Implementation of Reversible Substitution box using LFSR based dynamic key in Advanced Encryption Standard Algorithm

Mrs.B.Sarala, Bhavan Kumar C, Deva S, Dhivyan K

sarala@svce.ac.in, bhar19118@gmail.com, deva432002@gmail.com, dhivyankumar253@gmail.com

Abstract

Data security has been a major concern as that of the faster processing of data. As the capability of data processing is being evolved, the attacks on these devices for data extraction have also been increasing daily. It is well known that the AES algorithm has been registered as the standard encryption model since 2001. The purpose of this work is to optimise the security of current crypto coprocessors with the help of a Linear Feedback Shift Register (LFSR). This project work offers a simple and innovative method for building dynamic and key dependant S-boxes utilising Multiplicative Inverse and Affine transformation. Here AES with plain text (128-bit) given to the S-box, the 128 bits or 16 bytes is converted into four subcomponents each with 4 bytes. Each byte or 8-bit is given to the multiplicative inverse and affine transform, The output of this above process will be 128-bit cipher text. The 128-bit in the LFSR is xor with the output to become a dynamic s-box. The same will be continued for N rounds. Followed by this, reversible logic is applied. Reversible logic has numerous methods. Toffoli, one of the methods in reversible logic, is used to maximize speed and reduce energy consumption. The proposed model of generating dynamic and key depended on s-box is the alternative to the existing s-box. As a result, the efficiency of the s-box increases, and encryption becomes stronger.

I. INTRODUCTION

The Advanced Encryption Standard (AES) is one of the most popular ones. The US government has approved it as a standard for encryption. It is offered in a variety of encryption packages. The National Security Agency (NSA) initially approved an open, widely usable cipher, AES, for use with top-secret data. Joan and Vincent Rijmen, two cryptographers from Belgium, created AES. Rijndael is a block cipher used by AES. The extremely durable AES algorithm has proven impervious to all known cryptographic assaults to date. AES is a symmetric block cipher that can handle 128-bit data blocks with cryptographic keys that are 128 bits long. AES-128 process the data block in 10 iterations, commonly known as "rounds" (AES rounds), of a predefined series of changes. