IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received October 27, 2018, accepted November 29, 2018, date of publication January 9, 2019, date of current version January 23, 2019.

Digital Object Identifier 10.1109/ACCESS.2018.2886384

A Low Power Cryptography Solution Based on Chaos Theory in Wireless Sensor Nodes

MASOUMEH SHARAFI[1], FARANAK FOTOUHI-GHAZVINI[1], MOHSEN SHIRALI[2], (Member, IEEE), AND MONA GHASSEMIAN[2], (Senior Member, IEEE)

[1]Department of Computer Engineering and Information Technology, University of Qom 3716146611, Qom, Iran
[2]School of Computer Science and Engineering, Shahid Beheshti University, Tehran 1983969411, Iran

Corresponding author: Faranak Fotouhi-Ghazvini (f-fotouhi@qom.ac.ir)

ABSTRACT Unobtrusive personal data collection by wearable sensors and ambient monitoring has increased concerns about user privacy. Applying cryptography solutions to resource constraint wireless sensors as one of the privacy-preserving solutions demand addressing limited memory and energy resources. In this paper, we set up testbed experiments to evaluate the existing cryptographic algorithms for sensors, such as Skipjack and RC5, which are less secure compared to block cipher based on chaotic (BCC) on existing IEEE802.15.4 based SunSPOT sensors. We have proposed modified BCC (MBCC) algorithm, which uses chaos theory characteristics to achieve higher resistance against statistical and differential attacks while maintaining resource consumption. Our comparison observations show that MBCC outperforms BCC in both energy consumption and RAM usage and that both MBCC and BCC outperform RC5 and Skipjack in terms of security measures, such as entropy and characters frequency. Our comparison analysis of MBCC vs BCC suggests 13.44% lower RAM usage for encryption and decryption as well as 6.4 and 6.6 times reduced consumed time and energy for encrypting 32-bit data, respectively. Further analysis is reported for increasing the length of MBCC key, periodical generation of master key on the base station and periodical generation of round key on the sensors to prevent the brute-force attacks. An overall comparison of cipher techniques with respect to energy, time, memory and security concludes the suitability of MBCC algorithm for resource constraint wireless sensors with security requirements.

INDEX TERMS Chaos, data security, encryption, performance evaluation, prototypes, wireless sensor networks.

I. INTRODUCTION

In smart environments, information can be collected by ambient or wearable sensors. Unobtrusive data collection, processing and sharing raise concerns in the privacy field [1]. On the other hand, in their lifetime, these sensors face challenges such as limited energy and memory, and unreliable communication. Hence, in [2], studies have been conducted to find an efficient energy protocol for data collection in smart homes. In addition, an adaptive collection algorithm for data acquisition by wearables was designed to reduce energy consumption [3]. The most critical problem in designing such networks is ensuring secure data collection [4]. For instance, in smart buildings, a large amount of personal data is collected by a variety of heterogeneous devices. Using the collected data, there is the possibility of tracking the daily habits of individuals or care for the elderly [2], [5]. Privacy should ensure that the data (or part of the data) is only accessed by the authorized users [5], [6]. Considering the resource constraints in such networks, employing data collection techniques and conventional cryptography algorithms for privacy-preserving solutions are not considered to be appropriate. Considering the constraints on power consumption in wireless sensors, the low power encryption algorithms such as RC5 [7] and Skipjack [8] have been introduced. In order to reduce the algorithm complexity, however, the security metrics in these algorithms tend to be weaker in comparison with other algorithms such as AES [9], [10] and Block Cipher based on Chaotic (BCC) [11]. On the other hand, due to the sensitivity of the data collected by sensors for monitoring or controlling personal or ambient systems, security is a crucial requirement.

In this work, we have set up a testbed to implement and compare the energy efficiency and complexity as well as the security performance to experiment the existing

TABLE 1. List of studied block ciphers algorithm (extracted from [18]).

Algorithm	*Year*	*Key Size (bits)*	*Block Size (bits)*	*Round Count*	*Algorithm Structure*
Twine	2013	80/128	64	32	FN
LEA	2013	128/192/256	128	24/28/32	FN
BCC	2012	64	128	12	SPN
Present	2007	80/128	64	31	SPN
Hight	2006	128	64	32	FN
Skipjack	1999	80	64	32	FN
AES	1998	128/192/256	128	10/12/14	SPN
RC5	1995	128/192/256	128	10/1214	FN

cryptographic algorithms for sensors, namely Skipjack and RC5 and BCC using physical SunSPOT (Sun Small Programmable Object Technology) sensors [12]. To address the resource limitations of existing protocols, we aim to propose an algorithm based on chaos theory to achieve higher resistance against statistical and differential attacks while maintaining resource consumption according to the sensor limitations. Thus, in this paper, we propose and evaluate an encryption method with a focus on its energy consumption and security criteria. Chaos encryption [13], which is based on chaos theory, embodies the features of high sensitivity to initial conditions and control parameters, random-like behaviors, and simplicity [14], [15]. In chaos-based systems, a slight change in the initial state and parameter configurations after a few iterations, the system leads to enormous, significant and unpredictable variations in the final state [16], [17]. This characteristic of chaos makes this solution very suitable for encryption. In this paper, MBCC aims to improve BCC in terms of security and energy consumption. It is based on the chaos theory for power-limited wireless sensor devices. In the following, MBCC, along with BCC, RC5, and Skipjack are implemented on the SunSPOT and their performance and security metrics are measured. Moreover, the periodical key change scenario in MBCC is analyzed. The acquired results, exhibit a better performance metrics of MBCC in comparison with BCC, and a superior security metrics of MBCC compared to those of RC5, and Skipjack.

The rest of this paper is structured as follows: Section II reviews the existing block encryption algorithms and encryption algorithms based on chaos theory applicable for wireless sensor environments. Section III describes the design of the proposed MBCC algorithm. Section IV presents the performance and security analysis for the implemented algorithms and the comparison considering the periodical key change in MBCC with respect to the security. Finally, Section V concludes the paper and suggests future work.

II. RELATED WORKS

In this section, we present related existing encryption algorithms and provide an intuitive comparison between them. Furthermore, we study the appropriateness of chaos theory to improve encryption algorithms used for resource constraint sensors.

A. LOW RESOURCE ALGORITHMS

A number of block encryption algorithms are proposed suitable for wireless sensor networks set up in resource constraints environments. These algorithms typically use a fixed or variable key size, block size of 64 or 128 bits, and based on a Feistel network (FN) or substitution-permutation network (SPN) structure. A summary of low resource algorithms is extracted from [18] and are listed in Table 1.

Software implementation analysis of these algorithms is, typically based on the complexity of an algorithm using a combined space complexity and time complexity. Based on this start point, code size and random-access memory (RAM) size are used to describe the occupancy of the microcontroller's space. Furthermore, processing time is defined as the number of processor's cycles to deal with one block [19]. A summary of the advantages and disadvantages of low resource algorithms with respect to different security attacks are presented in Table 2.

For a more accurate comparison of the proposed algorithm (MBCC), with other algorithms under identical conditions, the software implementation of algorithms was performed on a single sensor. However, since it was not possible to compare the proposed method with all of the algorithms in Table 1, only three algorithms were selected for implementation and comparison. Meanwhile, the BCC algorithm, due to its similarity to the proposed algorithm, and RC5 and Skipjack due to many references to them in different articles (especially regarding security) and use them in wireless sensor network security protocols has been chosen. In addition, Skip and RC5 are considered to be the most suitable algorithms for WSNs [20].

B. CHAOS-BASED SOLUTIONS

Symmetric chaos encryption is performed in two forms; Stream-based and block-based [11]. Stream-based cipher uses a random-like (pseudo-random) number attained from the chaos map for plaintext encryption. In block-based encryption, the plaintext is divided into blocks, which employs encryption operations such as substitution and shift rotate. Substitution boxes are considered the main core of the block-based cipher and are vastly used in all regular block encryptions such as DES and AES [21]. These boxes play

TABLE 2. Some advantage and disadvantage of studied block ciphers algorithm (Revised from [18], [23]–[34]).

Algorithm	*Advantage*	*Disadvantage*
Twine	Suitable performance in w.r.t code size, processing time, low implementation complexity	Vulnerable to differential fault analysis, meet-in-the-middle attack
LEA	Suitable performance in processing time Tiny code size	Vulnerable to the side-channel analysis attack
BCC	Use the features of chaos theory High security	High RAM , ROM memory consumption High energy, time consumption Inadequate key length
Present	Suitable performance in RAM memory consumption Suitable performance w.r.t code size	Vulnerable to differential attacks, key-recovery attack, statistical saturation attack
Hight	Suitable performance w.r.t code size	Vulnerable to Related key attack
Skipjack	Suitable performance in processing time	Low security Vulnerable to the brute-force attack Inadequate key length
AES	Suitable performance energy and processing time	Vulnerable to differential fault analysis, meet-in-the-middle attack Large code and data memory sizes
RC5	Suitable performance in code memory and data memory	Vulnerable to differential attacks, timing attack Not supported by embedded systems such as Intel architecture Poor key agility[1]

a significant role as non-linear transforms. Appropriate substitution boxes prompt algorithm resilience against external attacks such as statistical or differential attacks [14].

The security level of the chaotic block cipher is fixed by the properties and the implementation method of chaos function. In this encryption, chaos maps are used to generate keystream or substitution boxes [15]. RC5 [22] is a symmetric encryption algorithm with variable key size, block size, and round count. This algorithm employs a few data iterations, small-size tables and simple operations of addition, shift, and XOR [7], [17].

The SPINS security protocol in wireless sensor networks uses an optimized version of RC5. The Skipjack algorithm is an algorithm based on an unbalanced Feistel network [35], with an 80-bit key size, 32 rounds, and 64-bit blocks. Since the key size is 80 bits only, the brute force attack to Skipjack tends to be easy [8]. This algorithm is employed as an encryption-based method in TinySec and MiniSec protocols [20], [36]. Employing Skipjack is recommended for applications requiring lower levels of security [37].

The chaos maps are often based on the floating-point computations [16], [38], which demand greater precision and energy, hence inappropriate for systems with resource constraints such as wireless networks. In some of the proposed methods, chaos maps along with the floating-point computations are also used prior to the encryption process, e.g., key generation process [39]. In the central core of the chaos method algorithms, Feistel structure is employed. Due to the floating-point computations' overhead, discrete chaos maps have substituted the usual chaos maps [7], [14]. In case of using the discrete map, the chaos sequence period will be shortened; hence the risk of statistical attacks is increased.

