **do**
    $v_2(j) = v_1(j)$
  end for
end for

Table 1 presents the execution time of the obtained algorithm and the reference algorithms. Comparing the results presented in Table 1, one can verify that the algorithm proposed in this work has a processing time reduction of 28%. In particular, it can be seen that CRYSSED2 has a processing time reduction of 84% over XXTEA. These times demonstrate the low computational cost of the proposed algorithm when compared with the reference algorithms, thus increasing the energy efficiency of the wireless sensor devices. It is important to note that the tests were performed on a computer with an Intel Core i5-2520M 2.5 GHz processor, with 8 GB of RAM DDR3-1333 SDRAM, Windows 7 Ultimate 64-bit operating system.

Table 1. Cryptographic Algorithm Execution Time.

Algorithm	Execution time (μs)		
	Encryption Time	Decryption time	Total time
AES	0.708	0.706	1.414
XXTEA	1.1097	1.097	2.294
Cryseed	0.2718	0.22	0.4918
Cryseed2	0.175	0.175	0.35

The level of quality of an encryption depends on how much greater the difference between the original data and the encrypted data, and even if it spreads the data in an irregular way, therefore a good cryptographic algorithm must be able to generate large values of D_M (maximum deviation) and small D_I values (irregular deviation). In this context, a performance analysis of the proposed algorithm through the MIDR, where the MIDR is given by:

$$MIDR = \left(\frac{D_M}{D_I} \times \frac{DI_M}{DM_M}\right), \quad (3)$$

where DM_M and DI_M are the maximum and irregular deviations of the best individual.

The images used for the comparative test among the algorithms were: lena, the most used image in scientific articles; cameraman, used for having shades of gray and light and dark areas; mandrill, an image with high contrast of high frequency colors; testpat, an image with squares of various sizes of various shades of gray and in the center the image lena. These images are shown in Figure 4. All images were obtained from The USC-SIPI Image Database [25]. Tests with other images were also performed to obtain more significant results from the proposed technique, these images are illustrated in Figure 5. The results of this encryption were used to feed the adjust the importance of each metric in the quality assessment cryptographic. In the figure below we can observe the images used in encryption.

Table 2 show the results obtained through simulations the performance of the proposed algorithm and the reference algorithms, in terms of MIDR, each value of the images refers to the Maximum and Irregular Deviation Deviation of each algorithm related to the figures. It is observed that CRYSSED 2 had a considerable gain over other reference algorithms such as AES and XXTEA. It is observed that this gain of CRYSEED2 in the MIDR occurs through its encryption of the proposed algorithm, showing that the encrypted algorithm is superior to its predecessors comparing the means MIDR obtained by the reference algorithms, thus becoming the best algorithm analyzed in this work demonstrating the metric Coefficient of Correlation applied was better in the cryptographic analysis of the images.

Coefficient of Correlation (CC) is a measure of similarity between two variables [22, 23, 21]. If the two variables are an image and its encryption, then they are in perfect correlation (that is, CC equals one) if they are highly dependent (iden-

tical). In this situation, the encryption process failed to hide the details of the original image. If the correlation coefficient is zero, then the original image and the encrypted image is totally different, that is, the encrypted image has no recognizable feature and is highly uncorrelated from the original image. If the CC equals -1, this means that the encrypted image is the negative of the original image. The CC is measured by:

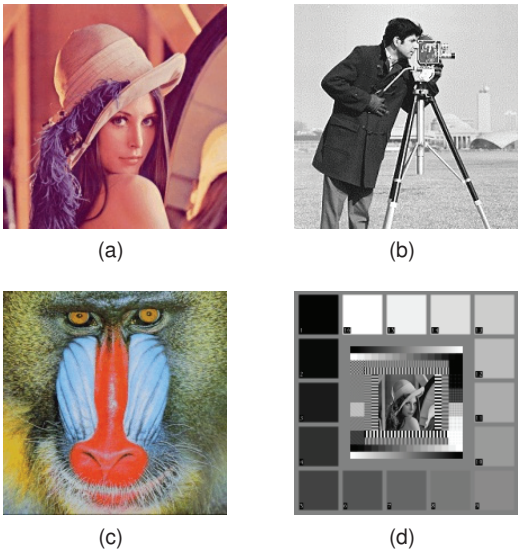
$$cc = \frac{cov(x,y)}{\sigma_x \sigma_y} = \frac{\sum_{i=1}^{N}(x_i - E(x))(y_i - E(y))}{\sqrt{\sum_{i=1}^{N}(x_i - E(x))^2}\sqrt{\sum_{i=1}^{N}(y_i - E(y))^2}} \tag{4}$$

Where $E(x) = \frac{1}{N}\sum_{n}^{i=1} x_i$, and x and y are the gray values of the original and encrypted image pixel. The success of the encryption process means lower CC values

The standardized CC is given by:

$$CC_N = \frac{CC_M}{CC} \times 100 \tag{5}$$

Where CCN is the normalized value in percentage and CCM is the lowest correlation coefficient between the algorithms studied for each image.



