Hindawi
Security and Communication Networks
Volume 2018, Article ID 9802475, 11 pages
https://doi.org/10.1155/2018/9802475

Research Article

RK-AES: An Improved Version of AES Using a New Key Generation Process with Random Keys

Rahul Saha,[1] **G. Geetha**,[1] **Gulshan Kumar**,[1] **and Tai-hoon Kim**[2]

[1]*Lovely Professional University, Jalandhar-Delhi, G.T. Road, Phagwara, Punjab, India*
[2]*Department of Convergence Security, Sungshin Women's University, Seongbuk-gu 02844, Republic of Korea*

Correspondence should be addressed to G. Geetha; gitaskumar@yahoo.com

Received 4 July 2018; Revised 12 October 2018; Accepted 25 October 2018; Published 6 November 2018

Academic Editor: Clemente Galdi

Advanced Encryption Standard (AES) is a standard algorithm for block ciphers for providing security services. A number of variations of this algorithm are available in network security domain. In spite of the strong security features, this algorithm has been recently broken down by the cryptanalysis processes. Therefore, it is required to improve the security strength of this algorithm as AES is popular in commercial use. In this paper, we have shown the reasons of the loopholes in AES and also have provided a solution by using our Symmetric Random Function Generator (SRFG). The use of randomness in the key generation process in block cipher is novel in this domain. We have also compared our results with the original AES based upon some parameters such as nonlinearity, resiliency, balancedness, propagation characteristics, and immunity. The results show that our proposed version of AES is better in withstanding attacks.

1. Introduction

Cryptology is an important domain of security measure for providing confidentiality, authentication, and other services [1]. It contains two major parts as cryptography and cryptanalysis. With the progress of technology, where the new cryptographic algorithms are emerging, the cryptanalysis processes are also getting improved; to countermeasure those more secure algorithms are getting developed. So, the cyclic process of cryptography and cryptanalysis goes on. The trend of converging to IoT exhibits an urge of improving the cryptographic algorithms for applications to be secure [2, 3]. Cryptographic algorithms are broadly categorized in two ways: (a) block ciphers and stream ciphers depending upon the format of the message processing; (b) symmetric and asymmetric depending upon number of keys used for the algorithms [1]. Designing such algorithms is another concern where a number of principles are needed to be maintained such as key size, message size, number of rounds, round function, and so on. The selection of key and its size is a major concerning factor in cryptography. A weak key can reveal the plaintext message with least time. Though we know that cryptographic algorithms face brute-force attacks problems, brute-force is not considered as its complexity is higher than any other process of cryptanalysis. The objective of a third party attacker is to break the ciphertext code or to reveal the key or part of the key to get access of the plaintext. So, the weak keys must be avoided in the algorithms. Further, it may happen that the previously considered strong key is now made weak by the sophisticated technology or large computational abilities of the attackers. So, the need of strength analysis to withstand with attacks makes the evolving changes in the cryptographic algorithms.

Cryptographic algorithms primarily depend on the structure of the algorithms and their corresponding functions [4]. Apart from using basic gates such as AND, OR, NOT, and XOR in the algorithms, researchers also have shown some specialized Boolean functions for the symmetric property. The generic Boolean functions have created the basic functionalities of generating any cryptographic function. However, the technology progress and enhancing computational ability of the attackers have urged a need of introducing new features in the function generators so that they can provide more strength to the ciphers. Physical Unclonable Functions (PUFs) [5, 6] are providing solutions for this but as per the cryptographic features requirements; PUFs are

not efficient for cryptographic algorithms. Further, PUF is applicable for FPGA implementation as it is more hardware oriented. Though the objective of the presented approach and PUF is same their orientation and process is totally different. Moreover, it has been shown that PUF is used as seed which again leads to the tendency of pseudorandomness in key generation process which is not desirable. Balancedness, nonlinearity, resiliency, immunity, correlation, and propagation characteristics are some of the important parameters to evaluate the strength of the ciphers. In this paper, we have considered Advanced Encryption Standard (AES) for our experimentation of randomness feature. We have attributed the key generation module of AES undergoing through our Symmetric Random Function Generator (SRFG) [7]. We have evaluated the modified AES with the parameters said above.

The rest of the paper has been organized as follows. Section 2 summarizes the various attacks on AES algorithm. Section 3 describes the original AES algorithm. Section 4 shows the proposed modification for AES and in Section 5 we have explained its properties. Section 6 analyses the security and Section 7 compares the related results. Finally, Section 8 concludes the paper.

2. Related Work

One of the most popular and commercialized algorithms is AES. This algorithm provides the encryption for web security processes as used by different applications such as e-commerce, router applications, and WiFi security. Being so rigorously used in real life applications, AES faces a number of attacks. Some of the recent attacks are mentioned below.

A new kind of fault base attack has been proposed in [8] which uses zero valued sensitivity model for masked AES. Combining the Faulty Sensitivity Analysis (FSA) and zero valued sensitivity, the proposed method of cryptanalysis is able to break code of the S-boxes in masked AES. The attack procedure shows that the zero value input of S-box reveals the key eventually. The authors in the paper [9] have shown a differential faulty approach used in the mix column component of AES. The results show that AES-128 is breakable by such process only using two faulty inputs of ciphertexts. This attack has been proved better as compared to other differential attacks on AES as shown in [10–12]. Another improved version of faulty attack on AES has been executed in the paper [13]. The authors show that a single random byte fault at the input of the eighth round of the AES algorithm is sufficient to deduce the block cipher key. Simulations show that when two faulty ciphertexts pairs are generated, the key can be exactly deduced without any brute-force search. The minimal fault against AES has been used in [14]. The authors show that AES-192 is breakable by using two pairs of correct and fault ciphertexts whereas AES-256 is broken by using three pairs of correct and fault ciphertexts. The work shown previously in [13] was having a key space of 2^{32} which has been reduced by the authors in [15]. Key recovery attacks on AES have been described in [16]. In this paper, the authors have shown practical complexity based attacks against AES-256. The use of two related keys and 2^{39} time complexity has been proved to be sufficient to recover the complete 256-bit key of a 9-round version of AES-256. Another attack works on 10 round version of AES-256 in 2^{45} time complexity. An improved version of the previous related key attack has been shown in [17] against round transformation and key expansion module in AES. The round has been now minimized from 9^{th} to 7^{th} which means that AES is vulnerable even for the starting rounds. The complexity of the attack has also been reduced from 2^{192} to 2^{104}. Another voltage based fault induction method has been introduced in [18]. The authors show a fault model for a constantly underfed RISC CPU. The faults are described in terms of position, recurring patterns, and timing, then the corresponding errors induced in the computation outcomes are specified. The model also support multibit patterns. The use of biased faults also provides an efficient way to for fault injection attacks in cryptanalysis. Such a procedure has been shown in [19].

A collision based attack against AES-192/256 has been shown in [20]. The authors have used 4-round distinguisher for 7-round reduced AES. In the paper [21], the authors have used variable key for AES sing pseudorandom number generator for providing better security to the algorithm, but the approach faces the problem of using biased keys against AES rounds. Biased keys are able to reveal the pseudorandomness of the approach and the key is deduced further by applying differential methods or fault injection as shown before. Multiple deductions-based algebraic trace driven cache attack on AES has been shown in [22]. The behaviour of the cache reveals the input whole or partially. Same input to a particular module and the changes of the cache properties are the key features of this approach. The authors have identified the causes of a bias fault and also have compared different biased fault attacks introduced till. Quantum related key attacks have been shown in [23].

A solution to the fault based injection attacks has been provided in [24]. The proposed scheme is independent of S-box and inverse S-box and achieves more than 95% fault coverage. A recent approach against fault injection or fault analysis has been shown in [25]. It combines the principles of redundancy with that of fault space transformation to achieve security against both DFA and DFIA based attacks on AES-like block ciphers.

After surveying the attacks on AES, it is obvious that fault injection attacks are more efficient in revealing the key in AES. Such fault injections are using the biased input too to distinguish the subkeys or other parts of the algorithm. Moreover, as AES is depending upon finite field operations of 8-bit bytes, the attacks are also executable with finite quantified complexities as we have seen above. The biased inputs along with fault bytes create error in the process and those are denoted for performing differential analysis or linear analysis. Eventually, the key is revealed. Therefore, to overcome such problems, we have introduced the randomness and the balanced symmetric feature in the functional output, specifically in the keys. We have named this modified AES as Random Key AES (RK-AES). As a result, even though attackers are deducing a part of key or injecting a

b_{00}	b_{01}	b_{02}	b_{03}
b_{10}	b_{11}	b_{12}	b_{13}
b_{20}	b_{21}	b_{22}	b_{23}
b_{30}	b_{31}	b_{32}	b_{33}

Figure 1: State matrix representation.

biased fault, the fault will be converted to a symmetric output rather than revealing the original key or plaintext.

The main contributions of our research work are as follows:

(1) Use of randomness in key generation process of AES.

(2) Confirming high nonlinearity, resiliency, balancedness, propagation, and immunity in key generation process.

(3) Ensuring high confusion and avalanche effect in key generation.

3. AES Algorithm

The Advanced Encryption Standard (AES) [26] was published by the National Institute of Standards and Technology (NIST) in 2001. AES is a symmetric block cipher where a single key is used for both encryption and decryption process. The input and output for the AES algorithm each consist of sequences of 128 bits. The key used in this algorithm consists of 128, 192, or 256 bits. AES operates on 8-bit bytes. These bytes are interpreted as the elements of finite field using the following polynomial representation:

$$f(x) = b_{n-1}x^{n-1} + b_{n-2}x^{n-2} + \cdots + b_1 x + b_0 = \sum_{i=0}^{n-1} b_i x^i \quad (1)$$

where each b_i is having the value of 0 or 1.

The 128-bit input block of AES is arranged in a state matrix of size 4×4 as shown in Figure 1. The elements of the matrix are represented by the variable b_{ij} where $0 \le i, j \le 3$ and i,j denotes the row and column number, respectively. Depending upon the size of the bits in keys variables rounds are allowed for AES. For our experimentation we have used key size of 256-bit concept and therefore, the number of rounds used is 14 rounds represented as Nr. Key scheduling algorithm is also used in AES to provide keys to each of the rounds. The design of the key scheduling algorithm is such that the revealing any round key deduces the original input key from which the round keys are derived. The input state matrix is processed by the various round transforms. The state matrix evolves as it passes through the various steps of the cipher and finally produces the ciphertext. Each round in AES follows the following steps.

SubBytes. This is a nonlinear step in the AES. It uses an S-box applied to the bytes of the state matrix. Each byte of the state matrixes is replaced by its multiplicative inverse, followed by an affine mapping as follows:

$$b_i' = b_i \oplus b_{(i+4) \bmod 8} \oplus b_{(i+5) \bmod 8} \oplus b_{(i+6) \bmod 8} \oplus b_{(i+7) \bmod 8} \oplus c_i, \quad \text{for } 0 \le i < 8 \quad (2)$$

where b_i is the i^{th} bit of the byte and c_i is the i^{th} bit of a byte c with the value 63 or 01100011. Thus the input byte x is related to the output y of the S-box by the relation, $y = A.x^{-1} + B$, where A and B are constant matrices [27].

Shift Rows. The last three rows of the state matrix is rotated by a certain number of byte positions. It is executed as follows:

$$s'_{r,c} = s_{(r,(c+shift(r+Nb))mod\ Nb)} \quad \text{for } 0 < r < 4 \text{ and } 0 < c < Nb \quad (3)$$

where Nb is the number of words in the state matrix (each column in the state matrix is considered as word). In AES, $Nb = 4$ always as the input size is 128 bits and arranged in state matrix of size 4×4. Each cell in the state matrix is denoted as s with the index of row r and column c.

MixColumns. This transformation operates on the state matrix column-by-column, considering each column as a four-term polynomials over GF (2^8) and multiplied modulo x^{4+1} with a fixed polynomial a(x), given by

$$a(x) = \{03\}x^3 + \{01\}x^2 + \{01\}x^1 + \{02\} \quad (4)$$

The multiplication process with the columns of state matrix is given by

$$s'(x) = a(x) \otimes s(x) \quad (5)$$

where $\otimes$ is multiplication modulo of polynomials and s(x) is a state in the state matrix.

AddRoundKey. In this process, a round key is added to the state by a simple bitwise XOR operation. Each round key is having the size of Nb words from the key schedule. Those Nb words are each added to the columns of the state matrix to satisfy the following condition:

$$\left[s'_{0,c}, s'_{1,c}, s'_{2,c}, s'_{3,c}\right] = \left[s_{0,c}, s_{1,c}, s_{2,c}, s_{3,c}\right] \oplus \left[w_{round \times Nb + c}\right], \quad \text{for } 0 \le c < Nb \quad (6)$$

where $\oplus$ is the bitwise XOR and round is the round number at which round key is added and $0 \le round < Nr$.