Table 3 shows the characteristics of a number of chaos-based encryption algorithms in wireless sensors that are compared in two dimensions of performance and security. The article [38] and [39] were implemented and [7], [14], and [16] were simulated to evaluate the algorithms. In literature for performance evaluation, the memory, time and energy consumption of algorithms are often considered for investigation. Meanwhile, in the security evaluation, a variety of analysis such as statistical and differential analysis examine the algorithms resilient to cryptographic attacks. Besides, the long key length and the large block size are considered as the advantages of the encryption algorithms. Accordingly, all algorithms except the work conducted by Mansour *et al*. [38] where the key length is set to 94 bits, have a sufficient key length against attacks such as brute-force attack. Also, the algorithm's block size is appropriate except for [7] and [14]. The larger the block length is, the less access to encrypted instances is possible. In addition, proposed algorithms by Xiao-Jun *et al.* [14] and Tong *et al.* [7] are based on the Feistel structure using a similar circuit by reordering the sequence of round keys for both encryption and decryption process which can be considered as one of the advantages of

[1] Ability to change keys quickly with a minimum amount of resources.

TABLE 3. The comparison of the chaos based cipher algorithms in wireless sensor networks.

Research	*Tong et al.[7]*	*Xiao-Jun et al.[14]*	*Wang et al.[16]*	*Mansour et al.[38]*	*Biswas et al.[39]*
Chaos System	Logistic, Cubic	Tent	Logistic-Tent	PWLCM (Piecewise linear chaotic map)	N-Logistic-Tent
Key Size (bits)	128	128-160	≈ 128	94	448
Block Size (bits)	32	32	128	128	128
Analysis Environment	Simulation	Simulation in CC2430	Simulation	TelosB/Matlab	Mica2
Performance Evaluation	Speed, memory	Time, memory	Time, memory	Speed, memory, energy	Speed, memory, time
Security Evaluation	Statistical analysis (balance, character frequency), entropy analysis, diffusion-confusion analysis	Key space, diffusion-confusion analysis, statistical analysis (uniform distribution, characters frequency), entropy analysis	Key space, statistical analysis (histogram)	Key space, statistical analysis (histogram), differential attacks (# of pixel changing rate and unified averaged changed intensity)	Key space, statistical analysis (histogram, correlation), differential attacks (# of pixel changing rate and unified averaged changed intensity)

such algorithms. Also, Mansour *et al.* [38] algorithm contains two chaos maps and one XOR which the algorithm benefits from the usage of an identical circuit for encryption and decryption process is required. While, in Wang [16] and Biswas *et al.* [39], the decryption process is the inverse of encryption, they require two separate circuits for encryption and decryption processes.

III. THE PROPOSED ALGORITHM: MODIFIED BCC (MBCC)

In the wireless sensor networks, time and memory usage are just as profoundly important as security.

In this research, the time and energy consumption factors are decreased while preserving the security measures of the algorithm. The proposed algorithm is aimed to design an encryption method with lower complexity than BCC. It has minimal resource requirements suitable for constrained wireless devices while maintaining more resilience to the attacks compared to RC5 and Skipjack. Therefore, a chaos-based symmetric block encryption algorithm is proposed with a substitution-permutation structure. The proposed Modified BCC (MBCC) applies improved BCC, which is introduced in 2012 for encryption in wireless sensor networks. BCC is a symmetric block encryption algorithm with substitution-permutation structure, and a 64-bit- key size. It is designed based on chaos theory where chaos is used for generating the permutation table [11]. By decreasing the number of permutations and increasing the size of the cipher key in MBCC, improvements have been achieved in performance and security metrics of the algorithm, respectively. In the following subsections, the details of BCC and MBCC algorithms are further delineated.

A. BCC ALGORITHM

The BCC encryption process is conducted in four steps. In the first step, a $2r$ number of 64-bit sub-keys are generated through the round-key generation method where r is the number of algorithm rounds. In each round, a pair of round-keys is used to encrypt the two halves of the block. In the second step, the input of each block cipher –which is a 128-bit data –is divided into two 64-bit data blocks, namely the left block (L) and the right block (R). In the third step, L_i and R_i blocks are computed according to (1) over r iterations. In the final step, L_r and R_r blocks from the last r round get swapped which forms the final encrypted data, as follows:

$$\begin{cases} L_i = G\left(L_{i-1} \oplus R_{i-1}, R_{i-1}\right) + K\left[2\left(i-1\right)\right], \\ R_i = G\left(R_{i-1} \oplus L_i, L_i\right) + K\left[2i-1\right] \end{cases} \tag{1}$$

where G is a substitution-permutation network of tables, and $K\left[2\left(i-1\right)\right]$ and $K\left[2i-1\right]$ are the round keys for the left and right 64-bit blocks, respectively. The main core of the algorithm is the G method. The basic operations of encryption - substitution and permutation –, which are used for confusion and diffusion, reside in the G method. **G** is a 3-layer substitution-permutation network in BCC. The input **a**, is a plaintext (i.e., unencrypted) byte and S is substitution box, as shown in Fig. 1.

The round key generation process in BCC (64 bits) is illustrated in Fig. 2 and is implemented according to (2) and (3):

$$\mathrm{S}\left[0\right] = \mathrm{K}\left[0\right] = \mathrm{P} + \mathrm{Q} \tag{2}$$

$$\mathrm{K}\left[\mathrm{i}+1\right] = \mathrm{ROL}\left(\left(\mathrm{S}\left[\mathrm{i}+1\right] + \mathrm{ROL}\left(\mathrm{K}\left[\mathrm{i}\right], \mathrm{Key}\right)\right), \mathrm{S}\left[\mathrm{i}+1\right]\right) + \mathrm{ROL}\left(\mathrm{K}\left[\mathrm{i}\right], \mathrm{Key}\right) \tag{3}$$

where the P and Q variables have constant values, and ROL is the rotation of the dependent data, which is addressed in RC5 according to (4):

$$\mathrm{Rol(x, y)} = \{\mathrm{x} \ll (\mathrm{y\&(w-1)})\} | \{\mathrm{x} \gg (\mathrm{w} - (\mathrm{y\&(w-1)}))\} \tag{4}$$

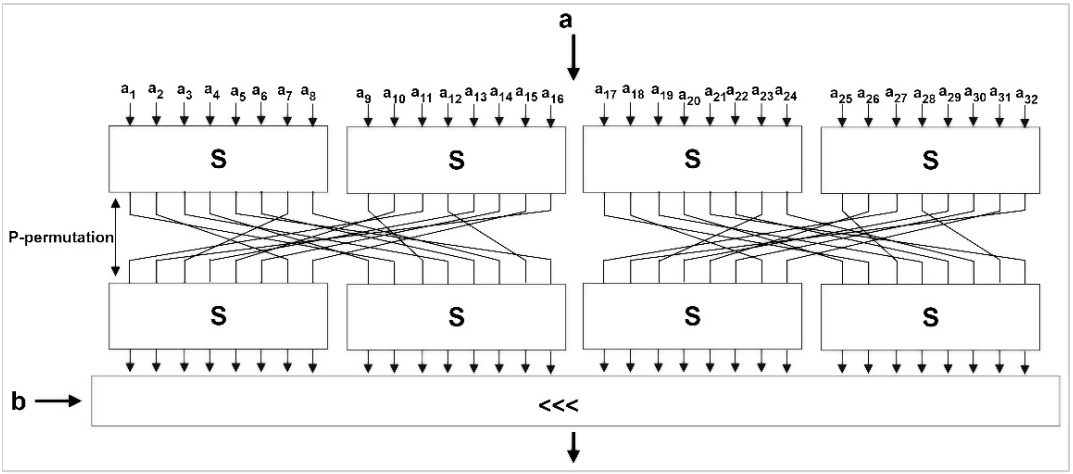

FIGURE 1. The G method in BCC [11].

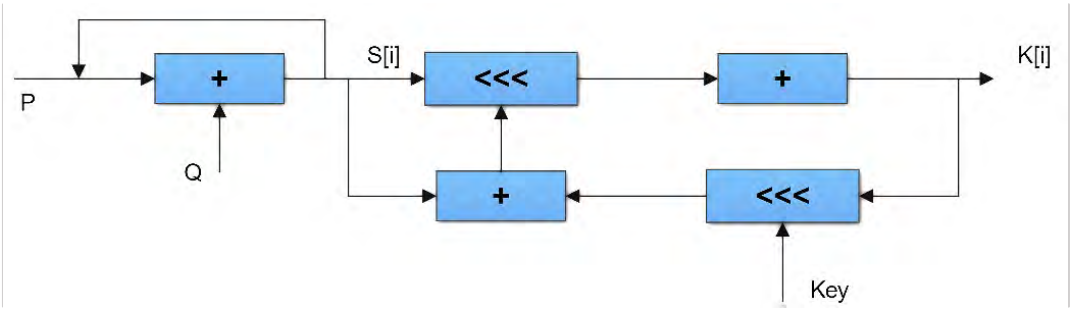

FIGURE 2. Round key generation process in BCC [11].

where & is the symbol for the "*and*" operation and | refers to the "*or*" operation.

B. MBCC ALGORITHM

According to WSN constraints and requirements, the BCC algorithm is supposed to be improved in terms of time and energy consuming. Hence, we try to find a way to optimize the time consuming for data encryption in BCC. For this purpose, we conducted a series of tests and evaluations to find the most time-consuming parts of BCC algorithm as possible options for improvement of algorithm's operation time.

Each round of BCC algorithm contains two G-method calls as a main core of the algorithm which in turn consists of two substitutions, one permutation and one shift-rotate operation. Contrary to the simple and low-cost implementation of bit permutations in hardware [18], the software implementation is expensive from the aspect of processing time. To investigate the effect of permutation in encryption time, we experiment removing the permutation operations from the encryption process in BCC. The results show that the encryption time of 32-bytes data is reduced from 220.74 milliseconds to 15.75 milliseconds. Therefore, permutation tends to be the longest operation in the G-method and by improving this section, the overall time consumed for the algorithm execution will also improve. For this reason, we extend our experiment by reducing the number of permutation operations. Since permutation operations are within the main core of the BCC algorithm, the number of these permutations is directly related to the number of rounds. Thus, to reduce the number of permutations, the best solution is to exclude the permutation from the main core of encryption operation so that the number of permutations is independent of the round number. Subsequently, the permutation operation call is eliminated from G-method as depicted in Fig. 3 and transmitted before the encryption block (i.e., the beginning of the encryption process and the end of the decryption process) as shown in Fig. 4. As a result, by reducing the number of permutations from 2r to 2, a remarkable improvement in the encryption operational time is achieved. For instance, the encryption time of 32-bytes data is observed to be reduced to 33.17 milliseconds, which is 84.97% less than the initial state according to the results presented in Table 4.

Since the reduction of the number of permutations can negatively affect the security coefficient of the proposed algorithm, the next improvement to provide a better algorithm, is to enhance the security. Increasing the key length of the encryption key will increase the complexity and the search space to strengthen the algorithm against brute force attacks.

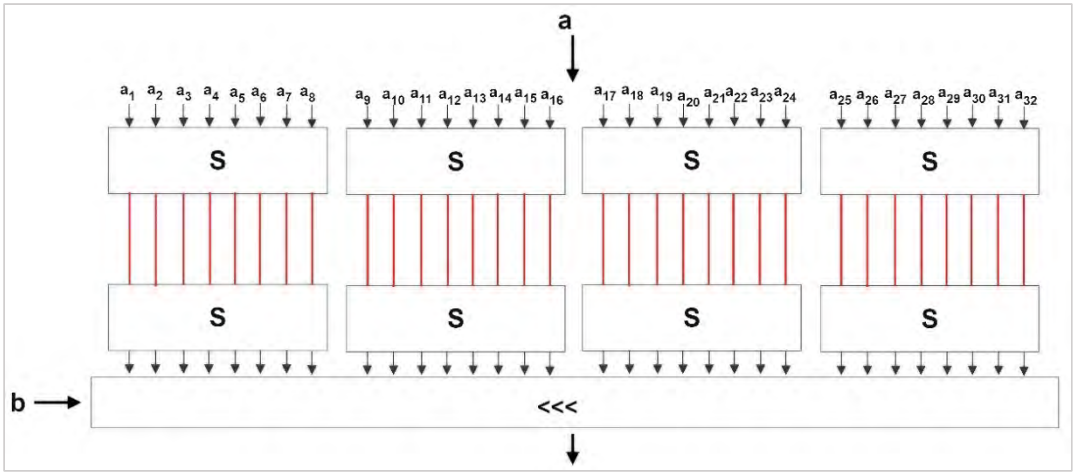

FIGURE 3. The G method in MBCC.

TABLE 4. Require time for Encryption data (32 bytes) in different scenarios with the different number of rounds (millisecond).

Encryption process Scenarios	*2*	*4*	*6*	*8*	*10*	*12*
BCC algorithm	37.93	74.39	111.21	147.81	184.35	220.74
BCC algorithm -without permutation	3.06	5.51	7.91	10.46	13.1	15.75
BCC algorithm - with one permutation	20.51	23.01	25.53	27.94	30.51	33.17
BCC algorithm - with once permutation (key length 128 bit) (MBCC)	20.92	23.48	26.15	28.71	31.36	34.02

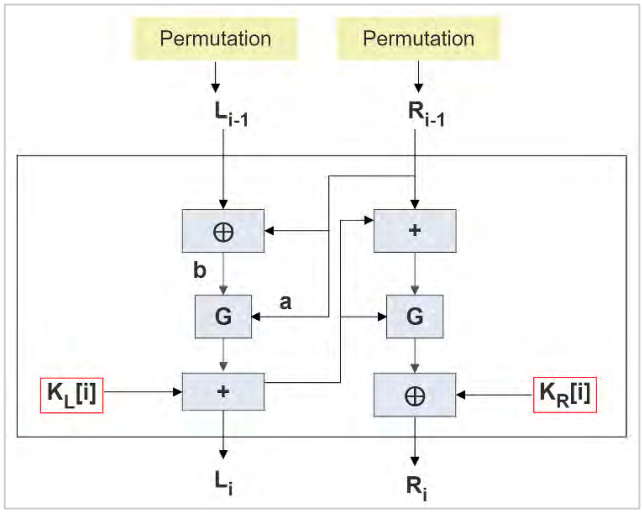

FIGURE 4. The process of one encryption round in MBCC.

In return, increasing the key length will increase the time and energy consumption of the encryption algorithms but it can be affected by the significant improvements of time and energy consumption which is achieved by reducing the number of permutations. The effect of increasing the key length on encryption time is also investigated by increasing key length from 64 bits (which is the insufficient length in BCC) to 128 bits. The evaluations indicate that by increasing key length, time consumption has only increased by 2.5%. Hence, the key length of the proposed algorithm is increased from 64 bits to 128 bits. The results of time consumption investigation are shown in Table 4. However, the round key generation process in MBCC is almost similar to the process of key generation in BCC (Fig. 2) with two differences. With the changes applied to the round-key generation method in the software implementation of MBCC, the size of the master key and the two variables P and Q are increased from 64 bits to 128 bits. Therefore, the generated round keys in MBCC, unlike in BCC, are 128 bits long. Incrementing the length of the round keys results in generating as many round keys in an encryption process, as the number of encryption rounds. In BCC, the length of the round keys is 64-bit which results to twice the number of the algorithm's round count.

The proposed encryption process in MBCC is designed and executed as shown in Fig. 4. The encryption process in MBCC is similar to BCC. However, in MBCC, the permutation operations with the two halves of the plaintext block as input, is called before the block encryption operation. The output of the permutation operation is considered as the input of block encryption operation. In each round of encryption in MBCC, each half of the round key is used for encrypting a half of the data block; this is due to the increased key length to 128 bits. In each round, the two left and right halves of the

TABLE 5. MBCC encryption algorithm.

```
Input: D [128]
Output: Cipher [128]
Dividing Data into two 64-bit blocks as Left and Right;
L ← {D [0], D [1], D [2], D [3], D [4], D [5], D [6], D [7]} ;
R ← {D [8], D [9], D [10], D [11], D [12], D [13], D [14], D [15]};
L ←Permutation (L);
R ←Permutation (R);
for i ←1 to 12 step 2 do
    K_L ←{K[i][0], K[i][1], K[i][2], K[i][3], K[i][4], K[i][5], K[i][6], K[i][7]};
    K_R ←{K[i][8], K[i][9], K[i][10], K[i][11], K[i][12], K[i][13], K[i][14], K[i][15]};
    C_1 ←L⊕R ;
    C_2 ←G(R,C_1);
    L ←C_2 +K_L;
    C_3 ←L+R;
    C_4 ←G(L,C_3);
    R ←C_4 ⊕K_R;
Cipher ←{R[0], R[1], R[2], R[3], R[4], R[5], R[6], R[7], L[0], L[1], L[2], L[3], L[4], L[5], L[6], L[7]};
```

TABLE 6. MBCC decryption algorithm.

```
Input: C [128]
Output: Data [128]
Dividing Cipher into two 64-bit blocks as Left and Right;
L ← {C [0], C [1], C [2], C [3], C [4], C [5], C [6], C [7]} ;
R ← {C [8], C [9], C [10], C [11], C [12], C [13], C [14], C [15]};
for i ←1 to 12 step 2 do
    K_L ←{K[i][0], K[i][1], K[i][2], K[i][3], K[i][4], K[i][5], K[i][6], K[i][7]};
    K_R ←{K[i][8], K[i][9], K[i][10], K[i][11], K[i][12], K[i][13], K[i][14], K[i][15]};
    D_1 ←L⊕R ;
    D_2 ←G_n (R,D_1);
    L ←D_2 -K_L;
    D_3 ←L-R;
    D_4 ←G_n (L,D_3);
    R ←D_4 ⊕K_R;
L ←Permutation (L);
R ←Permutation (R);
Cipher ←{R[0], R[1], R[2], R[3], R[4], R[5], R[6], R[7], L[0], L[1], L[2], L[3], L[4], L[5], L[6], L[7]};
```

block cipher are calculated according to (5) in cryptography:

$$\begin{cases} L_i = G\left(L_{i-1} \oplus R_{i-1}, R_{i-1}\right) + K_L[i], \\ R_i = G\left(R_{i-1} \oplus L_i, L_i\right) + K_R[i] \end{cases} \tag{5}$$

where $K_L[i]$ and $K_R[i]$ are the round keys of the 64-bit left and right blocks, respectively which have been obtained from the 64-bit left and right halves of the 128-bit round key. In addition, the pseudo-codes for the encryption and decryption procedures in the proposed algorithm are shown in Table 5 and Table 6 respectively. In the decryption process of MBCC (Table 6) similar to BCC, the inverse of substitution and permutation tables are used. In addition, the left shift rotate is replaced with the right shift rotate. The addition operation is also replaced with the subtraction. The sequences of round keys in decryption and encryption (Table 5) are reverse, and unlike the encryption, the permutation operation is conducted at the end of the decryption process.

IV. IMPLEMENTATION RESULT ANALYSIS

Cryptographic algorithms are often implemented as hardware modules on sensor nodes; however, methods such as software implementation or hardware/software co-design are considered as suitable solutions for off-shelf nodes with no existing hardware security implementation. Furthermore, since adding new hardware circuitry to off-shelf nodes (namely SunSPOT sensors) is not possible; the encryption algorithms for such sensors can be implemented and evaluated on software. However, software implementations have reached to

TABLE 7. The general properties of the implemented algorithms.

Algorithm	*Key Size (bit)*	*Block Size (bit)*	*Round Count*	Algorithm Structure
RC5	64	64	12	FN
Skipjack	80	64	32	FN
BCC	64	128	12	SPN
MBCC	128	128	12	SPN

more mature performance metrics and measurements. In software implementations, the main design goals are to reduce the memory occupation and to optimize the throughput and power saving of the processor [19]. The parameters set used in our implementations, such as key size, block size and number of rounds of the implemented algorithms in this work, are provided in Table 7.

The structure of the algorithms depends on the applied network type. Substitution-Permutation Network consists of a network that takes blocks of plaintext and keys. SPN applies alternating rounds of substitution layers (aka S-boxes) and permutation layers (aka P-boxes) to produce the final ciphertext. A Feistel cipher or Feistel network is known as the symmetric structure used in the construction of the block ciphers. Such a structure has the advantage of identical or very similar encryption and decryption operations with only a reversal of key schedule. Therefore, the size of the circuitry and the codes are nearly halved [35]. The security of the Feistel structure depends on the applied round function as well as the number of rounds. The simplicity of this structure makes it a suitable candidate for wireless sensor networks [40]. In the following, the analysis and results of comparisons of MBCC, BCC, RC5 and Skipjack will be discussed regarding the performance and security metrics attained.

A. PERFORMANCE METRICS

Performance metrics play an important role when different cipher algorithms are being compared. Consequently, a uniform platform and consistently agreed on metrics are required. Accordingly, we have implemented the proposed MBCC algorithm (Section III.B) and existing algorithms including Skipjack, RC5 and BCC on SunSPOT sensors to compare them.

Due to the memory, battery and the processing resource constraints in wireless nodes, in this study, memory, time, and energy consumption parameters are measured in order to evaluate the overall performance of the implemented encryption algorithms.

1) ENERGY CONSUMPTION

Limited energy is one of the key constraints to be considered in the wireless sensors. Although most nodes are equipped with batteries, they can be located in areas with no chance for recharging. Some nodes even use solar energy [41], but still, batteries are their main resource. The energy consumed in sensors is used for sensing, processing and data transmission. In order to measure the magnitude of the consumed energy, the average instantaneous power consumption value is used in our analysis taken from the execution time of each algorithm.