(a) (b) (c) (d)

Fig. 4. Standard images: (a) lena; (b) cameraman; (c) mandril (d) testpat.

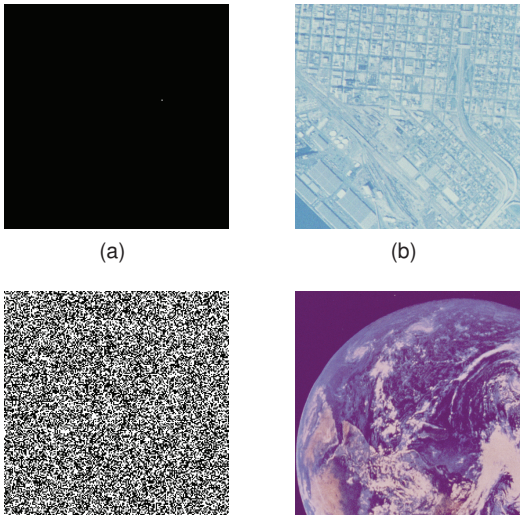
Table 2. Maximum and Irregular Deviation Ratio.

Algorithm	Maximum and Irregular Deviation Ratio							
	lena	cman	mandrill	testpat	black	arial	random	earth
AES	1.24	1.93	1.86	1.67	0.58	0.80	0.58	1.50
XXTEA	0.89	2.19	2.67	1.04	0.25	0.79	0.34	2.46
CRYSEED	1.14	2.62	2.05	1.37	0.34	0.65	0.25	1.77
CRYSEED2	1.12	2.61	2.06	1.39	0.34	0.65	0.34	1.81

The results obtained in the computational simulations with the objective of measuring the correlation coefficient can be seen in 3. It should be noted that the results obtained with the images lena, black, arial and mandrill were the better results.The proposed technique obtained good gains in the correlation coefficient showing that the analyzed images are different from the originals,since CC is negative. So it shows the efficiency of the proposed technique.

Table 3. Coefficient of Correlation.

Algorithm	Coefficient of Correlation							
	lena	cman	mandrill	testpat	black	arial	random	earth
AES	0.14	−0.01	0.005	−0.05	−0.27	−0.10	−0.07	−0.02
XXTEA	0.014	−0.002	−0.003	−0.02	0.02	0.002	−0.001	0.01
CRYSEED	−0.1	−0.18	−0.07	−0.10	−0.19	−0.03	0.01	−0.05
CRYSEED2	−0.16	−0.17	−0.07	−0.08	−0.20	−0.03	0.01	−0.08



(a) (b) (c) (d)

Fig. 5. Standard images: (a) black; (b) arial; (c) random (d) earth.

The Table 4 presents the performance of the algorithms analyzed in terms of occupied memory, in which one can analyze the gain of the proposed algorithm, specifically if we compare CRYSEED2 in relation to AES.

Table 4. Cryptographic Occupied memory.

Algorithm	Occupied memory (Bytes)
AES	$9,437$
XXTEA	$1,570$
CRYSEED	$1,078$
CRYSEED2	$1,078$

Conclusions

The genetic programming methodology was applied and improved for the development of cryptographic algorithm for 8-bit devices, widely used in wireless sensor networks. The CRYSEED2 algorithm was automatically developed from the use of the lil-GP library, adapted to find an efficient cryptographic algorithm for 8-bit devices with high-quality encryption. Also shown performed comparative tests between the algorithm proposed in relation to widely used algorithms literature, in which the following comparison metrics were used: runtime and Maximum and Irregular Deviation Ratio, space occupied in memory and correlation coefficient. The obtained results demonstrate that CRYSEED2 is competitive in comparison to the other presented algorithms. Making possible the insertion of the proposed algorithm in wireless sensor networks, since CRYSEED 2 has low computational complexity and good cryptographic performance. Finally, these results help to prove that genetic programming can create efficient cryptographic algorithms and is a significant advance in the diffusion of artificial intelligence techniques.

***Authors**: Msc. Xiankleber C. Benjamim, Brazil, E-mail: xiankleber@dca.ufrn.br: Msc. Felipe O. S. Gama, Brazil, E-mail:felipe.gama@dca.ufrn.br: Dr. Rodrigo S. Semente, Brazil, E-mail: rodrigo.semente@ufesa.edu.br: Dr. Elmer R. L. Villarrea, Brazil, E-mail: Dr. Andrés O. Salazar, Brazil, andres@dca.ufrn.br*

REFERENCES

[1] Shim, K.A. A Survey of Public-Key Cryptographic Primitives Issues in Wireless Sensor Networks **2012**.

