Towards an ultra lightweight block ciphers for Internet of Things

Layth Sliman, Tasnime Omrani, Zahir Tari, Abed Ellatif Samhat, Rhouma Rhouma

Towards an ultra lightweight block ciphers for Internet of Things

Layth Sliman[a,b,*,1], Tasnime Omrani[c,1], Zahir Tari[d], Abed Ellatif Samhat[e], Rhouma Rhouma[c]

[a] EFREI Paris, Villejuif, France
[b] Université Paris-Saclay, Univ Evry, IBISC, Evry-Courcouronnes, France
[c] National Engineering School of Tunis, Tunis, Tunisia
[d] Royal Melbourne Institute of Technology, Melbourne, Australia
[e] Lebanese University, Faculty of Engineering-CRSI, Hadath, Lebanon

Conventional cryptographic methods are not appropriate for IoT environments due to the specific IoT devices constraints, such as memory usage, time and computational costs. This leads to the emergence of the lightweight cryptography field. This paper investigates the different lightweight cryptographic design methods and proposes an IoT-based cryptographic method, called Ultra-Lightweight method (ULM), to enhance the performance, memory usage and security of IoT devises. The proposed method is based on three methods (i.e, bitslice, WTS and involutive), and thus accumulating their various advantages such as, memory use, efficiency and security. To validate our proposal, a cryptosystem is designed using ULM method. The designed cryptosystem is benchmarked based on the following metrics: performance (by measuring its execution time and the number of clock cycles needed to run it), the quantity of used memory (by measuring ROM and RAM consumption), and security level (by measuring its diffusion and confusion levels along with its resistance to linear and differential attacks). The results show that the cryptosystem designed using the proposed method outperforms existing methods in terms of memory use, security and performance.

1. Introduction

The Internet of Things (IoT) involves an increasing number of physical objects enabled with data exchange capabilities. IoT can also involve different kinds of smart objects including smart vehicles, smart industrial machines, smart energy grids, smart homes and buildings, and smart portable and wearable devices, just to cite few. With billions of sensors and actuators processing data, IoT becomes an opportunity at a mass scale. IoT security, however, remains a challenging issue that hinders the spread of this technology [1–3]. Actually, in most cases, security is not considered from design time [4]. Rather, this is dealt with at the end of the chain using solutions that are not originally designed for constrained environments such IoT. This entails many performance issues due to the limited resources embedded in IoT objects [5,6]. One of the most resource greedy security function is cryptography, as it uses complex algorithms which are based on mathematical functions to hide the content from unauthorized parties. To make block cipher algorithms adaptable to IoT devices, many methods have been proposed, such as ARX (Addition/Rotation/XOR) [7], WTS (wide-trail strategy) [8], LTS (Long Trail Strategy) [7], Hybrid [9], Bitslice [10] and involutive method [11]. Indeed, ARX [7] is a method used to design lightweight crypto-algorithms with the aim of minimizing memory use. This is done using only three operations: modular addition, rotation and XOR. WTS [8] is used to design a lightweight crypto-algorithm with a high security level. This is carried out by prioritizing the use of linear functions to achieve diffusion along with Sbox substitution table to ensure confusion. LTS [7] attempts to optimize memory use while keeping a high security level, and this is done by combining WTS and ARX methods. Bitslice [10] represents the cryptosystem functions in term of single-bit logical operations, which are then carried out by running multiple instances of the function in parallel using bitwise operations. Consequently, on a processor with n-bit, a logical instruction corresponds to the parallel execution of n logical operations which accelerate the processing time of the cryptosystem. The involutive method [11] is characterized by the use of a set of involutive operations. This means that the operations used during the decryption process are the same as the ones used in the encryption process. This therefore reduces the memory usage to store operations. Hybrid method [9] combines both SPN and Feistel structures to ensure a very high level of security as well as to reduce ROM memory usage by utilizing some same functions for both encryption and decryption.

* Corresponding author at: EFREI Paris, Villejuif, France.
E-mail address: layth.sliman@efrei.fr (L. Sliman).
[1] These authors contributed equally to this work.

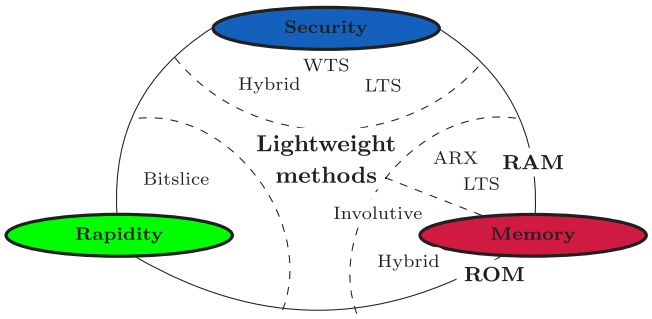

Fig. 1. Lightweight methods advantages.

These methods differ from each other by the building blocks they use, their structure as well as the way they are implemented. Each method is supposed to be adapted to one or more IoT constraints (e.g. memory use, time, speed and computational cost). However, as we will show later sections, neither of these methods consider all the aforementioned constraints together. To overcome this issue, we propose a new block cipher for IoT devices based on WTS, Bitslice and Involutive. That is to say that our proposal prioritizes the use of linear functions to achieve diffusion along with Sbox substitution table to ensure confusion. It then represents the cryptosystem functions in term of single-bit logical operations, which are then carried out by running multiple instances of the function in parallel using bitwise operations and uses a set of involutive operations. This enables designing cryptosystems that optimize memory use, reduces the execution time and provides a high level of security. The proposed block cipher is compared to existing ones (e.g., WTS, LTS, ARX, Bitslice, Involutive) and Hybrid methods and the results show a good improvement regarding performance and security while keeping a reasonable memory usage.

The rest of the paper is organized as follows. Section 2 presents the related works and briefly explains some of the existing methods to design lightweight cryptosystem. In Section 3, with the aim of covering the different IoT constraints, we describe the new proposed method and then validate it by designing and implementing a new cryptosystem based on this method. In Section 4, the proposed cryptosystem is evaluated with regard to performance (processing time, clock cycle), memory use (RAM, ROM), and security (confusion level, diffusion level and resistance to linear and differential attacks). We then compare the performance of the proposed cryptosystem with the up-to-date lightweight cryptosystems. Finally, we conclude the paper in Section 5.