The power consumption value is calculated by defining the amount of current and voltage. The presented results are based on our hardware testbed implementation using SunSPOT sensors. SunSPOT processor board includes an ARM architecture 32 bit CPU with the ARM920T core running at 180 MHz. It has 512 KB RAM and 4 MB flash memory. A 2.4 GHz IEEE 802.15.4 radio has an integrated antenna and a USB interface is included. The unit uses a 3.7V rechargeable 750 mAh lithium-ion battery, has a 30 uA deep sleep mode, and battery management provided by software [42]. However, based on hardware studies [43], the calculation values by the SunSPOT API are not sufficiently precise; hence we have designed a circuit to calculate power consumption that allows tracking current and voltage consumed in specified periods. For the design of the measurement circuit, the idea presented in [43] has been used. Since we have the voltage and current values, the power consumption is calculated using $P_{(w)} = I_{(A)} \times V_{(v)}$ based on the current and voltage consumption. Accordingly, after receiving the maximum current within the specified period, it is possible to calculate the maximum power consumption, hence the energy consumption using equation 6:

$$E_{(J)} = P_{(w)} \times T_{(s)} \tag{6}$$

where E is the energy consumption value and T is the operation time. It should be noted that the average of the maximum amount of current values over a minimum number of repetition, i.e. 20. is used to calculate the maximum current.

The upper and lower bound for energy consumption of each algorithm was calculated based on $average - CI \leq H \leq average + CI$ where average and CI are mean and confidence interval of the samples which calculated for 95% as the confidence level.

Based on the results presented in Fig. 5, BCC is by far ranked the first with respect to the average energy consumption. In the second place, with quite significantly lower consumption is MBCC. Such tangible difference results from a reduction in the number of permutations and a reduction in the operating time in MBCC than that of BCC. Moreover, in order to evaluate the scenario of periodical key change in MBCC, the energy consumed in generating the round-keys in the sensor node is analyzed. Such that, the averaged power consumption and the energy consumption for generating 12 round-keys in MBCC are 1.3680 W and 17.7690 mJ, respectively.

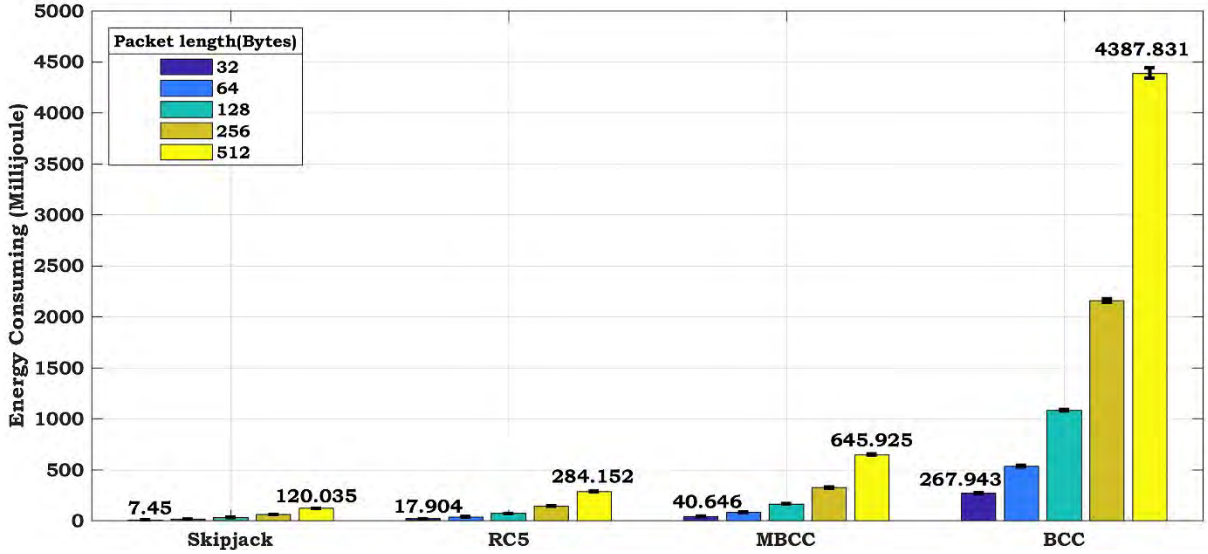

FIGURE 5. The average energy consumption for each encryption algorithm in mJ for data of various sizes.

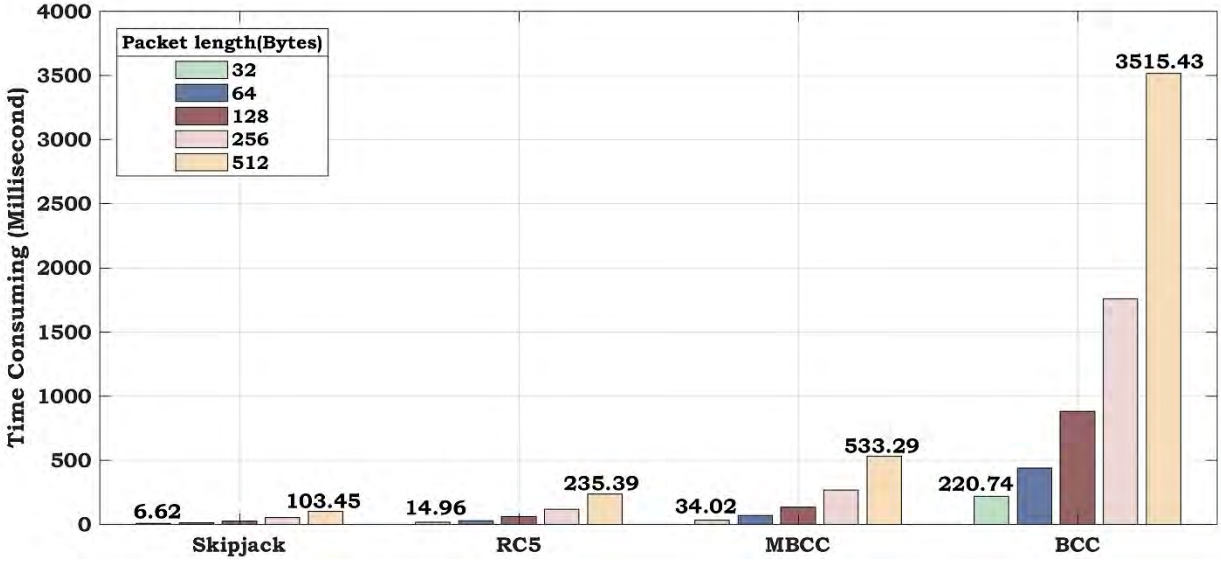

FIGURE 6. The required time for encrypting data considering various sizes in different algorithms (ms).

2) ENCRYPTION TIME

The faster an algorithm runs, the higher its performance is.

According to the results shown in Fig. 6, BCC takes the longest time to carry out the encryption operation while by reducing the number of permutations from 2r to 2, the MBCC algorithm's required time is improved significantly. The RC5 and Skipjack algorithms take less time for encryption in comparison with BCC and MBCC, since they employ the basic operations such as shift and XOR. Their security is, however, questionable, which we will further discuss in this section. Additionally, in the evaluation of the periodical key change scenario in MBCC, the round-key generation time can play a significant role in making decisions. The required time for generating 12 round-keys in MBCC on a sensor node is 13 milliseconds. It is approximately the same as the key generation time for a similar number of round-keys in BCC. The MBCC key generation time is, however, less than RC5 with a key generation time of 29.56 milliseconds. This difference can be justified by the applied complicated key generation in RC5.

3) MEMORY USAGE

The storage capacity in wireless sensors is limited, mostly used for the operating system and the sensed data. Therefore, there is not much memory space left for implementing security algorithms. As a result, the complex encryption algorithms are not appropriate for implementation on the sensor nodes. In this study, the RAM and ROM usage in each one of

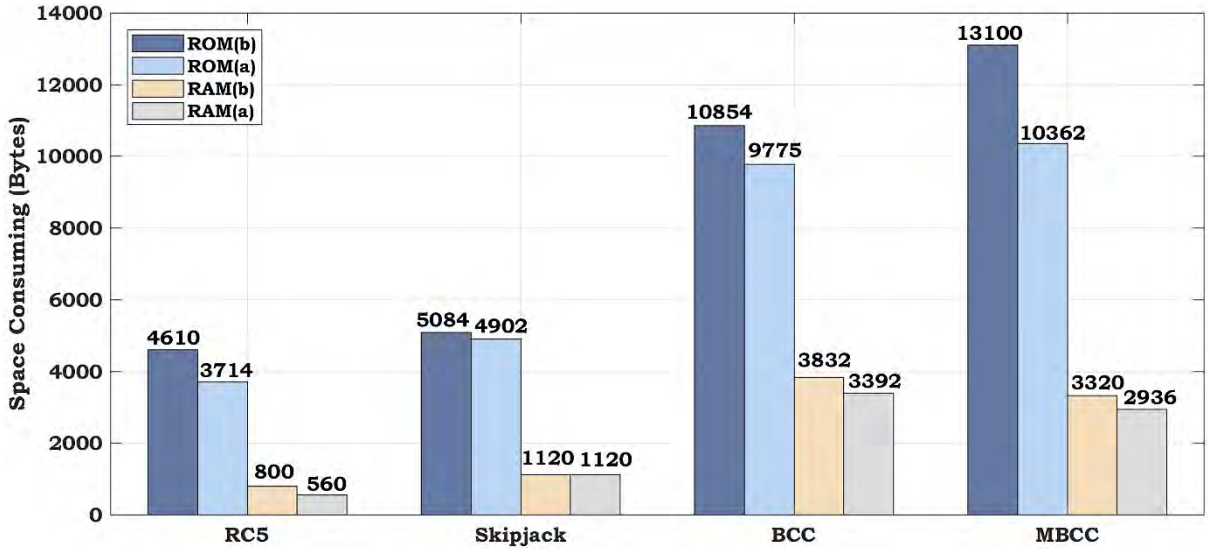

FIGURE 7. Memory usage for scenario (a) encryption and decryption operation and scenario (b) encryption, decryption and round-key generation operations by each algorithm, for RAM and ROM usage.

the remarked algorithms is evaluated. The ROM usage is the space required by the codes that are placed permanently on the sensor nodes. On the other hand, the space required for any runtime code is considered as RAM usage which is used to store the stack and variables of intermediate calculation results. Since RAM is the memory used at runtime, impacts sensor performance directly, hence it is of higher importance than ROM [44].

The value inside the substitution boxes (S-Box) and permutation tables must be saved permanently on the sensor node. Therefore, the ROM usage in MBCC and BCC is high. Meanwhile, RC5 and Skipjack exhibit lower ROM usage due to less complex methods applied.

In (Fig. 7-a), the records of memory usage of algorithms for both encryption and decryption are illustrated. The MBCC algorithm requires the most ROM usage comparing to other algorithms. However, the RAM usage in MBCC is smaller than that of BCC. The RAM is more expensive than the ROM memory and requires complex technologies to implement [44]. In addition, the memory usage for generating the round-key also reported in (Fig. 7-b). The results of this section guide the key-generation study for MBCC in the periodical key change scenario. By adding round-key generation methods, ROM and RAM usages increase by 26.4% and 13%, respectively in MBCC.

As discussed in [45] ultra-lightweight implementations require up to 4 KB ROM and 256 bytes RAM, low-cost implementations require up to 4 KB ROM and 8KB RAM, and lightweight implementations require up to 32 KB ROM and 8 KB RAM. Based on this classification, the BCC and MBCC algorithms can be considered as light weight algorithms.

B. PERIODICAL KEY CHANGE ANALYSIS IN MBCC

The periodical key change increases the data security in the encryption process while reducing the time available for the statistical attacks on the received packets. The key change period initially requires regeneration of the master key, and the regeneration of the round keys afterward. In this process, the generation of each key can be performed in either the base station or the sensor node. Generating the master key in the base station is preferred since there is no power constraint in the base station.

Furthermore, when generated by the base station, the master key generation method is not required in the sensor node. Therefore, the complex mechanisms and the chaos maps can be used to generate this key. Employing such a complex mechanism is inconvenient in constraint sensor nodes. On the other hand, if the master key generation algorithms are simple and reside on the sensor node, in case of physical capture of the sensors, the attackers could regenerate the previous keys and decrypt the previously sniffed messages.

According to (Fig. 7-b), in case of generating the round keys in the sensor node, adding the key generation methods to the node increases the memory usage. In this case, the consumed energy is equal to the total energy consumption for the decryption of the received 128-bit master key from the base station and the round key generation process. On the other hand, if the round keys are generated in the base station, the energy consumption in the sensor node will be equal to the energy for decrypting twelve 128-bit round keys. According to the results presented in Table 8, round key generation in the sensor nodes instead of the base station reduces the energy consumption in the sensor node by 1781 mJ. Therefore, considering the higher priority of the energy resource constraints relative to the memory resource limitation, the round key generation in the sensor node is more efficient than in the base station. In this study, the cost of receiving and storing new key in the sensor nodes is considered negligible.

TABLE 8. Analysis of memory and energy consumption increase in order to modify the round keys in the sensor node.

Scenario	*Energy Consumption*	*RAM Usage Increase*	*ROM Usage Increase*
Round Key Generation in Base Station	1962.2844 mJ	0%	0%
Round Key Generation in Sensor Node	181.2927 mJ	%13	%26.4

TABLE 9. Average count of '1's and '0's in 100 encrypted samples with the size of 10000 byte.

Algorithm	*'0's count (k1)*	*'1's count (k2)*	*|k1-k2|/n*
Plain	37545	42455	0.061375
RC5	40118	39882	0.00295
Skipjack	39828	40172	0.0043
BCC	40109	39891	0.00272
MBCC	39888	40112	0.00280

C. SECURITY METRICS

As illustrated in Fig. 5 and Fig. 6, by increasing the packet size, the energy consumption and the latency of the encryption operation is increased. Since the data packets transmitted by the sensor nodes are of small size, for the performance evaluation metrics, it is assumed that the maximum size of a packet is 512 bytes.

However, to study the security metrics with acceptable precision, data of greater sizes are required. Therefore, there are 100 random text samples, sizing 10000 bytes each, generated through Lorem-Ipsum library [46]. The encrypted outputs from each algorithm are employed in the security metric analysis.

1) KEY SIZE (LENGTH)

The size of the initial key is very crucial in the achieved security level of the encryption algorithms, for resisting against the brute force attacks. The longer the key size is, the securer the encryption algorithms will be.

On the other hand, the longer the key size is, the higher the key processing time and memory usage will be required. Hence, the appropriate key size must be decided based on the required security and the available resources.

According to Table 7, amongst the presented algorithms, BCC and RC5 have a 64-bit key size. The key size in Skipjack and MBCC is 80 and 128 bits, respectively. Therefore, compared to all other algorithms discussed in this paper, the possibility of the brute force attack to MBCC is less likely.

2) BALANCE ANALYSIS

In balance analysis, the ciphertext is converted to binary data. In this analysis, the total number of bits in the encrypted output, with 1s and 0s value are calculated individually. The closer the two total values are, the better the encryption output will be, which results in a random encryption output. Therefore, the attacker can extract and gain little statistical information by analyzing the encrypted outputs. The achieved outputs, considering the employed 10000-byte sample text, are shown in Table 9. Accordingly, BCC (with a value of 0.00272) has the minimum random distribution due to its permutation and substitution operations in each round of data block encryption. Furthermore, Skipjack (with the value of 0.0043) has the maximum random distribution since it only uses operations such as shift and XOR.

3) CHARACTER FREQUENCY ANALYSIS

In encrypted ciphertext, the characters are distributed more evenly with lower fluctuation, which makes them less predictable compared with unencrypted plaintext.

The smaller the character frequency distribution domain is, the chances of obtaining information through the ciphertext analysis are lowered. The character distribution in the outputs of the encryption algorithms is shown in a range of 0 to 256 ASCII characters in Fig. 8. The BCC algorithm has the smallest frequency distribution domain due to the use of permutation and substitution operations in each round of data block encryption. The MBCC algorithms' frequency distribution domain has increased compared with BCC.

This small growth can be due to the reduced permutation operations in the block operation. In RC5, the addition, subtraction, shift, and XOR operations are employed. There are no permutation tables in this algorithm. The frequency distribution domain of Skipjack is greater than that of RC5. The round count of Skipjack equals 32, but it only uses operations such as shift and XOR. Hence, the frequency distribution domain of this algorithm covers greater values comparing with other algorithms. It is inferred that the algorithms with substitution-permutation structures have smaller frequency distribution domains. Thus, the small values of frequency distribution domain related to MBCC and BCC are indicators of their relatively higher security in comparison to the other two. The statistical distribution range of the characters in the encrypted outputs is plotted in Fig. 9.

4) INFORMATION ENTROPY ANALYSIS

Information entropy exhibits the discrete probability of random events and can be used to measure the system confusion. The higher the system confusion is, the greater the entropy

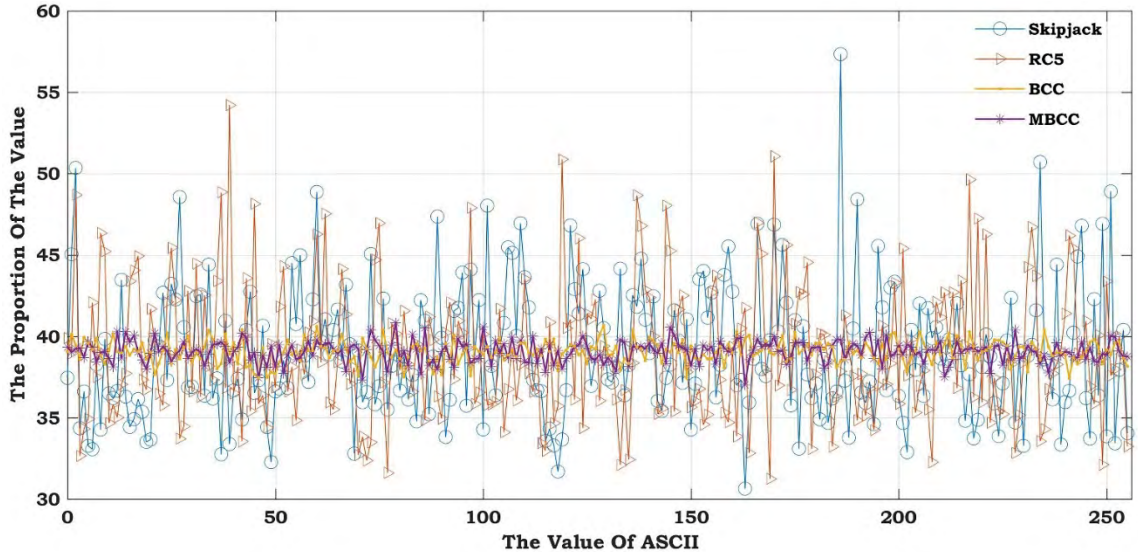

FIGURE 8. The statistical distribution of characters in the ciphertext.

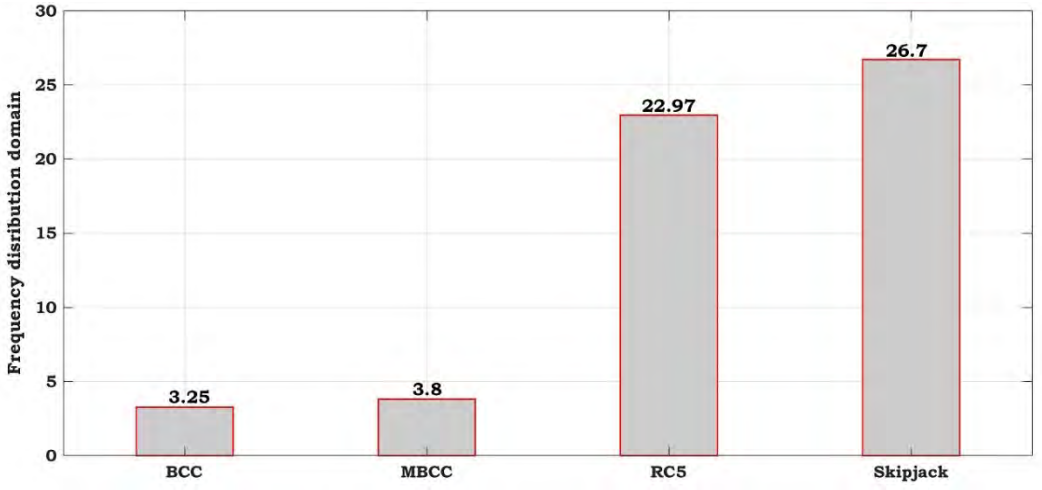

FIGURE 9. Comparison of the ciphertext characters' distribution domain for the implemented algorithms.

value becomes. In the entropy analysis, the random events can be presented as sequence values in bytes. The ideal value for entropy equals 8, for when the values of the encrypted string are fully distributed [7]. The system entropy is calculated according to (7), where P(si) is the probability of each ASCII character appearing in the cipher output:

$$H(S) = \sum_{s} P(S_i) \log_2 \frac{1}{P(s_i)} \tag{7}$$

In the results achieved, as depicted in Fig. 10, similarly with the character frequency distribution domain metric, BCC and MBCC are the two algorithms with higher entropy.

5) DIFFUSION AND CONFUSION ANALYSIS

Diffusion and confusion are the two basic principals in the cipher design [47]. Confusion and diffusion respectively complicate the statistical relation between the key and the ciphertext and causes each input bit to affect multiple ciphertext bits [7]. In addition, the properties of diffusion and confusion of MBCC and BCC as relative to text sensitivity will be investigated with respect to completeness, avalanche effect, and strict avalanche effect metrics.