All these steps are performed for each of the round in the AES excluding the last round. In the last round the

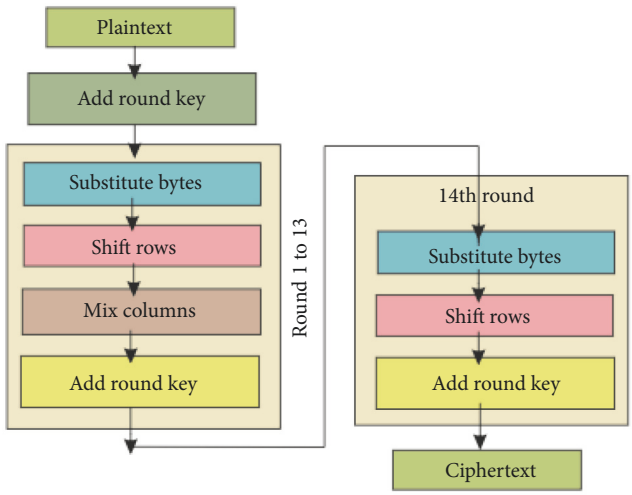


Figure 2: Round function steps in 14-round AES.

MixColumn step is not performed. For a 14-round AES, the round function process is shown in Figure 2. One of the important parts of the round function stages is adding of round keys as these keys are generated by the key expansion routine. The key expansion generates a total of $Nb(Nr + 1)$ words: the algorithm requires an initial set of Nb words, and each of the Nr rounds requires Nb words of key data. The resulting key schedule consists of a linear array of 4-byte words, denoted by $[w_i]$, $0 \le i \le Nb(Nr + 1)$. It uses a function SubWord () that takes these 4-byte words as input and applies S-box to each of these words. Another function Rotword () is used to perform a circular permutation. The round constant array Rcon[i] contains the values specified as $[x^{i-1}, \{00\}, \{00\}, \{00\}]$ with x^{i-1} powers of x in the following equation:

$$Rcon\,[i] = x^{(i-4)/4} \bmod \left(x^8 + x^4 + x^3 + x + 1\right), \quad \text{where } i \text{ is the current round} \tag{7}$$

The key expansion routine for 256-bit keys ($Nk = 8$) is slightly different than for 128- and 192-bit keys. If $Nk = 8$ and i-4 is a multiple of Nk, then SubWord () is applied to w[i-1] prior to the XOR. Nk is the number of 32-bit words of a key.

4. RK-AES

The main problem in the key expansion of the AES algorithm is that the words w_i generated from the original key are related to each other. If any word is traceable, the overall key is deduced by the differential method or liner methods of cryptanalysis. Though the XOR operation, S-boxes, and the shifting in g function, shown in Figure 3, are providing the confusion characteristics to the algorithm, the reverse engineering process can easily get back to the original key space. Moreover, the biased inputs in the key space reveal the differences between the words to partially gain the key space. To solve this problem in AES, we have modified the key expansion module of AES with Symmetric Random Function Generator (SRFG) [7]. SRFG produces the symmetric balanced output in the sense of the number of 1's and 0's in the output string irrespective of the input string. It outputs a combined function comprised of universal GATEs (AND, OR, NOT, and XOR). The expression for the proposed combined function generator is given as

$$f_c = \otimes f_i^L \tag{8}$$

where i = 1, 2, . . . , 4, four universal GATES: AND, OR, NOT, and XOR; L represents the expression length (number of terms in the combined function f_c); and ⊗ represents the random combination. In our experiments we have used L = 5. To emphasize the randomness in such combined function generator, the above equation can be further expressed in terms of N input variables' randomness in selection, as shown in (2).

$$f_c\,(V_1, V_2, \ldots, V_N) = \otimes f_i^L \;\; [\text{rand}\,(V_1, V_2, \ldots, V_N)] \tag{9}$$

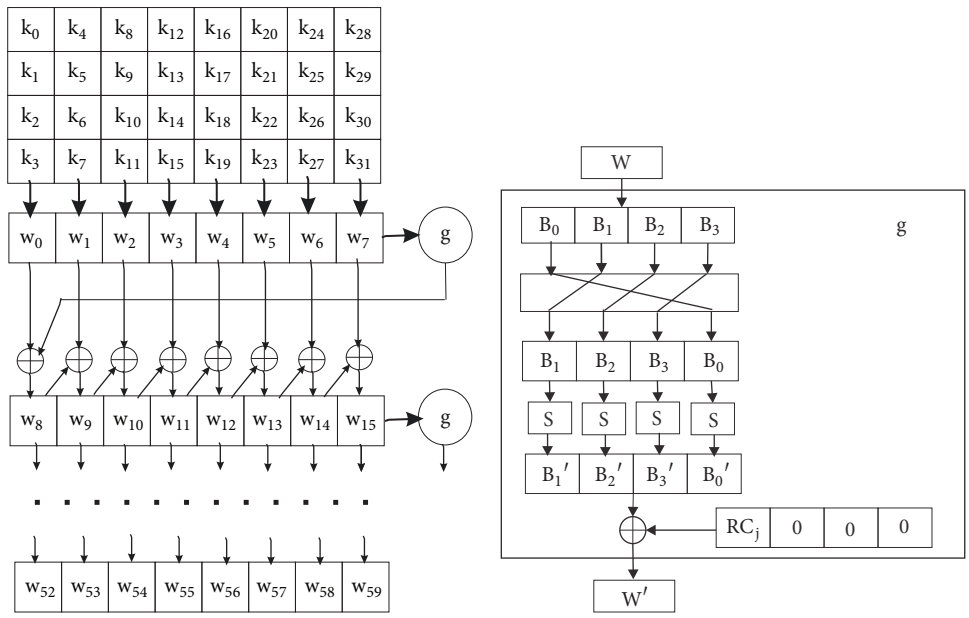
For, our experimentation, the above equation is rewritten as

$$f_c\,(V_1, V_2) = \otimes f_i^5 \;\; [\text{rand}\,(V_1, V_2)] \tag{10}$$

The main objective of adding SRFG in AES is to enable the key expansion module with some randomness feature. This will help to prevent deducing the words of keys even though partial key is in hand. The modified key expansion module has been shown in Figure 4; the changes are highlighted in yellow colour. The randomness of SRFG has been used in three parts: first, in the function of g, secondly, the recursive word generation from key spaces, and thirdly but most prominently, addition of RC_i and SRFG for generating the words from w_0 to w_7. According to Figure 4(a), each column in the key space is considered as w_i word. As the key size is 256 bits, we shall have eight words $w_0, w_1, w_2, w_3, w_4, w_5, w_6, w_7$ in the very first step. The 8^{th} word, i.e., w_7, is going through a function g. This function is also using SRFG just before the output of the function as in Figure 4(b). The output of g is then used to generate the other words processing through a series of SRFGs. The same process is repeated till we get the required number of words for the 14 rounds in AES. For the decryption process, we have saved the generated words and used them reversely with the ciphertext to get back to the plaintext. In future, we shall work upon direct transmission of the keys rather than storing them for decryption.

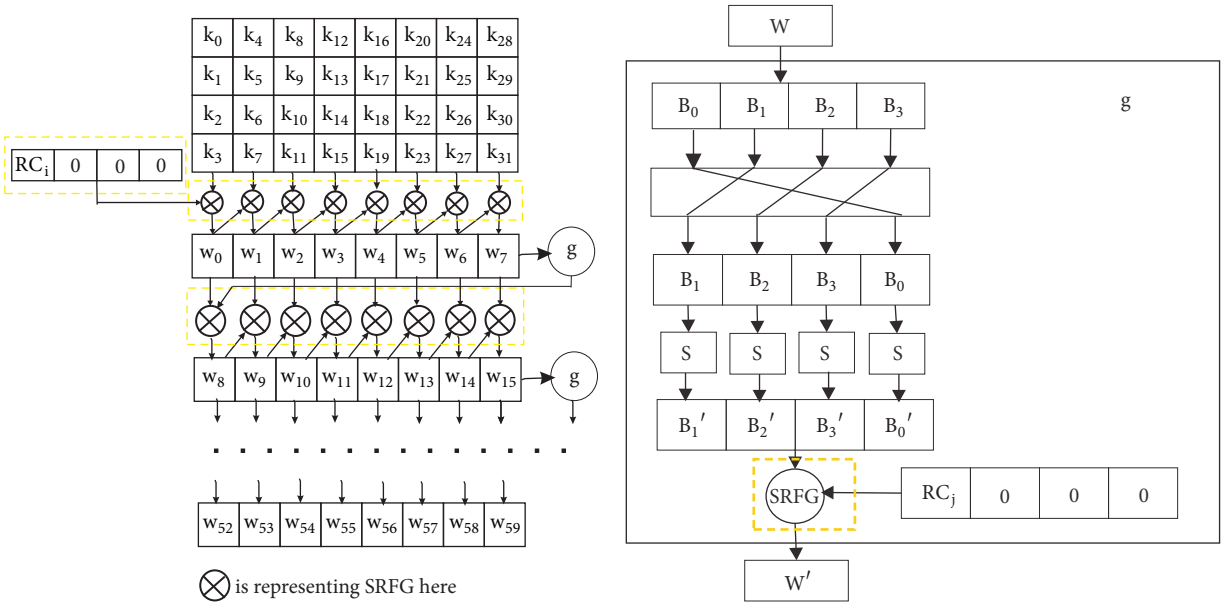
5. Feature Analysis of RK-AES

We have emphasized the key generation module of AES-14 round, so that the effect of biased inputs in the key bytes can be removed from deducing the overall key bytes. The keys are deducing if the cryptanalysis process is able to infer a linear or differential equation out of the words generated from the key expansion module. For the cryptanalysis process, it is not always necessary to have the whole key in hand; rather a single part of key if in the capture, the relationship between different words is sufficient in revealing the overall key space. With the progress of cryptanalysis technologies,



(a) Word key generation (b) g function

Figure 3: Key expansion for 14-round AES.



(a) Word key generation (b) g function

Figure 4: Proposed key expansion for 14 round AES.

generating such relations or deducing keys from subkeys is getting faster with less complexity as we have seen in the literature review. We have identified some of the parameters for our proposed key-expansion module for RK-AES such as nonlinearity, balancedness, resiliency, propagation criterion, and immunity. Each word w_i in the key space is comprised of 32 bits (4 bytes) which is considered as 32-bit word vector in our experimentation. Let $\mathscr{B}_2$ be the set of all symmetric random combined functions on two variables of all the functions from F_2^2 into F_2 where $F_2^2 = (w_1, w_2) \mid w_i \in F_2$. F_2 is the finite field of two elements 0, 1 and $\oplus$ is any operation of the field F_2.

Any combined function $f_c \in \mathscr{B}_2$ of five terms is expressed as a polynomial which is basically termed as Algebraic Normal Form (ANF) of the function and given as

$$f_c(w_1, w_2) = \oplus \lambda_u \left(\prod_{i=1}^{2} \text{rand}(w_i)^{u_i} \right)^s, \quad \lambda_u \in F_2,\ u \in F_2^2 \text{ and } L \in \mathbb{Z} \tag{11}$$

$$\lambda_u = \oplus f_c(v), \quad w \preceq u,\ \forall w_i = w_{i_1}, w_{i_2}, \ldots, w_{i_{32}} \tag{12}$$

where

$$\left(w_{i_1}, w_{i_2}, \ldots, w_{i_{32}}\right) \preceq (u_1, u_2, \ldots, u_{32}) \quad iff\ \forall i, j, w_{i_j} \preceq u_i\ and\ j = 1, 2, \ldots, 32 \tag{13}$$

The output of f_c depends on the weight of its input variables (number of 1s in the variable). As a result, f_c corresponds to a function $g_c : \{0, , 1, \ldots, 32\} \longrightarrow F_2$ such that $\forall x \in F_2^2$, $f_c(x) = g_c(wt(x))$. The sequence $g_c(f_c) = (g_c(0), g_c(1), \ldots, g_c(32))$ for 32-bit word vector is considered as simplified value vector of f_c. To establish the relation between simplified value vector and arithmetic normal form, (11) can be rewritten as shown in (13).

$$f_c(w_1, w_2) = \oplus\lambda_f(j) \oplus \left(\prod_{i=1}^{2} \text{rand}(w_i)^{u_i}\right)^L \tag{14}$$

$$= \oplus\lambda_f(j)\, \mathcal{X}_{j,N} \tag{15}$$

where $\lambda_f(j)$, $u \in F_2^2$, and $L \in \mathbb{Z}, j = \{1, 2\}$. $\mathcal{X}_{j,N}$ is the elementary polynomial of degree j with 2 variables. The coefficients of arithmetic normal form of f_c are represented by 32-bit vector, $\lambda(f_c) = \{\lambda_f(0), \lambda_f(1), \ldots, \lambda_f(32)\}$, called simplified vector of ANF of f_c.

5.1. Nonlinearity. Nonlinearity is an important design characteristic for cryptographic functions used in cryptographic algorithms to prevent different types of correlation or linear attacks or even related attacks. This feature is depending on the bits of the word vectors w_i. w_i is also considered as the affine transformations of the functions generated from the SRFG used. The nonlinearity is calculated by the hamming distance between two affine transformations. For example, two word are w_i and w_j of 32 bits each.

$$\mathcal{N}l\left(w_{i_k}, w_{j_k}\right) = \sum_{k=1}^{n} w_{i_k} \neq w_{j_k}, \quad where\ n = 32 \tag{16}$$

Each of the rounds in AES is using 4 words (128 bits) as subkeys. The nonlinearity between two subkeys used for any two rounds r_i, r_j can be measured as

$$\mathcal{N}l\left(r_i, r_j\right) = \sum_{k=1}^{n} r_{ik} \neq r_{j_k}, \quad where\ n = 128 \tag{17}$$

5.2. Balancedness. Balanced property of our proposed key expansion function f_c exists if its simplified value vector g_c follows the following condition:

$$\forall i = \{1, 2\}, \quad g_c(i) = g_c(2 - i) \boxplus 1, \quad where \boxplus\ is\ sum\ over\ F_2 \tag{18}$$

The above equation also provides the feature of trivial balancedness corresponding to symmetric functions. Therefore, f_c verifies the condition $D_1 f_c = 1$. The functions having $D_1 f_c = 1$ do not exist for even values of n (here n = 32 for words and n = 128 for rounds) because for any word vector w such that $wt(w) = n/2$ (where $wt(w)$ is the weight of word vector defined as number of 1s in it), we can calculate the $D_1 f_c$ as

$$D_1 f_c = f_c(w) \boxplus f_c(w + 1) = g_c\left(\frac{n}{2}\right) \boxplus g_c\left(\frac{n}{2}\right) = 0 \tag{19}$$

5.3. Resiliency. The correlation between the output of the key expansion function and a small subset of its input variables leads to the correlation attack [28], linear or differential cryptanalysis [29]. Therefore, it is necessary for the key expansion function to achieve the high resiliency property. A function f_c of N variables each of having n bits is m-resilient if it remains balanced when any m input variables are fixed and remaining $(n - m)$ bits are altered. The function is more resilient if m is higher. The property of resiliency is related to the weights of the restrictions of the f_c to some subspaces.

$\forall f_c \in B_2$ and any affine any subspace $\mathcal{S} \subset F_2^2$, the restriction of f_c to $\mathcal{S}$ is the function given as

$$f_{\mathcal{S}} : \mathcal{S} \longrightarrow F_2 \tag{20}$$

$$x \longrightarrow f_c(x), \quad \forall x \in \mathcal{S} \tag{21}$$

where, $f_{\mathcal{S}}$ can be determined by the with a function of $\dim(\mathcal{S})$ variables. The subspace $\mathcal{S}$ is spanned by k canonical basis vectors and its supplementary subspace is $\overline{\mathcal{S}}$. The restrictions of f_c to $\mathcal{S}$ and to all its cosets are given by a+ $\mathcal{S}$ where $a \in \mathcal{S}$. Being f_c symmetric and balanced, $\mathcal{S}$ is represented as $\mathcal{S} = (s_1, s_2, \ldots, s_k)$ and $f_{a+\mathcal{S}}$ becomes symmetric and balanced too. Moreover, for all $s \in \mathcal{S}$, we can write the following:

$$f_{a+\mathcal{S}}(s) = f(a + s) = g_c(wt(a) + wt(s)) \tag{22}$$

which actually depends upon the weight of s when a is fixed. The simplified value vector and the simplified ANF vector of $f_{a+\mathcal{S}}$ can be deduced from f_c as given below.

$$g_{c_{f_{a+\mathcal{S}}}}(i) = g_c(i + wt(a)), \quad \forall i,\ 0 \leq i \leq k \tag{23}$$

$$\lambda_{c_{f_{a+\mathcal{S}}}}(i) = \oplus\lambda_f(i + j), \quad \forall i,\ 0 \leq i \leq k\ and\ j \preceq wt(a) \tag{24}$$

5.4. Propagation Criterion. Propagation criterion is determined by the cryptographic properties of the derivatives of the functions. For the efficiency of a cryptographic function, the function needs to propagate its properties to all its derivatives. All derivatives of the key expansion function are linearly equivalent when they have a fixed hamming weight of $n/2$ [7]. Our proposed approach of key expansion N variables applied from our previous work [7] satisfies the propagation criterion of degree k and order m if any affine function obtained from the outputs by keeping m input bits constant

satisfies the propagation criterion of degree k. Considering each round for experimentation, one has the following.

Let $f_c \in \mathscr{B}_2$ and let $r_i, r_j \in F_2^2$, $\forall i, j = 1, 2, \ldots, 14$ such that $wt(r_i) = wt(r_j) = n/2$. Then, $D_{r_i} f_c$ and $D_{r_j} f_c$ are linearly equivalent.

This signifies that if we change the input variables with a linear permutation μ of F_2^2, $D_{r_i} f_c = D_{r_j} f_c \circ \mu$, where $\circ$ is composite function. The permutation μ exists on the variable in a way so that that $r_j = \mu(r_i)$. Since f_c is symmetric and balanced, we can have

$$D_{r_j} f_c(\mu(a)) = D_{r_i} f_c(a), \quad \text{where } a \in \overline{\mathcal{S}} \tag{25}$$

Let k be an integer, $1 \le k \le n-1$, $z \in \overline{w_i} = (w_{i_1}, w_{i_2}, \ldots, w_{i_{n-k}})$, and $\epsilon_k = w_{n-k+1} + \cdots + w_n$. Then for any $z = a + r_j$, with $a \in \overline{\mathcal{S}}$, then we can have the following:

$$wt(z) = wt(a) + wt(r_j) \tag{26}$$

$$\begin{aligned} wt(z + \epsilon_k) &= wt(a) + wt(r_j + \epsilon_k) \\ &= wt(a) + k - wt(r_j) \end{aligned} \tag{27}$$

Thus, $\forall\ a \in V$,

$$\begin{aligned} D_{\epsilon_k} f_c(a + y) &= f_c(a + b) \boxplus f_c(a + \epsilon_k + r_j) \\ &= wt(a) + wt(r_j + \epsilon_k) \\ &= wt(a) + k - wt(r_j) \\ &= g_c(wt(a) + w(r_j)) \\ &\quad \boxplus (wt(a) + k - w(r_j)) \end{aligned} \tag{28}$$

Equation (28) signifies that g_c follows the symmetric property. This means that partial derivatives of our proposed key expansion outputs are also propagated with the propagation features.

5.5. Immunity. The proposed key expansion module deals with the variables (words) with 32 bits (no modification has been done on bit size). Two types of immunity are in concern: correlation immunity and algebraic immunity. For, correlation immunity, considering each of the two input variables w_i as 32-bit binary vector the outputs are correlation immune if

$$Prob(f_c = w_i) = \frac{1}{2}, \quad 1 \le i \le 32 \tag{29}$$

The probability distribution must be equal for all the bits and therefore, the output words w_o have the following property:

$$\left|\min\left[M_0\left(w_o, (w_o)^r\right) - M_1\left(w_o, (w_o)^r\right)\right]\right| = \min[m] \longrightarrow 0 \tag{30}$$

where $[M_0(w_o, (w_o)^r)]$ is the matching of output words from the key expansion process and its reverse with respect to value 0 and $[M_1(w_o, (w_o)^r)]$ is the matching of output words from the key expansion process and its reverse with respect to value 1. Following the above property, an interesting feature of our proposed key expansion module has been identified and the proposition has been given as follows.