[2] IEEE Standards 802.15.4. Technical report, The Institute of Electrical and Electronics Engineers, Inc., New York, USA., 2003.

[3] ZigBee Specification Version 1.0. Technical report, ZigBee Aliance, 2004.
[4] IEEE 802.15.4e - IEEE Standard for Local and metropolitan area networks–Part 15.4: Low-Rate Wireless Personal Area Networks (LR-WPANs) Amendment 1: MAC sublayer. Technical report, The Institute of Electrical and Electronics Engineers, Inc., New York, USA., 2012.
[5] Biryukov A., Dunkelman O., Keller N., Khovratovich D., Shamir A.: Key recovery attacks of practical complexity on AES-256 variants with up to 10 rounds **2010**. pp. 299–319.
[6] Biryukov, A., Großschädl J.: Cryptanalysis of the full AES using GPU-like special-purpose hardware. *Fundamenta Informaticae* **2012**, *114*, 221–237.
[7] Rinne S., Eisenbarth T., Paar C.: Performance analysis of contemporary light-weight block ciphers on 8-bit microcontrollers. Ecrypt Workshop SPEED. Citeseer, 2007, pp. 33–43.
[8] Meiser G., Eisenbarth T., Lemke-Rust K., Paar C.: Efficient implementation of eSTREAM ciphers on 8-bit AVR microcontrollers. Industrial Embedded Systems, 2008. SIES 2008. International Symposium on. IEEE, 2008, pp. 58–66.
[9] Polimón J., Hernández-Castro J.C., Estévez-Tapiador J.M., Ribagorda A.: Automated design of a lightweight block cipher with Genetic Programming. *International Journal of Knowledge-based and Intelligent Engineering Systems* **2008**, *12*, 3–14.
[10] Estevez-Tapiador J.M., Hernandez-Castro J.C., Peris-Lopez P., Ribagorda A.: Automated Design of Cryptographic Hash Schemes by Evolving Highly-Nonlinear Functions. *Journal of Information Science & Engineering* **2008**, *24*.
[11] Picek S., Carlet C., Jakobovic D., Miller J.F., Batina L.: Correlation immunity of boolean functions: an evolutionary algorithms perspective **2015**. pp. 1095–1102.
[12] Semente R.S., Salazar A.O., Oliveira F.D.: CRYSEED: An automatic 8-bit cryptographic algorithm developed with genetic programming. Instrumentation and Measurement Technology Conference (I2MTC) Proceedings, 2014 IEEE International. IEEE, 2014, pp. 1065–1068.
[13] Daemen J., Rijmen V.: AES proposal: Rijndael **1999**.
[14] Wheeler D.J., Needham R.M.: TEA, a tiny encryption algorithm. International Workshop on Fast Software Encryption. Springer, 1994, pp. 363–366.
[15] Needham R.M., Wheeler D.J.: Tea extensions. *Report (Cambridge University, Cambridge, UK, 1997) Google Scholar* **1997**.
[16] Wheeler D.J., Needham R.M.: Correction to xtea. *Unpublished manuscript, Computer Laboratory, Cambridge University, England* **1998**.
[17] Gaba S., Aggarwal I., Pandey S.: Design of efficient XTEA using Verilog. *International Journal of Scientific and Research Publications* **2012**, *2*.
[18] Zafar F. *Tiny encryption algorithm for cryptographic gradient noise*; University of Maryland, Baltimore County, 2010.
[19] Yarrkov E. Cryptanalysis of XXTEA. *IACR Cryptology ePrint Archive* **2010**, *2010*, 254.
[20] Koza J.R. Hierarchical Genetic Algorithms Operating on Populations of Computer Programs. IJCAI, 1989, pp. 768–774.
[21] Ahmad J., Ahmed F.: Efficiency analysis and security evaluation of image encryption schemes. *International Journal of Video and Image Processing and Network Security* **2010**, *23*, 25.
[22] Ahmed H.E.d.H., Kalash H.M., Allah O.F.: Encryption efficiency analysis and security evaluation of RC6 block cipher for digital images. Electrical Engineering, 2007. ICEE'07. International Conference on. IEEE, 2007, pp. 1–7.
[23] El Fishawy N.F., Zaid O.M.A.: Quality of encryption measurement of bitmap images with RC6, MRC6, and Rijndael block cipher algorithms. *IJ Network Security* **2007**, *5*, 241–251.
[24] Hernandez-Castro J.C., Estevez-Tapiador J.M., Ribagorda-Garnacho A., Ramos-Alvarez B.: Wheedham: An automatically designed block cipher by means of genetic programming. Evolutionary Computation, 2006. CEC 2006. IEEE Congress on. IEEE, 2006, pp. 192–199.
[25] SIPI, U. The usc-sipi image database, 2016.