- Completeness: this metric is used to measure the amount of connectivity of each output bit in the encryption function, to all the input bits. The completeness value must equal to 1 for all the encryption algorithms and is calculated according to (8):

$$d_c = 1 - \frac{1}{nm} \not\equiv \left\{(i,j)\,|a_{ij} = 0\right\}, \quad (i = 1,\ldots,n; i = 1,\ldots,m) \tag{8}$$

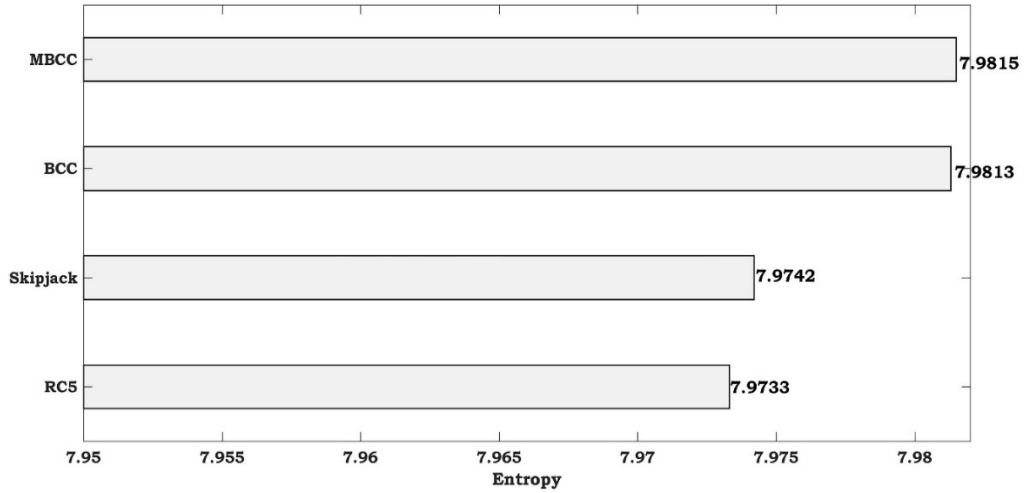

FIGURE 10. The selected algorithms' entropy comparison for 100 ciphertexts with the size of 10000 bytes.

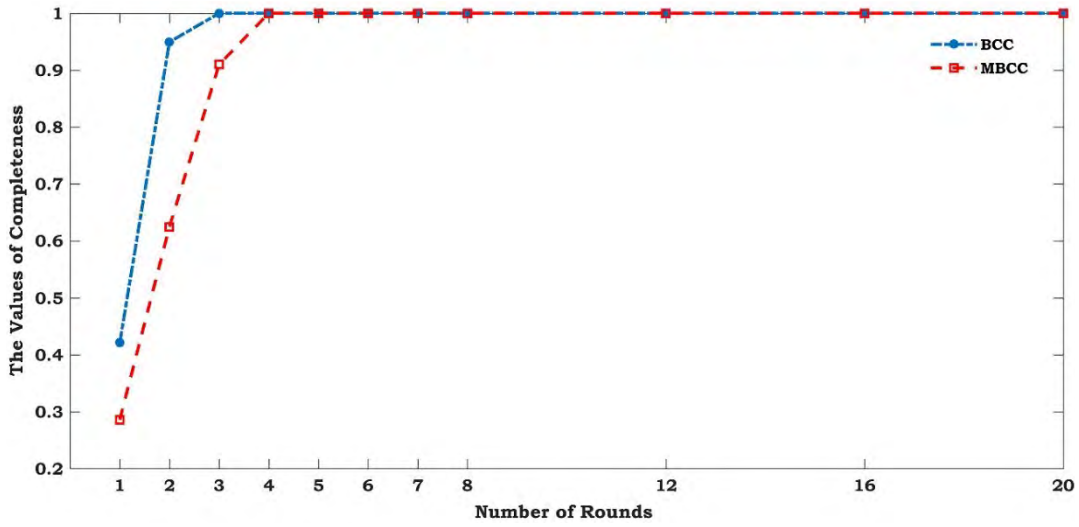

FIGURE 11. Completeness in BCC and MBCC.

where n is the number of input bits to the encryption method, and m, is the number of output bits [11]. According to Fig. 11, BCC and MBCC are converged to 1 after third and fourth rounds, respectively.

- Avalanche effect: this metric is employed to address the necessity of each input bit affecting half of the encryption output bits. Hence, the avalanche effect value for a secure algorithm is presumed to impact on the half of the block bits. This impact means in case one bit in the input data changes, about half of the output bits must vary.
  The value for avalanche effect is acquired according to (9):

$$d_{a1} = \frac{1}{\#X * n} \sum_{i=1}^{n} \left(\sum_{x \in X} W_H \left(F(x) \oplus F(x^{(i)}) \right) \right) \tag{9}$$

where #X, n, and WH, represent cipher data count, bit count and hamming distance of each data, respectively. Furthermore, F(x) is the cipher data and $F(x^{(i)})$ is equivalent to the cipher data with one difference in the ith bit [11].
Observed by the avalanche effect analysis shown in Fig. 12, MBCC grows less than BCC. The reason for the decline in metric growth in MBCC is the reduction in the number of permutations, which reduces the energy consumption. The avalanche metric value for BCC rises from 21.2 in the first round and to 50.9 in the second round. The anticipated value of 64 bits (i.e., half the block) is achieved in the fourth round. While in MBCC, the metric value slowly grows from 12.3 in the first round, to 27.0 in the second, MBCC comes closest to the anticipated value (i.e., 64) after the sixth rounds. For BCC this value equals 4.

- Strict avalanche effect: in case of a variation in any of the input bits, each output bit must vary by with the probability of 1/2. Equation (10) illustrates the value for

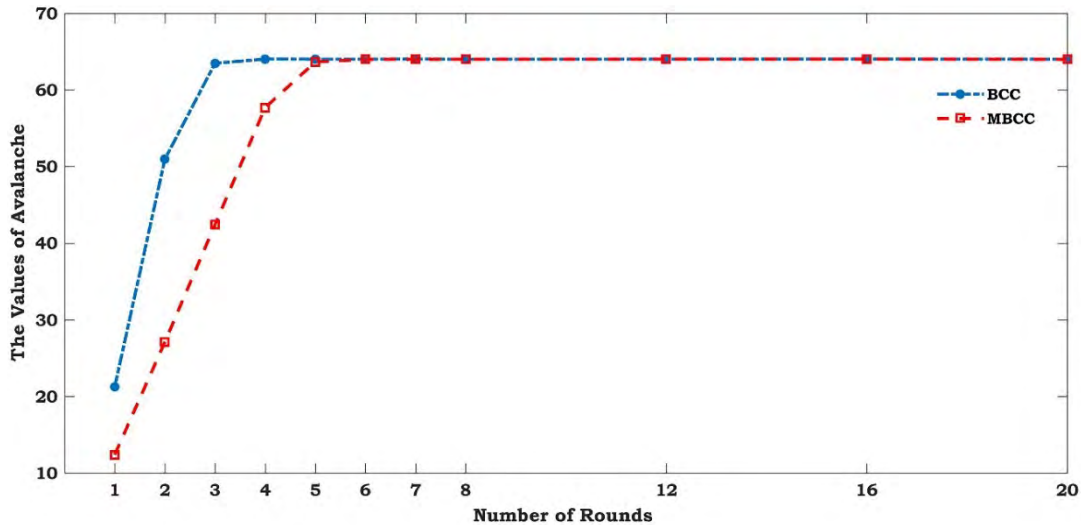

FIGURE 12. Avalanche effect in BCC and MBCC.

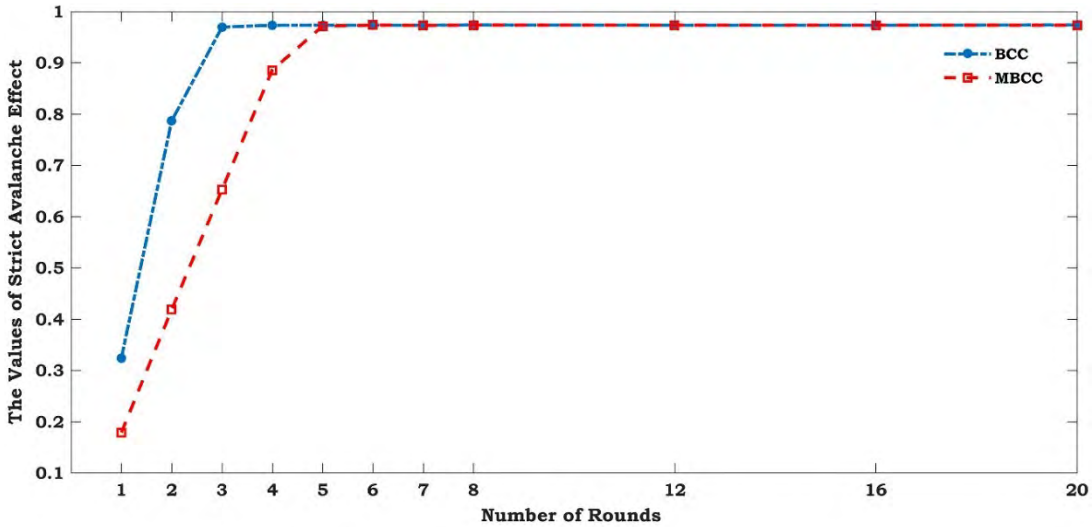

FIGURE 13. Strict avalanche effect in BCC and MBCC.

avalanche effect:

$$d_{a1} = \frac{1}{\#X * n} \sum_{i=1}^{n} \left(\sum_{x \in X} W_H \left(F(x) \oplus F\left(x^{(i)}\right) \right) \right) \tag{10}$$

where $\#X$, n, and m respectively represent the cipher data count, the individual data bit count, and the individual cipher data bit count. The anticipated value for this equation is 1 [11]. As shown in the strict avalanche effect analysis in Fig. 13, the values found by MBCC have reached the acceptable values after the fifth round, while in BCC; similar values can be procured through three iterations.

D. CRYPTANALYSIS

Cryptanalysis is based on two different categories of statistical and differential attacks. In statistical cryptanalysis, the linear relationship between the plaintext, the ciphertext and the master key (or one of the round keys) is investigated. On the other hand, differential cryptanalysis analyses the effect of differences in plaintext pairs on the differences of the ciphertext pairs [48]. As explained in Section III.B, the number of permutation operations in MBCC is decreased compared to BCC. Furthermore, the utilized S-Box in BCC and MBCC are similarly designed based on the Logistic [49] and Baker [50] chaos maps. The utilization of chaos map results in the incomprehensibility of the connections between the input and output bits, and the stability of the S-box against statistical and differential attacks.