Proposition 1. *In AES-256, if* $[M_0(w_o, (w_o)^r)] = m_0$ *and* $[M_1(w_o, (w_o)^r)] = m_1$, *then* $Nl(w_o, (w_o)^r) = m_0 + m_1$.

Algebraic immunity is related to the annihilator of a function [30]. To evaluate this property for our proposed key expansion we can consider the following.

Given, $f_c \in \mathscr{B}_2$, any function of the set $A(f_c) = \{g \in B_2 \mid gf = 0\}$ is defined as the annihilator of the function f_c. The algebraic immunity of f_c is denoted by $AI(f_c)$ is minimum degree of all nonzero annihilators of $f_(c)$ or $f_(c)+1$. The value of $AI(f_c)$ is given as

$$\begin{aligned} AI(f_c) = \min[\deg(g) \mid\ & g \ne 0, \\ & g \in A(f_c) \cup A(f_c + 1)] \end{aligned} \tag{31}$$

As we have used SRFG to generate the output words, minimum degree is always $n/2$. Therefore, the algebraic immunity of the outputs from it is always n/2 which is always optimal.

6. Security Analysis of RK-AES

In the above section, we have analysed the overall features of the proposed key expansion modification in AES-256 using the SRFG. To justify the features, in this section we have performed the security analysis on our modified AES key expansion module. We have considered two attacks: related attacks and fault analysis attacks.

6.1. Related Key Attack Analysis. Related key attacks use the linear relations or differential relations among the keys to deduce the original key.

Let *nz* be a known nonzero word difference for input and *o* be an output difference of S-box for the input difference *nz*. To execute the attack with this differences, the difference o can be one of $2^{14} - 1$ values, because of the symmetry of the XOR operation as used in generic AES-256 algorithm and the *nz* difference can be one of $2^{15} - 1$ differences including whitening of keys. Once these differences are in a bounded value region, the probability deducing of the key is also higher. In our proposed modified AES, the nonlinearity feature increases this difference and therefore, the key space of searching also increases drastically. For a 32-bit word in key space, the complexity of searching space increases with the following formula:

$$Complexity\ for\ key\ space\ search = 2^{32}.2^{Nl} \tag{32}$$

where Nl is the value of nonlinearity in the proposed AES key expansion and the average value of $Nl = 20.7$. Therefore, the complexity becomes $2^{52.7}$ which is more than the differential attacks key searching complexities on AES. This shows that our proposed algorithm is preventive in differential attacks.

Furthermore, the attacker uses four related but unknown keys as $K_{u1}, K_{u2}, K_{u3}, K_{u4}$. The objective of the attacker is to recover K_{u1}. The relation required to establish the attack is

$$K_{u2} = K_{u1} \oplus \triangle K^* \tag{33a}$$

$$K_{u3} = K_{u1} \oplus \triangle K' \tag{33b}$$

$$K_{u4} = K_{u1} \oplus \triangle K^* \oplus \triangle K' \tag{33c}$$

where $\triangle K^*$ is the cipher key difference used for the first related-key differential D^0 for 1 to 7 round and $\triangle K'$ is the cipher key difference used for the second related-key differential D^1 used for 8 to 14 round. Assuming that the attacker only has the information regarding $\triangle K^*$ and $\triangle K'$, the back tracing probability to recover any 32-bit words (any word out of the 60 words) is calculated as

$$P(w_i) = \frac{1}{\left(2^n.i.P_L^L.2^V\right)} \tag{34}$$

For our proposed modified AES-256 key expansion, number of bits in each word is n = 32, total number of words including whitening key words is i = 60, total number of expression length L = 5, and total number variables used for each operation is V = 2. Using the values, the probability becomes as

$$P(w_i) = \frac{1}{\left(2^{32} \times 60 \times P_5^5 \times 2^2\right)} = \frac{1}{2^{39} \times 225} \tag{35}$$

The above result show that the probability is too less to recover a single word of AES-256 using our proposed approach of key expansion.

In Figure 4, it is shown that the words are generated using SRFG rather than using simple XOR operation. Therefore, (33a), (33b), and (33c) will not be feasible for our proposed solution of AES using SRFG. It means, the proposed solution is related attack resistant. Moreover, $\triangle K^*$ and $\triangle K'$ are a factor in deducing the key. But, as our proposed solution provides a high nonlinearity, $\triangle K^*$ and $\triangle K'$ are not suitable to recover the words of the key space. From the observation of or experimentation, we have inferred a proposition as follows.

Proposition 2. *Considering $\triangle K^*$ is the cipher key difference used for the first related-key differential D^0 and $\triangle K'$ is the cipher key difference used for the second related-key differential D^1, nonlinearity is inversely proportional to the nonlinearity.*

$$D^0 \vdash \triangle K^* \propto \frac{1}{Nl} \text{ and } D^1 \vdash \triangle K' \propto \frac{1}{Nl} \tag{36}$$

$$\therefore D^0.D^1 \vdash \triangle K^*.\triangle K' \propto \frac{1}{Nl^2} \tag{37}$$

6.2. Fault Injection Analysis. In this part, we have only considered the fault injection in the key bytes. We assume that the faulty key byte is injected in the key matrix for any random original key byte. The faulty input is inferred from the biased input of all 0 bits byte or all 1 bits byte. In the original AES, using such faulty and biased inputs reveals the relationship among word byte or even words of round. Therefore in original AES, the key recovery space is reduced with less complexity as we have seen in the literature review.

Recollecting (11) and (12), we can have the following proposition for AES-256.

Proposition 3. *For AES-256, using SRFG with two variables and t expression terms, the complexity of key recovery with any two random faulty byte is calculated as*

$$Prob(FI) = \frac{1}{\sum \oplus \lambda_u \left(\prod_{i=1}^{2} \text{rand}(w_i)^{u_i}\right)^t .C_2^{60}} \tag{38}$$

For any random faulty key byte, the output of the layered SRFGs is always nonlinear and balanced. Therefore according to Proposition 2 the differences and/or the linear equations become invalid as the fault is not further propagated to other bytes. Therefore, our proposed key expansion is preventive even in fault injection bytes.

7. Results of Comparison

We have compared our experimentation results of RK-AES with the original AES algorithm. The comparison is done on the basis of some features: nonlinearity, balancedness, resiliency, propagation criterion, correlation immunity, and algebraic immunity. As we have modified only the key expansion module, the results are derived only for key expansion only without involving the plaintext processing or transformations in round function. We have compared 215 data samples for each RK-AES and original AES. The comparison results are shown in Table 1 by averaging all the results.

The comparison results in Table 1 signify that our proposed modification of key expansion is working efficiently in AES in terms of the above said features. Moreover, the balancedness and the correlation immunity are 0 in original AES. Our proposed modification is providing a higher value for balancedness which is useful for preventing bitsum attacks [31]. The high correlation immunity will also help the modified AES to prevent correlation attacks [28].