2. Related works

To design a Lightweight cryptography system, different methods could be used including ARX, WTS, LTS, Hybrid, Bitslice and Involutive. The advantages of each method are depicted in Fig. 1 and explained below. Examples of a cryptosystems designed using the different methods are also presented and will be used for the comparison with our proposal.

- **ARX method**: ARX is a method used to design lightweight crypto-algorithms with the aim of minimizing memory use. This is done by using only three operations: modular addition, rotation and XOR.
  The cryptosystems that follows this method are Hight [12], Simon [13], Speck [13], Lea [14] and Chaskey cipher [15]. There are two main aspects that distinguish these algorithms: the block and key size, and the adaptability to processor's registry size (8, 16, ...). For instance, Hight decomposes the input block on 8 subblocks of 8 bits; which makes it adaptable to 8 bit processors. Lea decomposes a block on 4 subblocks of 32 bits, which makes it adaptable to 32 and 64 bits processors. Speck and Simon decompose a block into two subblocks. These algorithms have different variants with a difference in the block and key sizes of course the differences influence the time, the quantity of memory used and the security. The bigger the block size is the more memory is consumed and the higher security level is provided. Unlike Simon and Speck, Lea records the highest block and key sizes. The most popular cryptosystem based on ARX method is Speck. Speck follows Feistel structure. It has 10 variants with a difference in the block size, key size and round number (see [13] for more details). It is usually chosen to optimize hardware and software implementations. For this objective, a round of speck is composed of the following operations:
  - an Xor operation with a subkey.
  - a modular addition.
  - a left rotation with a number of bits and also and a right rotation with a number of bits.

  The ARX method is characterized by the use of only three operations (XOR, modular addition and rotation). This enhances memory RAM consumption. However, it negatively influences the security as the nonlinearity is based only on modular addition. The ARX cryptosystem guarantees security by increasing the number of rounds, which negatively impacts the overall performance of the algorithm.
- **WTS method**: WTS is a method used to design lightweight algorithms with a high security level. This is carried out by prioritizing the use of linear functions to achieve diffusion along with Sbox substitution table to ensure confusion. The cryptosystems that implemented this method are mCrypton [16], Present [17], Led [18], Zorro [19], Skinny [20], Born and Gift [21]. The difference between these algorithms resides in message and key sizes, the rounds number and block representation (bit, matrix, set of bit, set of bytes, etc.). The choice of message representation influences the rapidity of the encryption algorithm. The number and type of the S-box represent also a difference between these cryptosystems. The most popular cryptosystem based on WTS method is Present, which is a lightweight block encryption algorithm standardized in 2012 (ISO/IEC 29192-2). The block size in this algorithm is 64 bits with a key size of 80 or 128 bits. Present is based on SPN structure with 31 rounds. This structure is also used in AES standard. Present has two variants: Present80 and Present-128 with a key size equals to 80 and 128 bits respectively (see [17] for details). Thanks to its lightweight components, this cryptosystem is suitable for hardware implementation. The WTS method makes a big interest in security level by using expensive linear operations. However, this negatively impacts the overall performance.
- **LTS method**: LTS tries to optimize memory usage while keeping a high security level. This is done by combining WTS and ARX methods. Indeed, LTS uses S-box represented using Addition, Rotation and XOR operations just like ARX, combined with linear functions like WTS. The only cryptosystem that adopts this method is Sparx [7]. Sparx follows Feistel structure with three variants Sparx-64/128, Sparx-128/128 and Sparx-128/256 (see [7] for details). The only operations used for implementing an instance of Sparx are:
  - Modular addition 16 bit.
  - XOR 16 bit.
  - Left and right rotation.

  The LTS method combines WTS and ARX methods. This is done to ensure a high security and a low memory consumption ensured by ARX. LTS method increases confusion level by using a voluminous nonLinear layer. This ensures a high confusion level with minimum RAM use and reasonable performance.

- **Bitslice method**: Bitslice represents the cryptosystem functions in term of single-bit logical operations, which are then carried out by running multiple instances of the function in parallel using bitwise operations. Consequently, on a processor with n-bit, a logic instruction corresponds to the parallel execution of n logical operations which accelerate the processing time of the cryptosystem. The cryptosystems that are based on this method are Rectangle [22], Fantomas [23] and Mysterion [24]. Similar to many Lightweight block encryption algorithms, the block size of Rectangle [22] is 64 bits with 80 bits key size. It follows the SPN structure composed of 25 rounds. A round function is composed of an addition with a subkey, a substitution using 4×4 S-box followed by a bitwise shift. After the last round, an addition with subkey is applied.
  The aim of the Bitslice method is to improve performance while minimizing RAM consumption. However, the security level is highly dependent on the components chosen to implement the system.
- **Involutive method**: The involutive method is characterized by the use of a set of involutive functions. An involutive function, say f_i, is a function that takes the form $f_i(f_i(x)) = x$. This means that the functions used during the decryption process are the same as the ones used in the encryption process. This therefore reduces the memory usage to store operations. The cryptosystems that follow this method are Prince [25], Klein [26], Midori [27] and Mantis [28]. The difference between these cryptosystems is principally the number of included involutives operations. Specifically, all the operations used in Prince and Midori are involutives, while in Klein only the substitution operation is involutive. Using involutive operations in a cryptosystem reduces the memory usage, however it also decreases security. As presented in [25], Prince follows the SPN structure of eleven (11) rounds. Each round is composed of an addition with a subkey, a substitution using 4×4 S-box. This is followed by a multiplication with three different matrices, one matrix for the first five rounds, another matrix for the sixth round and a third matrix for the last five rounds. The involutive method uses the same components in both encryption and decryption. Although involutive feature reduces ROM usage, it negatively impacts the security level. Similar to Bitslice method, rapidity and RAM use are highly dependent on the choice of the implemented components.
- **Hybrid method**: This method combines both SPN and Feistel structures to ensure a high level of security as well as to reduce ROM memory usage by utilizing some functions for both encryption and decryption (similarly to involutive method). The cryptosystems that are based on this method are Sea [29], Celfia [30], Gost revisited [31], Twine [32], Lblock [33], Piccolo [34], ITUbee [35], RoadRunner [36], LiCi [37], SIT [38], ANU-II [39] and Nux [40]. The principal difference between these cryptosystems is the way they use the linear layer embedded in the SPN structure. Unlike Twine and Gost revisited, Sea, Piccolo, RoadRunner, Lici and Nux are characterized by the use of a rather costly linear layer implementation. This is done to provide a higher level of diffusion.As previously, only one cryptosystem is presented, namely the LBlock cryptosystem. LBlock is a Feistel of 32 rounds which encrypts a block of 64 bits with a key of 80 bits. A round of LBlock consists of a 8 bits rotation and a Feistel function. This function consists of a subkey addition, a substitution operation with uses of eight 4×4 S-boxes. At the end of each round, a nybble permutation is done. This operation consists of a rearrangement by 4 bits (as in [33]).
  The Hybrid method combines between SPN and Feistel structures. This positively influences the security level. However, it requires a rather intensive use of memory. The rapidity of a Hybrid method based cryptosystem is highly related to the number of its rounds.

The cryptosystems described above deal with problems related to resource-limited devices, either in terms of software implementation or hardware implementation or both. Table 1 summarizes the characteristics of each cryptosystem, where we considered the following characteristics:

- The structure adopted in the cryptosystem: SNP or Feistel.
- Keyschedule: This is the function responsible for generating subkeys. In fact, some cryptosystems use complex functions to generate subkeys whose purpose is to provide a high security level. However, this function entails the use of a relatively memory space, a lot of energy and circuit surface.
- Whether the cryptosystem addresses the hardware implementation, software implementation.
- Principal operations used including Substitution, Permutation, SWAP, Multiplication, Shift, Key Whitening, Constant Addition, Mixing with subkey, etc.
- Key size, Block size, Number of Rounds.

3. The proposed Ultra-Lightweight Method (ULM)

The theoretical studies have showed a clear correlation between security, rapidity, ROM and RAM memory consumption. As previously mentioned, each method focuses on one or two of such metrics without considering the others. Thus, none of the methods considered together the specific requirements of IoT environments, namely high performance (from a computational perspective), high security level and low memory consumption. To deal with this limitation, we propose a new IoT-Oriented cryptographic method called ULM (Ultra-Lightweight Method). This method is then validated by a new cryptosystem ULC (Ultra-Lightweight Cryptosystem) which is then implemented using Arduino Uno platform and compared with the most existents Lightweight cryptosystems described above.

3.1. Details about ULM

The aim of the proposed method is to ensure a high security level, good performance of the underlying crypto-algorithm and a low memory usage. To do so, ULM combines the Bitslice, WTS and the Involutive methods, which makes it a hybrid method that optimizes the various metrics so to make it suitable for IoT environments. The method can be summarized by the following rules:

1. To ensure a high security level, prioritize the use of linear functions to achieve diffusion along with Sbox substitution table to ensure confusion.
2. To speed-up the algorithm, represent the cryptosystem functions in term of single-bit logical operations in such a way that to enable carrying them out by running multiple instances of the function in parallel using bitwise operations. This is especially useful in case of using on an n-bit processor.
3. Reduce the memory usage to store operations by maximize the use of involutive operations.

The comparison of the proposed method with the other methods in terms of security, performance and memory usage is summarized in Table 2. And for validation purpose, a new cryptosystem, called Ultra-lightweight Cryptosystem (ULC), is designed based on the ULM method.

3.2. Ultra-lightweight Cryptosystem (ULC)

As illustrated in Fig. 2, ULC is an SPN which encrypts a block of 64 bits represented in from of bytes. This cryptosystem is composed of n rounds, where n is chosen based the conducted evaluation in Section 4.1 (we use 15 rounds). Each round is composed of a subkey addition, a bitslice substitution and an involutive bit permutation. At the end of the rounds, a mixing with a subkey is applied (see Fig. 2). These operations are explained as follows.

Table 1
The characteristics of each method/cryptosystem.

Method	ARX Speck	WTS Present	LTS Sparx	Bitslice Rectangle	Involutive Prince	Hybride LBlock
Structure	Feistel	SPN	SPN+ARX	SPN	SPN	Feistel+SPN
Key schedule	Complex	Simple	Complex	Average	Simple	Complex
Hardware	No	Yes	No	Yes	Yes	Yes
Software	Yes	No	Yes	Yes	No	Yes
Substitution	yes	Yes	Yes	Yes	yes	Yes
Permutation	No	Yes	Yes	Yes	Yes	Yes
SWAP	Yes	No	Yes	No	Yes	Yes
Multiplication	No	No	No	No	Yes	No
Shift	No	No	Yes	No	Yes	Yes
Key Whitening	No	No	No	No	Yes	No
Constant Addition	No	No	No	No	Yes	No
Mixing with subkey	Yes	Yes	Yes	Yes	Yes	Yes
Key size	32 to 128	80/128	128/256	80/128	128	80
Block size	32 to 128	64	64/128	64	64	64
Number of rounds	22 to 34	31	24–40	25	12	32

Table 2
Comparison between ULM and the existent methods.

Method	Security	Performance	RAM consumption	ROM consumption
ARX	Low	Average	Low	Neutral
WTS	High	Low	Low	High
LTS	High	High	High	High
Hybride	High	High	High	Average
Bitslice	Neutral	High if n-bits processor is used	Low	Neutral
Involutive	Low	Neutral	High	Low
ULM	High	High	Low	Low

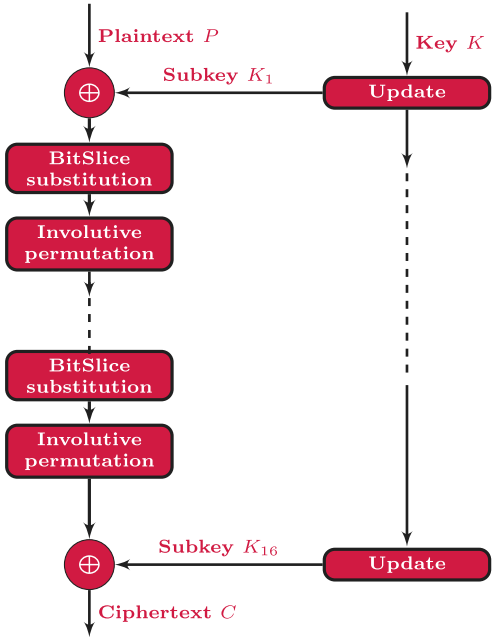

Fig. 2. The diagram of our proposal: ULC.

- **Mixing with a subkey:** it is a simple bitwise XOR operation between the block and the subkey K_i. This is shown in Eq. (2) below.

$$bloc_i = bloc_i \oplus K_i \text{ avec } 1 \leq i \leq 16 \quad (1)$$

- **Substitution using 4 × 4 S-box:** it is a non-linear function using the substitution table represented in Table 3. The Sbox is applied for each column of input block in parallel as follows. The substitution table is represented by logical operations form (AND (&), XOR (⊕), OR (|), NOT()) in order to follow the bitslice technique.

Table 3
S-box of ULC based on Rectangle.

x	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
S(x)	6	5	C	A	1	E	7	9	B	0	3	D	8	F	4	2

This function has been inspired from Rectangle cryptosystem (for which the efficiency and performance have been proven). The function is represented as shown in the following:

1. $T_1 = A_1$; 2. $T_2 = A_0 \& T_1$; 3. $T_3 = A_2 \oplus A_3$;
4. $B_0 = T_2 \oplus T_3$; 5. $T_5 = A_3 | T_1$; 6. $T_6 = A_0 \oplus T_5$;
7. $B_1 = A_2 \& T_6$; 8. $T_8 = A_1 \oplus A_2$; 9. $T_9 = T_3 \& T_6$;
10. $B_3 = T_8 \& T_9$; 11. $T_{11} = B_0 | T_8$; 12. $B_2 = T_6 \& T_{11}$;

with T_i is a temporary variable of 16 bit size, A_i is the line i of the substitution operation's input block and B_i is the line i of the substitution operations output block.

- **Bit permutation:** it is an operation that distributes each 4 bits of the substitution function's output on different subblocks. To make the this operation involutive, we used a transpose matrix of M of 16 × 4 size noted M^t. In the following we show the block represented as a 16 × 4 matrix.

$$M = \begin{bmatrix} b_{0,7} & b_{0,6} & b_{0,5} & b_{0,4} \\ b_{0,3} & b_{0,2} & b_{0,1} & b_{0,0} \\ b_{1,7} & b_{1,6} & b_{1,5} & b_{1,4} \\ b_{1,3} & b_{1,2} & b_{1,1} & b_{1,0} \\ .. & .. & .. & .. \\ .. & .. & .. & .. \\ .. & .. & .. & .. \\ .. & .. & .. & .. \\ .. & .. & .. & .. \\ .. & .. & .. & .. \\ .. & .. & .. & .. \\ .. & .. & .. & .. \\ b_{15,7} & b_{15,6} & b_{15,5} & b_{15,4} \\ b_{15,3} & b_{15,2} & b_{15,1} & b_{15,0} \end{bmatrix}$$

Table 4
Linear cryptanalysis of ULC: linear equation, it probability in every encryption round.

N	Equation	ϵ	Complexity
1	$X_{1,5} \oplus X_{1,7} \oplus X_{1,8} \oplus Y_{1,2} \oplus Y_{1,18} \oplus Y_{1,34} \oplus Y_{1,50} \oplus \sum K = 0$	$\frac{1}{4}$	$2^4 < 2^{80}$
2	$X_{2,2} \oplus X_{2,18} \oplus X_{2,34} \oplus X_{2,50} \oplus Y_{2,17} \oplus Y_{2,21} \oplus Y_{2,25} \oplus Y_{2,29} \oplus Y_{2,33} \oplus Y_{2,37} \oplus Y_{2,41} \oplus Y_{2,45} \oplus Y_{2,49} \oplus Y_{2,53} \oplus Y_{2,57} \oplus Y_{2,61} \oplus \sum K = 0$	$\frac{1}{2^4}$	$2^8 < 2^{80}$
...	...	...	...
10	$X_{10,23} \oplus X_{10,24} \oplus X_{10,27} \oplus X_{10,28} \oplus X_{10,39} \oplus X_{10,40} \oplus X_{10,43} \oplus X_{10,44} \oplus X_{10,55} \oplus X_{10,56} \oplus X_{10,59} \oplus X_{10,60} \oplus Y_{10,22} \oplus Y_{10,23} \oplus Y_{10,26} \oplus Y_{10,27} \oplus Y_{10,30} \oplus Y_{10,31} \oplus \sum K = 0$	$\frac{1}{2^{38}}$	$2^{76} < 2^{80}$
11	$X_{11,22} \oplus X_{11,23} \oplus X_{11,26} \oplus X_{11,27} \oplus X_{11,30} \oplus X_{11,31} \oplus Y_{11,6} \oplus Y_{11,7} \oplus Y_{11,8} \oplus \sum K = 0$	$\frac{1}{2^{41}}$	$2^{82} > 2^{80}$

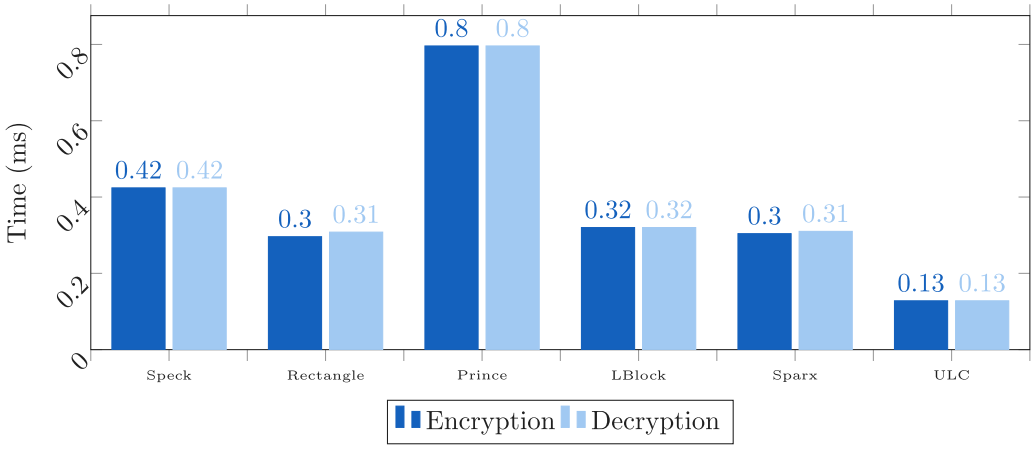

Fig. 3. Evaluation of the execution time of our cryptosystem (ULC) compared to other lightweight cryptosystems.

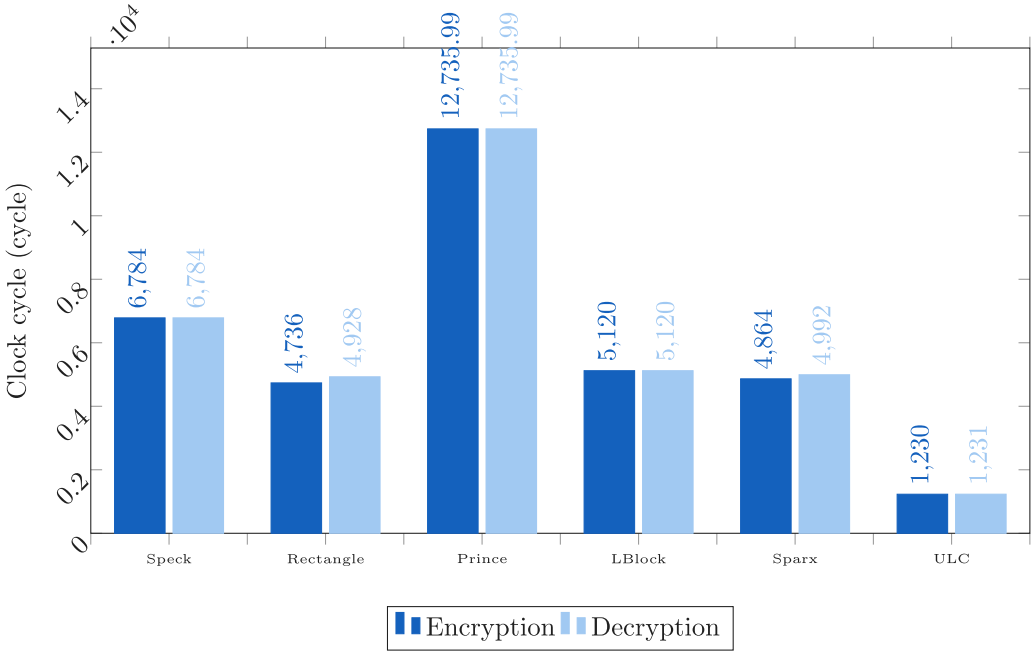

Fig. 4. Evaluation of the clock cycles of our cryptosystem (ULC) compared to other lightweight cryptosystems.

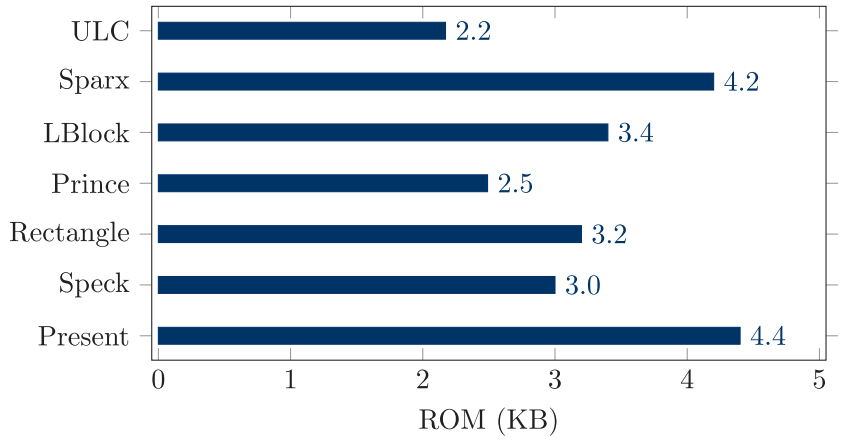

Fig. 5. Evaluation of total memory ROM consumed of our cryptosystem (ULC) compared to other lightweight cryptosystems.

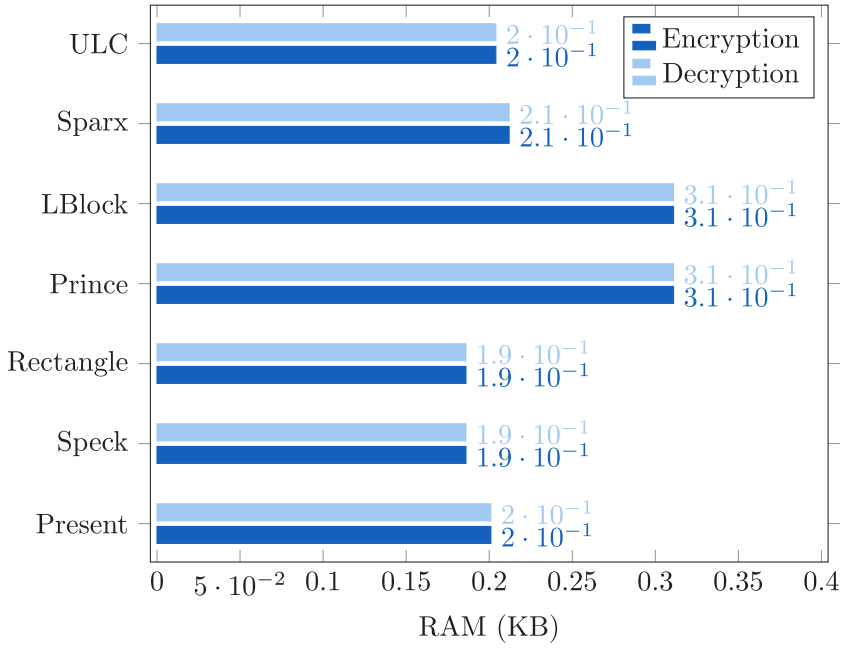

Fig. 6. Evaluation of memory RAM consumed of our cryptosystem (ULC) compared to other lightweight cryptosystems.

$$(M^i)^t = \begin{bmatrix} b_{0,7} & b_{0,6} & b_{0,5} & b_{0,4} \\ b_{0,3} & b_{0,2} & b_{0,1} & b_{0,0} \\ b_{1,7} & b_{1,6} & b_{1,5} & b_{1,4} \\ b_{1,3} & b_{1,2} & b_{1,1} & b_{1,0} \\ .. & .. & .. & .. \\ .. & .. & .. & .. \\ .. & .. & .. & .. \\ .. & .. & .. & .. \\ .. & .. & .. & .. \\ .. & .. & .. & .. \\ .. & .. & .. & .. \\ b_{15,7} & b_{15,6} & b_{15,5} & b_{15,4} \\ b_{15,3} & b_{15,2} & b_{15,1} & b_{15,0} \end{bmatrix}$$

- **keyschedule:** Let us consider the principal key of 80 bits $K = k_{79}k_{78}\dots k_0$. To compute the subkey k_i For the round number i, we consider the following steps:
  - Apply substitution operation S-box on the last 4 bits. $k_{79}k_{78}k_{77}k_{76} = S[k_{79}k_{78}k_{77}k_{76}]$
  - Rotate the left 61 bits of the principal key. Given that the principal key is $k_{79}k_{78}\dots k_0$. After rotation, the subkey became $k_i = k_{18}k_{17}\dots k_{20}k_{19}$.
  - Extract the 64 last bits for the subkey to become as following; $k_i = k_{63}k_{62}\dots k_0 = k_{79}k_{78}\dots k_{16}$

4. Methods' benchmarking

To assess the efficiency of the proposed method, in this section a benchmark is conducted. To do so, we have chosen the most popular cryptosystems designed using the different aforementioned methods (one for each method as stated in the first row of Table 1), including the new proposed method (via the designed cryptosystem). In the experimentations, we used the Arduino UNO platform. This choice is explained by the fact that this platform has limited processing and storage resources and hence could represent a good benchmarking platform for lightweight cryptosystems. Arduino UNO is characterized, in one hand, by the low memory which corresponds to 2 KB RAM and 32 KB ROM. In the other hand, it has an 8 bit processor with a low frequency which is equal to 16 MHz.

The Lightweight cryptosystems were benchmarked based on the following metrics: performance (i.e. execution time and the number of clock cycles), memory usage (i.e. the quantity of consumed memory by measuring the quantity of ROM and RAM used to encrypt and decrypt

Table 5
The output difference ΔY, its holding probability, and the found complexity in each round.

N	Output difference ΔY	Probability	Complexity
1	0×4	$\frac{1}{4}$	$2^2 < 2^{80}$
2	0×08000800	$\frac{1}{16}$	$2^4 < 2^{80}$
3	0×44	$\frac{1}{2^{10}}$	$2^{10} < 2^{80}$
4	...	...	...
14	$0xD000D..D0$	$\frac{1}{2^{74}}$	$2^{74} < 2^{80}$
15	$0 \times 2202000022020000$	$\frac{1}{2^{80}}$	$2^{80} \geq 2^{80}$

a message), and security level (using a random input of 64 bit). This testing is repeated five (5) times and the same results were obtained for all the conducted experiments. For the security level, it is verified by evaluating the confusion and diffusion levels using the metrics explained in [41]. This test is repeated 1000 times (using 1000 randomly generated content) and the average of the results is considered.

4.1. Linear and differential cryptanalysis

A cryptosystem can be considered acceptable (security wise) if and only if the complexity of successful linear and differential attacks are equal or superior to brute force attack [42]. Indeed, a linear cryptanalysis attempts to find a linear approximation of the cipher between plaintext, ciphertext and the secret keys, as described by the following equation:

$$\sum P_u \oplus \sum C_v = \sum K_w \tag{2}$$

where P_u is the uth bit of the input $P = [P_1, P_2, \ldots, P_n]$ and C_v is the vth bit of the output. Where K_w is the kth bit of the key.

This linear approximation holds with a probability p and its quality is measured with a bias ϵ equal to $\epsilon = |P - \frac{1}{2}|$. By knowing the bias, we can determine the needed pairs of numbers of plaintext/ciphertext to break the key. This number can be estimated by the following equation:

$$LC = \frac{1}{\epsilon^2} \tag{3}$$

Table 4 summarizes the found linear equation, its probability and its corresponding bias in every encryption round. Then the result of our evaluation shows that the complexity of any successful linear attack is equal or superior to 2^{82} at the 11th round.

Differential cryptanalysis studies the propagation of two inputs difference along the cipher encryption rounds and how it affects their corresponding outputs difference. Based on [42], we derived in Table 5 the output difference ΔY, its holding probability, and the found complexity in each round. Then the result shows that the complexity of any successful differential attack is equal or superior to 2^{80} at the 15th round.

4.2. Performance evaluation

We evaluated the Lightweight cryptosystems in term of performance using 5 inputs of 64 bit. The same result have been derived for all the conducted tests. The processing time for present cryptosystem is around 65 ms which is equivalent to more than 1 million clock cycles. The results of the other cryptosystems are depicted in Figs. 3 and 4. One can see that ULC processing time is around 0.13 ms which is equal to 1230 clock cycles. Hence, it can be considered as the best choice for software implementation i.e. when the hardware platform is not customizable (predefined). This is explained by the adoption of Bitslice in the proposed method.

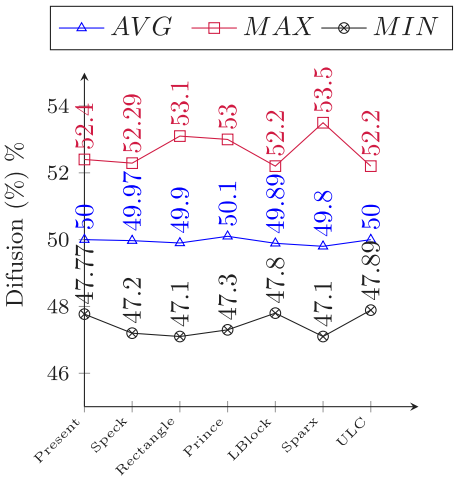

Fig. 7. Evaluation of diffusion level of our cryptosystem (ULC) and the existents lightweight cryptosystems.

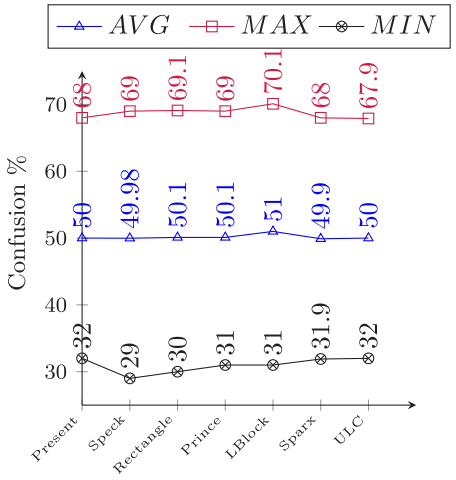

Fig. 8. Evaluation of confusion level of our cryptosystem (ULC) and the existents lightweight cryptosystems.

4.3. Memory consumption

We evaluated the Lightweight cryptosystems in term of memory use using 5 inputs of 64 bit. The same result was obtained for all the conducted tests. The results in Figs. 5 and 6 show the efficiency of the proposed cryptosystem in term of memory usage. One can see in Fig. 5 that the proposed ULC has the lowest ROM usage compared to the other studied systems (2.2 KB for both encryption and decryption operations). It keeps also a rather low RAM usage as shown in Fig. 6 which is very close to Rectangle and Speck (0.203 KB for both encryption and decryption operations). This result is explained by the Involutive property of the proposed method.

4.4. Confusion and diffusion levels

For security level, we evaluated the confusion and diffusion properties of Claude Shannon. These properties are applied for 1000 random plaintexts for each cryptosystem. In Figs. 7 and 8, the maximum, the minimum and the average values of diffusion and confusion levels for the 1000 plaintext are shown respectively. As we can see, ULC's diffusion and confusion are very close to those of Present (which is considered the best cryptosystem regarding security level it provides). These results are guaranteed by the WTS method.

5. Conclusion

This paper focused on the design of lightweight cryptosystems to satisfy the IoT requirements and constraints. We first investigated existing methods used to design block cipher lightweight cryptosystems. We later showed that none of these methods satisfies all IoT constraints together (memory usage, performance and security). For this reason, we proposed the new Ultra-lightweight method to enhance the performance, memory usage and security of IoT devices. The method is validated by a new lightweight cryptosystem ULC (Ultra-lightweight Cryptosystem). The evaluation of ULC showed an improvement with regards to performance and security while keeping a reasonable memory usage compared to the studied cryptosystems. As future work, we will investigate more criteria such as algorithms complexity and the adequacy of each cryptosystem to different kinds of contents.

CRediT authorship contribution statement

Layth Sliman: Conceptualization, Methodology, Investigation, Resources, Writing - original draft, Supervision, Project administration, Funding acquisition. **Tasnime Omrani:** Conceptualization, Methodology, Software, Writing - original draft. **Zahir Tari:** Methodology. **Abed Ellatif Samhat:** Investigation. **Rhouma Rhouma:** Conceptualization, Project administration, Funding acquisition.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

References

[1] Wazid M, Das A, Bhat K V, Vasilakos A. LAM-CIoT: Lightweight authentication mechanism in cloud-based IoT environment. J Netw Comput Appl 2020;150. http://dx.doi.org/10.1016/j.jnca.2019.102496.

[2] Mohd BJ, Hayajneh T, Vasilakos AV. A survey on lightweight block ciphers for low-resource devices: Comparative study and open issues. J Netw Comput Appl 2015;58:73–93.

[3] Wazid M, Das AK, Kumar N, Vasilakos AV, Rodrigues JJPC. Design and analysis of secure lightweight remote user authentication and key agreement scheme in internet of drones deployment. IEEE Internet of Things J 2019;6(2):3572–84. http://dx.doi.org/10.1109/JIOT.2018.2888821.

[4] Bera B, Saha S, Das AK, Vasilakos AV. Designing blockchain-based access control protocol in IoT-enabled smart-grid system. IEEE Internet Things J 2021;8(7):5744–61. http://dx.doi.org/10.1109/JIOT.2020.3030308.

[5] Jing Q, Vasilakos AV, Wan J, Lu J, Qiu D. Security of the Internet of Things: Perspectives and challenges. Wirel Netw 2014;20:2481–501.

[6] Zhou J, Cao Z, Dong X, Vasilakos AV. Security and privacy for cloud-based IoT: Challenges. IEEE Commun Mag 2017;55(1):26–33.

[7] Dinu D, Perrin L, Udovenko A, Velichkov V, Großschädl J, Biryukov A. Design strategies for arx with provable bounds: Sparx and lax. In: International conference on the theory and application of cryptology and information security. Springer; 2016, p. 484–513.

[8] Daemen J, Rijmen V. The wide trail design strategy. In: IMA international conference on cryptography and coding. Springer; 2001, p. 222–38.

[9] Avanzi R. A salad of block ciphers. IACR Cryptol ePrint Arch 2016;2016:1171.

[10] Pornin T. Implantation et optimisation des primitives cryptographiques [Ph.D. thesis], Université Paris 7; 2001.

[11] Canteaut A. Similarities between encryption and decryption: How far can we go? In: Selected areas in cryptography-SAC. 2013.

[12] Hong D, Sung J, Hong S, Lim J, Lee S, Koo B-S, et al. Hight: A new block cipher suitable for low-resource device. In: International workshop on cryptographic hardware and embedded systems. Springer; 2006, p. 46–59.

[13] Beaulieu R, Shors D, Smith J, Treatman-Clark S, Weeks B, Wingers L. Simon and speck: Block ciphers for the Internet of Things. 2015, Cryptology ePrint Archive, Report 2015/585, http://eprint.iacr.org/2015/585.

[14] Hong D, Lee J-K, Kim D-C, Kwon D, Ryu KH, Lee D-G. LEA: A 128-bit block cipher for fast encryption on common processors. In: International workshop on information security applications. Springer; 2013, p. 3–27.

[15] Mouha N, Mennink B, Van Herrewege A, Watanabe D, Preneel B, Verbauwhede I. Chaskey: An efficient MAC algorithm for 32-bit microcontrollers. In: International workshop on selected areas in cryptography. Springer; 2014, p. 306–23.

[16] Lim CH, Korkishko T. mCrypton–A lightweight block cipher for security of low-cost RFID tags and sensors. In: International workshop on information security applications. Springer; 2005, p. 243–58.

[17] Bogdanov A, Knudsen LR, Leander G, Paar C, Poschmann A, Robshaw MJB, et al. PRESENT: An ultra-lightweight block cipher, In: Cryptographic hardware and embedded systems - CHES 2007: 9th international workshop. Proceedings. 2007. p. 450–66.

[18] Guo J, Peyrin T, Poschmann A, Robshaw M. The LED block cipher. In: Preneel B, Takagi T, editors. Cryptographic hardware and embedded systems – CHES 2011. Springer Berlin Heidelberg; 2011, p. 326–41.

[19] Gérard B, Grosso V, Naya-Plasencia M, Standaert F-X. Block ciphers that are easier to mask: How far can we go? In: International workshop on cryptographic hardware and embedded systems. Springer; 2013, p. 383–99.

[20] Beierle C, Jean J, Kölbl S, Leander G, Moradi A, Peyrin T, et al. The skinny family of block ciphers and its low-latency variant mantis. In: Annual cryptology conference. Springer; 2016, p. 123–53.

[21] Banik S, Pandey SK, Peyrin T, Sasaki Y, Sim SM, Todo Y. Gift: A small present. In: International conference on cryptographic hardware and embedded systems. Springer; 2017, p. 321–45.

[22] Zhang W, Bao Z, Lin D, Rijmen V, Yang B, Verbauwhede I. Rectangle: A bit-slice lightweight block cipher suitable for multiple platforms. Sci China Inf Sci 2015;58(12):1–15.

[23] Grosso V, Leurent G, Standaert F-X, Varıcı K. LS-designs: Bitslice encryption for efficient masked software implementations. In: International workshop on fast software encryption. Springer; 2014, p. 18–37.

[24] Journault A, Standaert F-X, Varici K. Improving the security and efficiency of block ciphers based on LS-designs. Des Codes Cryptogr 2017;82(1–2):495–509.

[25] Borghoff J, Canteaut A, Güneysu T, Kavun EB, Knezevic M, Knudsen LR, et al. Prince–A low-latency block cipher for pervasive computing applications. In: International conference on the theory and application of cryptology and information security. Springer; 2012, p. 208–25.

[26] Gong Z, Nikova S, Law YW. Klein: A new family of lightweight block ciphers. In: International workshop on radio frequency identification: Security and privacy issues. Springer; 2011, p. 1–18.

[27] Banik S, Bogdanov A, Isobe T, Shibutani K, Hiwatari H, Akishita T, et al. Midori: A block cipher for low energy. In: International conference on the theory and application of cryptology and information security. Springer; 2014, p. 411–36.

[28] Beierle C, Jean J, Kölbl S, Leander G, Moradi A, Peyrin T, et al. The skinny family of block ciphers and its low-latency variant mantis. In: Annual cryptology conference. Springer; 2016, p. 123–53.

[29] Standaert F-X, Piret G, Gershenfeld N, Quisquater J-J. Sea: A scalable encryption algorithm for small embedded applications. In: International conference on smart card research and advanced applications. Springer; 2006, p. 222–36.

[30] Shirai T, Shibutani K, Akishita T, Moriai S, Iwata T. The 128-bit blockcipher clefia. In: International workshop on fast software encryption. Springer; 2007, p. 181–95.

[31] Poschmann A, Ling S, Wang H. 256 bit standardized crypto for 650 GE–GOST revisited. In: International workshop on cryptographic hardware and embedded systems. Springer; 2010, p. 219–33.

[32] Suzaki T, Minematsu K, Morioka S, Kobayashi E. Twine: A lightweight, versatile block cipher. In: ECRYPT workshop on lightweight cryptography, vol. 2011, 2011.

[33] Wu W, Zhang L. Lblock: A lightweight block cipher. In: Applied cryptography and network security. Springer; 2011, p. 327–44.

[34] Shibutani K, Isobe T, Hiwatari H, Mitsuda A, Akishita T, Shirai T. Piccolo: An ultra-lightweight blockcipher. In: International workshop on cryptographic hardware and embedded systems. Springer; 2011, p. 342–57.

[35] Karakoç F, Demirci H, Harmancı AE. Itubee: A software oriented lightweight block cipher. In: International workshop on lightweight cryptography for security and privacy. Springer; 2013, p. 16–27.

[36] Baysal A, Şahin S. Roadrunner: A small and fast bitslice block cipher for low cost 8-bit processors. In: International workshop on lightweight cryptography for security and privacy. Springer; 2015, p. 58–76.

[37] Patil J, Bansod G, Kant KS. LiCi: A new ultra-lightweight block cipher. In: Emerging trends & innovation in ICT, 2017 international conference. IEEE; 2017, p. 40–5.

[38] Usman M, Ahmed I, Aslam MI, Khan S, Shah UA. Sit: A lightweight encryption algorithm for secure internet of things. 2017, arXiv preprint arXiv:1704.08688.

[39] Dahiphale V, Bansod G, Patil J. Anu-II: A fast and efficient lightweight encryption design for security in IoT. In: Big data, IoT and data science, 2017 international conference. IEEE; 2017, p. 130–7.

[40] Bansod G, Sutar S, Patil A, Patil J. NUX: A lightweight block cipher for security at wireless sensor node level. World Academy of Science, Engineering and Technology; 2018.

[41] Becheikh R, Omrani T, Rhouma R, Belghith S. Risc: A robust image symmetric cryptosystem. Multimedia Tools Appl 2018;1–28.

[42] Heys HM. A tutorial on linear and differential cryptanalysis. Cryptologia 2002;26(3):189–221.