Furthermore, the value of Avalanche effect shows the sensitivity of the algorithm to the plaintext and the effect of the plaintext changes on the ciphertext, hence BCC shows higher sensitivity than MBCC. Therefore, the more number of the plaintext-ciphertext pair is required in BCC compared to MBCC to break encryption algorithm. It can be concluded that the possibility of vulnerabilities of the MBCC to differential attacks is increased relative to BCC. However, MBCC due to the use of nonlinear elements is more resistant to differential and statistical attacks compared to algorithms such as RC5 and Skipjack, which utilize simple operations such as XOR. In the XOR operation, any change of input reflects directly on the output.

E. TRADE-OFF POINTS

The fair trade-off between security and performance will help put forward effective solutions based on applications. As discussed in the security metric (subsection C), while the proposed algorithm outperforms BCC with respect to energy efficiency, there exist other encryption algorithms such as skipjack and RC5 that are designed for energy-limited systems but with lower security. MBCC aims to provide better security compared to Skipjack and RC5 in the expense of reducing the energy overhead compared to BCC. For encrypting 32-bit data, Skipjack and RC5 consume 81.67% and 55.95% less energy and 61.85% and 80.93% of RAM less than MBCC.

On the other hand, MBCC has the higher level of security for statistical attacks based on all tested security metrics including entropy, balance analysis and distribution of characters compared with Skipjack and RC5. Furthermore, due to the use of the nonlinear structure of the substitution box in this algorithm, it is more resistant to differential attacks.

The trade-off point with BCC is reduced security level due to the reduction of the number of permutation operation that reduces entropy and statistical distribution of characters, and in turn prune to statistical attacks for similar rounds (e.g., 12 rounds). MBCC reaches the security level with higher rounds compared to the BCC algorithm in diffusion and confusion analysis. According to this analysis, the MBCC requires the higher number of rounds to achieve the ideal value for avalanche effect. It can be concluded that the possibility of vulnerabilities of the MBCC to differential attacks is increased relative to BCC, but with more rounds, MBCC reaches the ideal value.

The MBCC requires more number of rounds to achieve the optimal resistance and the acceptable security levels following the proposed changes of MBCC in Section III.B. However, the time needed to run additional rounds in comparison with BCC is negligible. As the results show for encrypting 32-bit data MBCC consumes 84.83% less energy and 13.44% of RAM compared with BCC.

The radar chart depicted in Fig. 14 indicates a summary of implemented algorithms in this work based on security, energy, and time and memory consumption. An ideal encryption algorithm should have a high level of security

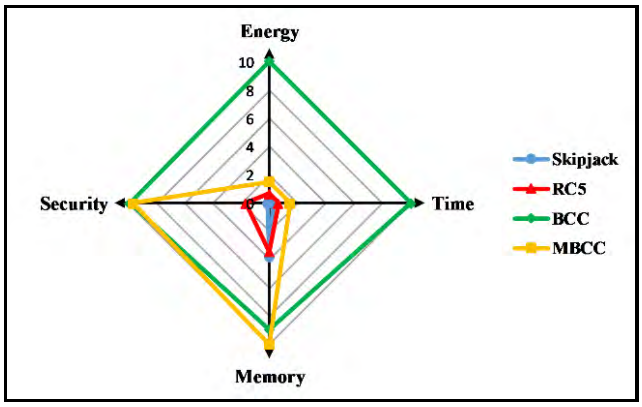

FIGURE 14. Comparison of cipher techniques with respect to energy, time, memory and security.

(10: highest) and low energy (0: lowest), time (0: lowest) and memory overhead (0: lowest) to be suitable for constraint wireless sensor applications. As it is obvious from Fig. 14, the MBCC algorithm has lower time and energy overhead compared to the BCC algorithm while preserving almost the same level of security at the expense of higher memory requirements. In addition, with respect to low energy algorithms such as RC5 and Skipjack, the gain in security is much more valuable than the imposed overhead of the increased energy, time and memory consumption. As a result, the MBCC can be considered as an overall suitable encryption algorithm for the battery-operated sensors while maintaining a high-security level.

V. CONCLUSION AND FUTURE WORK

Encryption techniques are the basic approaches to protecting the privacy of data in wireless sensor networks. Since the sensor nodes have limited resources, the algorithms that consume less energy and memory are required. In this paper, we propose MBCC, a modified version of BCC, which is a resource efficient algorithm, based on the chaos theory to reduce operation time as compared to BCC by reducing the number of permutation calls. Our testbed evaluation results show that the time for encrypting a 32-byte data is decreased from 220.74 milliseconds in BCC to 34.02 milliseconds in MBCC. In addition, MBCC outperforms BCC by 84.83% with respect to energy efficiency. On the other hand, due to modifications made to MBCC, higher ROM usage in is observed compared to BCC. However, MBCC performs better for RAM utilization which is more crucial resource than ROM. The total occupied memory for the encryption operation in MBCC and BCC are 13330 bytes and 13191 bytes, respectively. Thus, on account of sensor resource consumption and performance, MBCC is more suitable for implementation on sensor network nodes than BCC. Due to the use of more complex operators, the amount of time, energy, and memory consumed in MBCC are more significant than RC5 and Skipjack. Since the permutations operations do not have any impact on the nonlinear attacks (e.g., differential attacks), despite the reduction of permutation operation in MBCC, the security measures are within an acceptable range in comparison with

Skipjack and RC5. Furthermore, MBCC and BCC, which use chaos theory in their encryption core, are remarkably superior to RC5 and Skipjack with respect to entropy and character frequency metrics. The diffusion and confusion analysis for text sensitivity in BCC reaches an acceptable value in the fourth round and MBCC reaches the acceptable value on at least round six, due to the reduction of the number of permutations in MBCC. In addition, MBCC is more resilient against the brute force attack than BCC due to a greater key space. Further experiments on the periodical key change in MBCC results in the periodical key generation. It injects additional overhead on the sensor nodes, in terms of memory usage and energy consumption. Implementing the node with the option of round key generation results in the increase of the ROM and RAM usages by 26.4% and 13%, respectively. Additionally, our measurements show that for each round key generation process, approximately 181.2927 mJ energy is consumed by the sensor node. This is due to the master key decryption and round keys generation. Considering a set up that the round keys are generated in the base station, the energy consumed for decrypting keys in the sensor node is 1962.2844 mJ that compared to the energy consumed for the key generation at the sensor node is increased by 9.82 times in our testbed. In conclusion, MBCC exhibits better performance as compared to BCC and superior security metrics in comparison with RC5 and Skipjack.

Since the securer algorithms come with the higher degree of complexities and higher memory and energy consumption; therefore, MBCC can be employed in environments where security is of higher priority than performance.

As for our security attack analysis, avalanche effect which shows the sensitivity of the algorithm to the plaintext results in higher sensitivity in BCC compared to MBCC. Furthermore, vulnerabilities of the MBCC to differential attacks is increased relative to BCC. Our observations show MBCC is more resistant to differential and statistical attacks compared to algorithms such as RC5 and Skipjack due to the use of nonlinear elements.

The proposed algorithm can be further improved by carrying a performance study on the security metrics in various rounds in the MBCC. It is required to probe the possibility of reducing algorithm rounds while preserving the minimum-security criteria. Further analysis of the selection of different effective criteria is required to determine the period of key change for the sensor nodes.

Further research is required for the security assessment of cryptographic systems. Cryptanalytic attacks such as side-channel attacks (e.g. power-analysis attacks, timing attack, and cache-side channel attacks), chosen plaintext attack (e.g. differential cryptanalysis) and dictionary attacks are complementary to the security attack analysis provided in this work.

REFERENCES

[1] G. A. Fink, D. V. Zarzhitsky, T. E. Carroll, and E. D. Farquhar, "Security and privacy grand challenges for the Internet of Things," in *Proc. Int. Conf. Collaboration Technol. Syst. (CTS)*, Atlanta, GA, USA, 2015, pp. 27–34.

[2] A. Jafari, M. Shirali, and M. Ghassemian, "A testbed evaluation of MAC layer protocols for smart home remote monitoring of the elderly mobility pattern," in *Proc. CMBEBIH*, Singapore, 2017, pp. 568–575.

[3] H. Rezaie and M. Ghassemian, "An adaptive algorithm to improve energy efficiency in wearable activity recognition systems," *IEEE Sensors J.*, vol. 17, no. 16, pp. 5315–5323, Aug. 2017.

[4] A. Bhave and S. R. Jajoo, "Secure communication in wireless sensor networks using hybrid encryption scheme and cooperative diversity technique," in *Proc. 9th Int. Conf. Intell. Syst. Control (ISCO)*, Coimbatore, India, 2015, pp. 1–6.

[5] S. Pérez *et al.*, "A lightweight and flexible encryption scheme to protect sensitive data in smart building scenarios," *IEEE Access*, vol. 6, pp. 11738–11750, 2018.

[6] J. Lin, W. Yu, N. Zhang, X. Yang, H. Zhang, and W. Zhao, "A survey on Internet of things: Architecture, enabling technologies, security and privacy, and applications," *IEEE Internet Things J.*, vol. 4, no. 5, pp. 1125–1142, Oct. 2017.

[7] X.-J. Tong, Z. Wang, Y. Liu, M. Zhang, and L. Xu, "A novel compound chaotic block cipher for wireless sensor networks," *Commun. Nonlinear Sci. Numer. Simul.*, vol. 22, nos. 1–3, pp. 120–133, 2015.

[8] W. K. Koo, H. Lee, Y. H. Kim, and D. H. Lee, "Implementation and analysis of new lightweight cryptographic algorithm suitable for wireless sensor networks," in *Proc. Int. Conf. Inf. Secur. Assurance. (ISA)*, Busan, South Korea, Apr. 2008, pp. 73–76.

[9] M. Dener, "Security analysis in wireless sensor networks," *Int. J. Distrib. Sensor Netw.*, vol. 10, no. 10, p. 303501, 2014.

[10] J. Daemen and V. Rijmen, *The Design of Rijndael: AES—The Advanced Encryption Standard*. Berlin, Germany: Springer Science & Business Media, 2013.

[11] Y. Liu, S. Tian, W. Hu, and C. Xing, "Design and statistical analysis of a new chaotic block cipher for wireless sensor networks," *Commun. Nonlinear Sci. Numer. Simul.*, vol. 17, no. 8, pp. 3267–3278, 2012.

[12] (2009). *Sun SPOT World*. Accessed: Apr. 14, 2018. [Online]. Available: http://www.sunspotdev.org/docs/Yellow/SunSPOT-TheoryOfOperation.pdf

[13] M. S. Baptista, "Cryptography with chaos," *Phys. Lett. A*, vol. 240, nos. 1–2, pp. 50–54, 1998.

[14] T. Xiao-Jun, W. Zhu, and Z. Ke, "A novel block encryption scheme based on chaos and an S-box for wireless sensor networks," *Chin. Phys. B*, vol. 21, no. 2, p. 020506, 2012.

[15] G. Zaibi, F. Peyrard, A. Kachouri, D. Fournier-Prunaret, and M. Samet, "Efficient and secure chaotic S-Box for wireless sensor network," *Secur. Commun. Netw.*, vol. 7, no. 2, pp. 279–292, 2014.

[16] H. Y. Wang, "Wireless sensor networks based on a new block encryption algorithm," *Appl. Mech. Mater.*, vol. 551, pp. 454–459, May 2014.

[17] A. Pande and J. Zambreno, "A chaotic encryption scheme for real-time embedded systems: Design and implementation," *Telecommun. Syst.*, vol. 52, no. 2, pp. 551–561, 2013.

[18] B. J. Mohd, T. Hayajneh, and A. V. Vasilakos, "A survey on lightweight block ciphers for low-resource devices: Comparative study and open issues," *J. Netw. Comput. Appl.*, vol. 58, pp. 73–93, Dec. 2015.

[19] P. Chao, Y. Xiao, W. Liang, and X. Han, "Trade-off of security and performance of lightweight block ciphers in industrial wireless sensor networks," *EURASIP J. Wireless Commun. Netw.*, vol. 2018, no. 1, p. 117, 2018.

[20] C. Karlof, N. Sastry, and D. Wagner, "TinySec: A link layer security architecture for wireless sensor networks," in *Proc. ACM 2nd Int. Conf. Embedded Netw. Sensor Syst.*, 2004, pp. 162–175.

[21] Y. Wang, K.-W. Wong, X. Liao, and T. Xiang, "A block cipher with dynamic S-boxes based on tent map," *Commun. Nonlinear Sci. Numer. Simul.*, vol. 14, no. 7, pp. 3089–3099, 2009.

[22] A. Perrig, R. Szewczyk, J. D. Tygar, V. Wen, and D. E. Culler, "SPINS: Security protocols for sensor networks," *Wireless Netw.*, vol. 8, no. 5, pp. 521–534, Sep. 2002.

[23] D. Hong, J.-K. Lee, D.-C. Kim, D. Kwon, K. H. Ryu, and D.-G. Lee, "LEA: A 128-bit block cipher for fast encryption on common processors," in *Proc. Int. Workshop Inf. Secur. Appl.* Cham, Switzerland: Springer, 2013, pp. 3–27.

[24] B. Eli, A. Biryukov, and A. Shamir, "Cryptanalysis of Skipjack reduced to 31 rounds using impossible differentials," in *Proc. Int. Conf. Theory Appl. Cryptograph. Techn.* Berlin, Germany: Springer, 1999, pp. 12–23.

[25] L. Wei *et al.*, "Security analysis of the lightweight cryptosystem TWINE in the Internet of Things," *KSII Trans. Internet Inf. Syst.*, vol. 9, no. 2, pp. 793–810, 2015.

[26] Ö. Boztaş, F. Karakoç, and M. Çoban, "Multidimensional meet-in-the-middle attacks on reduced-round TWINE-128," in *Proc. Int. Workshop Lightweight Cryptogr. Secur. Privacy*. Berlin, Germany: Springer, 2013, pp. 55–67.

[27] J. Choi and Y. Kim, "An improved LEA block encryption algorithm to prevent side-channel attack in the IoT system," in *Proc. IEEE Asia–Pacific Signal Inf. Process. Assoc. Annu. Summit Conf. (APSIPA)*, Dec. 2016, pp. 1–4.

[28] B. Koo, D. Hong, and D. Kwon, "Related-key attack on the full HIGHT," in *Proc. Int. Conf. Inf. Secur. Cryptol.* Berlin, Germany: Springer, 2010, pp. 49–67.

[29] *Intel Architecture Software Developer'S Manual*, Santa Clara, CA, USA, Intel Corp., 1997.

[30] M. Tunstall, D. Mukhopadhyay, and S. Ali, "Differential fault analysis of the advanced encryption standard using a single fault," in *Proc. IFIP Int. Workshop Inf. Secur. Theory Practices*. Berlin, Germany: Springer, 2011, pp. 224–233.

[31] P. Derbez, P.-A. Fouque, and D. Leresteux, "Meet-in-the-middle and impossible differential fault analysis on AES," in *Proc. Int. Workshop Cryptograph. Hardw. Embedded Syst.* Berlin, Germany: Springer, 2011, pp. 274–291.

[32] H. Handschuh and H. M. Heys, "A timing attack on RC5," in *Proc. Int. Workshop Sel. Areas Cryptogr.* Berlin, Germany: Springer, 1998, pp. 306–318.

[33] A. Biryukov and E. Kushilevitz, "Improved cryptanalysis of RC5," in *Proc. Int. Conf. Theory Appl. Cryptograph. Techn.* Berlin, Germany: Springer, 1998, pp. 85–99.

[34] J. Nakahara, Jr., P. Sepehrdad, B. Zhang, and M. Wang, "Linear (hull) and algebraic cryptanalysis of the block cipher PRESENT," in *Proc. Int. Conf. Cryptol. Netw. Secur.* Berlin, Germany: Springer, 2009, pp. 58–75.

[35] J. H. Kong, L.-M. Ang, and K. P. Seng, "A comprehensive survey of modern symmetric cryptographic solutions for resource constrained environments," *J. Netw. Comput. Appl.*, vol. 49, pp. 15–50, Mar. 2015.

[36] M. Luk, G. Mezzour, A. Perrig, and V. Gligor, "MiniSec: A secure sensor network communication architecture," in *Proc. 6th Int. Conf. Inf. Process. Sensor Netw.*, Cambridge, MA, USA, 2007, pp. 479–488.

[37] Y. W. Law, J. Doumen, and P. Hartel, "Survey and benchmark of block ciphers for wireless sensor networks," *ACM Trans. Sensor Netw.*, vol. 2, no. 1, pp. 65–93, 2006.

[38] I. Mansour, G. Chalhoub, and B. Bakhache, "Evaluation of a fast symmetric cryptographic algorithm based on the chaos theory for wireless sensor networks," in *Proc. 11th Int. Conf. Trust, Secur. Privacy Comput. Commun. (TrustCom)*, Liverpool, U.K., 2012, pp. 913–919.

[39] K. Biswas, V. Muthukkumarasamy, and K. Singh, "An encryption scheme using chaotic map and genetic operations for wireless sensor networks," *IEEE Sensors J.*, vol. 15, no. 5, pp. 2801–2809, May 2015.

[40] J. Qi, X. Hu, Y. Ma, and Y. Sun, "A hybrid security and compressive sensing-based sensor data gathering scheme," *IEEE Access*, vol. 3, pp. 718–724, 2015.

[41] K. K. Win, X. Wu, S. Dasgupta, W. J. Wen, R. Kumar, and S. K. Panda, "Efficient solar energy harvester for wireless sensor nodes," in *Proc. Int. Conf. Commun. Syst. (ICCS)*, Singapore, 2010, pp. 289–294.

[42] A. Caicedo, "The sun small programmable object technology (sun spot): Java technology-based wireless sensor networks," in *Proc. Sun Tech Days Conf.*, India, Feb. 2007, p. 6.

[43] J. Zandi, A. N. Afooshteh, and M. Ghassemian, "Implementation and analysis of a novel low power and portable energy measurement tool for wireless sensor nodes," in *Proc. 26th Iranian Conf. Elect. Eng. (ICEE)*, 2018, pp. 1517–1522.

[44] Tech Differences. (2017). *Difference Between RAM and ROM Memory (With Comparison Chart)—Tech Differences*. Accessed: Apr. 28, 2018. [Online]. Available: https://techdifferences.com/difference-between-ram-and-rom-memory.html

[45] 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*. Berlin, Germany: Springer, 2014, pp. 333–349.

[46] (2018). *Anon*. Accessed: Jan. 14, 2018. [Online]. Available: https://www.lipsum.com/

[47] X.-Y. Wang and Q. Yu, "A block encryption algorithm based on dynamic sequences of multiple chaotic systems," *Commun. Nonlinear Sci. Numer. Simul.*, vol. 14, no. 2, pp. 574–581, 2009.

[48] E. Biham and A. Shamir, "Differential cryptanalysis of DES-like cryptosystems," *J. Cryptol.*, vol. 4, no. 1, pp. 3–72, 1991.

[49] R. M. May, "Simple mathematical models with very complicated dynamics," *Nature*, vol. 261, no. 5560, pp. 459–467, 1976.

[50] M. D. Esposti, S. Nonnenmacher, and B. Winn, "Quantum variance and ergodicity for the baker's map," *Commun. Math. Phys.*, vol. 263, no. 2, pp. 325–352, 2006.

MASOUMEH SHARAFI received the B.Sc. and M.Sc. degrees from the Department of Computer Engineering and Information Technology, University of Qom, in 2013 and 2018, respectively. She is a member of the IoT and Smart Environment Research Center. Her research interests include the IoT, security, privacy, and lightweight encryption.

FARANAK FOTOUHI-GHAZVINI received the M.Eng. degree (Hons.) in telecommunication engineering from King's College London, U.K., in 2001, and the Ph.D. degree from the Department of Informatics, Bradford University, U.K., in 2011. She is currently an Assistant Professor with the Department of Computer Engineering and Information Technology and the Director of IoT and Smart Environments Research Group, University of Qom. Her research interests include the IoT, smart home, and assistive living.

MOHSEN SHIRALI received the B.Sc. degree (Hons.) in software engineering from Islamic Azad University, Tehran North Branch, in 2010, and the M.Sc. degree (Hons.) in computer architecture from Shahid Beheshti University, Tehran, Iran, in 2013, where he is currently pursuing the Ph.D. degree with the eSense Research Lab, Department of Computer Science and Engineering. His Ph.D. research focus is mainly on data security and privacy solutions in wireless sensor networks, smart homes, and wearable systems for health and wellbeing applications. He researched on security of wireless sensor networks and standards and applications of industrial networks for his M.Sc. and B.Eng. projects, respectively.

MONA GHASSEMIAN received the Ph.D. degree in mobile and personal communications research from King's College London, in 2006, with NTT DoCoMo scholarship. After completion of her Ph.D. studies, she continued at King's College London as a Research Associate in eSense (6^{th} Framework Programme of the European Commission) project. She was with the University of Greenwich as a Senior Lecturer and a PG Programme Leader in computer networking, from 2007 to 2012, a Faculty Member with SBU and eSense Research Group Leader, from 2012 to 2018, and a Visiting Research Associate at King's College London, from 2008 to 2017. Her research interests include energy efficient communication protocols for wireless sensor networks with a focus on smart health and well-being applications. She served as the IEEE U.K. and Ireland Secretary, from 2012 to 2015, as the IEEE Region 8 Student Activities Committee Chair, from 2014 to 2016, and has been serving as the IEEE U.K. and Ireland Section Vice-Chair, since 2018.