Moreover, we have compared the computation time for our experiments with the original AES algorithm. In this comparison too, we have assumed the time for plaintext processing and transformations in round function are constant as no modification has done on them. Therefore, Table 2 only compares the time taken for the key expansion process.

The time comparison results show that using the SRFG in AES key expansion modification is increasing the time consumption in generating the key words and thus contributing to the trade-off between security and time consumption. To support this trade-off and overcome with the security issues, we have also compared the attack for both the original AES and the modified AES.

Table 3 describes the fact that the cost of the attacks for our proposed RK-AES is much higher than the original AES due to the use of randomness with SRFG in several layer. This signifies that RK-AES is better in terms of security.

Table 1: Comparison of parameters.

	Order of Non-linearity	Order of Balancedness	Order of Resiliency	Order of Propagation criterion	Order of Correlation immunity	Order of Algebraic immunity
Original AES-256	$n/2 - \log_2 n$	0	$n/2 - 2$	$\log_2 n + n/4$	$n/10$	$n/6$
Proposed RK-AES	$n/2 \times 0.65 + \log_2 5$	$n/2$	$n/2 - 2$	$n \times 0.73 + \log_2 5$	n	$n/2$
n is the value of bits in a word of key space						

Table 2: Time consumption comparison.

Hardware specification for computation: CPU: 2.6Ghz, i3 6th,Gen with 16 GB RAM			
	Average Time Consumption (in milliseconds)		
Schemes	32 bit key words w_0 to w_7	32 bit key words w_8 to $w_5 9$	g function
Original, AES	3.67	5.73	6.32
RK-AES	6.78	8.87	9.33

Table 3: Comparison of cost of attacks.

	Differential cryptanalysis	Linear cryptanalysis	Related key analysis	Fault injection attack analysis in key space
Original AES	2^{32}	$2^{14} - 1$	2^{32}	2^{26}
RK-AES	$2^{52.7}$	2^{60}	$2^{52.7}$	$\approx 2^{129}$

Lastly, we have compared two prime evaluation parameter of encryption algorithms: confusion and avalanche effect. Confusion property requires the statistical relationship of between the ciphertext and key to be more complex. Besides, avalanche effect requires change in the ciphertext bits if any single bit is changed in the key. We have calculated confusion property in terms of nonlinearity and resiliency. The avalanche effect is measured in terms of propagation criterion, correlation immunity, and algebraic immunity. The calculation formula for confusion and avalanche effect have been given below.

$$\begin{aligned} Confusion &= \omega_1 \times Nonlinearity + \omega_2 \times Resiliency \\ &\quad + \omega_3 \times Balancedness \end{aligned} \tag{39}$$

$$\begin{aligned} Avalanche &= \omega_1 \times Propagation\ criterion + \omega_2 \\ &\quad \times Correlation\ immunity + \omega_3 \\ &\quad \times Algebraic\ immunity \end{aligned} \tag{40}$$

where ω_1, ω_2, and ω_3 are the weights assigned to the features. We have considered for our experimentation of RK-AES, $\omega_1 = \omega_2 = \omega_3 = 0.33$, and n = 32 bit.

Therefore, following Table 1, the values for confusion and avalanche effect in RK-AES are

$$\begin{aligned} Confusion_{RK-AES} &= 0.33 \times \left(\frac{n}{2} \times 0.65 + \log_2 5\right) \\ &\quad + \left(0.33 \times \frac{n}{2}\right) + 0.33 \times \left(\frac{n}{2} - 2\right) \\ &\cong 22.621 \end{aligned}$$

$$\begin{aligned} Avalanche_{RK-AES} &= 0.33 \times (\mathrm{n} \times 0.73 + \log_2 5) \\ &\quad + (0.33 \times \mathrm{n}) + \left(0.33 \times \frac{n}{2}\right) \\ &\cong 24.31 \end{aligned} \tag{41}$$

Similarly, we have calculated the values for confusion and avalanche effect in original AES. Considering the orders in Table 1, the values are as follows:

$$\begin{aligned} Confusion_{AES} &= 0.33 \times \left(\frac{n}{2} - \log_2 \mathrm{n}\right) + (0.33 \times 0) \\ &\quad + 0.33 \times \left(\frac{n}{2} - 2\right) \cong 7.59 \end{aligned} \tag{42}$$

$Avalanche_{AES} = 0.33 \times (\log_2 \mathrm{n} + n/4) + (0.33 \times n/10) + (0.33 \times n/6) \cong 15.816$. The similar result of Avalanche effect is also experimented in the bit values of the data samples. Table 4 compares the avalanche effect.

The comparison results of confusion property and avalanche effect also show the improvement of the parameters as compared to the original AES algorithm.

8. Conclusion

AES is a popular symmetric block cipher used by different commercialization sectors. But this algorithm is facing a number of cryptanalysis effects as we have seen in the literature review. Therefore, in this paper we have tried to solve the problem by incorporating the changes in key expansion

Table 4: Avalanche effect comparison.

	Weighted value of Avalanche	Average number of bits changed
Original AES	15.816	17.3 bits out of 32 bits
Proposed RK-AES	24.310	≈ 25 bits out of 32 bits

module. The highlight of this work is to apply randomness in the key generation. Moreover, as per our previous work, using SRFG as a cryptographic function in AES has been proved beneficial. The justification for the same has been already shown in the paper. The results show that RK-AES is having three times better confusion property and 53.7% better avalanche effect as compared to the original AES. The limitation of our present work is about the time taken by the modified key expansion module which is actually creating a trade-off between security and time. It is also known that both these two cannot be achieved simultaneously. Therefore, if we ignore the part of the time, our proposed RK-AES is efficient in all respects of cryptographic algorithms. Furthermore, being a symmetric key algorithm AES uses the single key for both encryption and decryption. In our present work, the round keys are stored separately as each round keys are generated randomly and are used for decryption accordingly. In our future work, we shall try to work on the trade-off and also about the storing process of round keys.

Data Availability

The data will be made available on request.

Conflicts of Interest

The authors declare that there are no conflicts of interest regarding the publication of this paper.

References

[1] W. Stallings, *Cryptography and Network Security: Principles and Practices*, Pearson, 2005.

[2] S. Sciancalepore, G. Piro, G. Boggia, and G. Bianchi, "Public key authentication and key agreement in iot devices with minimal airtime consumption," *IEEE Embedded Systems Letters*, vol. 9, no. 1, pp. 1–4, 2017.

[3] S. Raza, L. Seitz, D. Sitenkov, and G. Selander, "S3K: Scalable security with symmetric keys - DTLS key establishment for the internet of things," *IEEE Transactions on Automation Science and Engineering*, vol. 13, no. 3, pp. 1270–1280, 2016.

[4] T. W. Cusick and P. Stanica, *Cryptographic Boolean functions and applications*, Elsevier/Academic Press, 2017.

[5] J. Zhang, Y. Lin, Y. Lyu, and G. Qu, "A PUF-FSM Binding Scheme for FPGA IP Protection and Pay-Per-Device Licensing," *IEEE Transactions on Information Forensics and Security*, vol. 10, no. 6, pp. 1137–1150, 2015.

[6] J.-L. Zhang, G. Qu, Y.-Q. Lv, and Q. Zhou, "A survey on silicon PUFs and recent advances in ring oscillator PUFs," *Journal of Computer Science and Technology*, vol. 29, no. 4, pp. 664–678, 2014.

[7] R. Saha and G. Geetha, "Symmetric random function generator (SRFG): A novel cryptographic primitive for designing fast and robust algorithms," *Chaos, Solitons & Fractals*, vol. 104, pp. 371–377, 2017.

[8] Q. Wang, A. Wang, L. Wu, and J. Zhang, "A new zero value attack combined fault sensitivity analysis on masked AES," *Microprocessors and Microsystems*, vol. 45, pp. 355–362, 2016.

[9] G. Piret and J. Quisquater, "A differential fault attack technique against spn structures, with application to the AES and khazad," in *Cryptographic Hardware and Embedded Systems - CHES 2003*, vol. 2779 of *Lecture Notes in Computer Science*, pp. 77–88, Springer, Berlin, Heidelberg, Germany, 2003.

[10] J. Blömer and J. Seifert, "Fault Based Cryptanalysis of the Advanced Encryption Standard (AES)," in *Financial Cryptography*, vol. 2742 of *Lecture Notes in Computer Science*, pp. 162–181, Springer Berlin Heidelberg, Berlin, Heidelberg, 2003.

[11] P. Dusart, G. Letourneux, and O. Vivolo, "Differential Fault Analysis on A.E.S," in *Applied Cryptography and Network Security*, vol. 2846 of *Lecture Notes in Computer Science*, pp. 293–306, Springer Berlin Heidelberg, Berlin, Heidelberg, 2003.

[12] C. Giraud, "DFA on AES," in *Proceedings of the 4th Int Conf AES 2004*, pp. 27–41, 2004.

[13] D. Mukhopadhyay, "An improved fault based attack of the advanced encryption standard," *Lect. Notes Comput. Sci. (including Subser. Lect.Notes Artif. Intell. Lect. Notes Bioinformatics)*, vol. 5580, pp. 421–434, 2009.

[14] C. H. Kim, "Differential fault analysis against AES-192 and AES-256 with minimal faults," in *Proceedings of the 7th International Workshop on Fault Diagnosis and Tolerance in Cryptography, FDTC 2010*, pp. 3–9, USA, August 2010.

[15] M. Tunstall, D. Mukhopadhyay, and S. Ali, "Differential fault analysis of the advanced encryption standard using a single fault," *Inf. Secur. Theory Pract. Secur. Priv. Mob. Devices Wirel. Commun*, pp. 224–233, 2011.

[16] A. Biryukov, O. Dunkelman, N. Keller, D. Khovratovich, and A. Shamir, "Key recovery attacks of practical complexity on AES-256 variants with up to 10 rounds," in *Advances in cryptology—EUROCRYPT 2010*, vol. 6110 of *Lecture Notes in Comput. Sci.*, pp. 299–318, Springer, Berlin, 2010.

[17] J. Cui, L. Huang, H. Zhong, and W. Yang, "Improved related-key attack on 7-round AES-128/256," in *Proceedings of the ICCSE 2010 - 5th International Conference on Computer Science and Education*, pp. 462–466, 2010.

[18] A. Barenghi, G. M. Bertoni, L. Breveglieri, and G. Pelosi, "A fault induction technique based on voltage underfeeding with application to attacks against AES and RSA," *The Journal of Systems and Software*, vol. 86, no. 7, pp. 1864–1878, 2013.

[19] N. Farhady Ghalaty, B. Yuce, and P. Schaumont, "Analyzing the Efficiency of Biased-Fault Based Attacks," *IEEE Embedded Systems Letters*, vol. 8, no. 2, pp. 33–36, 2016.

[20] J. Kang, K. Jeong, J. Sung, S. Hong, and K. Lee, "Collision attacks on AES-192/256, Crypton-192/256, mCrypton-96/128, and anubis," *Journal of Applied Mathematics*, vol. 2013, Article ID 713673, 10 pages, 2013.

[21] S. Sahmoud, "Enhancement the Security of AES Against Modern Attacks by Using Variable Key Block Cipher," *Int. Arab J. e-Technol*, vol. 3, pp. 17–26, 2013.

[22] X. Zhao, S. Guo, F. Zhang et al., "A comprehensive study of multiple deductions-based algebraic trace driven cache attacks on AES," *Computers & Security*, vol. 39, pp. 173–189, 2013.

[23] M. Roetteler and R. Steinwandt, "A note on quantum related-key attacks," *Information Processing Letters*, vol. 115, no. 1, pp. 40–44, 2015.

[24] H. Mestiri, F. Kahri, B. Bouallegue, and M. Machhout, "A high-speed AES design resistant to fault injection attacks," *Microprocessors and Microsystems*, vol. 41, pp. 47–55, 2016.

[25] S. Patranabis, A. Chakraborty, D. Mukhopadhyay, and P. P. Chakrabarti, "Fault Space Transformation: A Generic Approach to Counter Differential Fault Analysis and Differential Fault Intensity Analysis on AES-Like Block Ciphers," *IEEE Transactions on Information Forensics and Security*, vol. 12, no. 5, pp. 1092–1102, 2017.

[26] *Specification for the Advanced Encryption Standard (AES)*, Federal Information Processing Standards Publication 197, 2001.

[27] J. Daemen and V. Rijmen, *The Design of Rijndael: AES-The Advanced Encryption Standard*, Springer, Berlin, Germany, 2002.

[28] T. Siegenthaler, "Correlation-immunity of nonlinear combining functions for cryptographic applications," *Institute of Electrical and Electronics Engineers Transactions on Information Theory*, vol. 30, no. 5, pp. 776–780, 1984.

[29] Y. Wei and Y. Hu, "Linear-differential cryptanalysis for SPN cipher structure and AES," *Wuhan University Journal of Natural Sciences*, vol. 12, no. 1, pp. 37–40, 2007.

[30] O. R. B. de Oliveira, "An Alternative Method for the Undetermined Coefficients and the Annihilator Methods," 2011, https://arxiv.org/abs/1110.4425.

[31] Amandeep and G. Geetha, "Analysis of bitsum attack on block ciphers," *Journal of Discrete Mathematical Sciences & Cryptography*, vol. 19, no. 4, pp. 875–885, 2016.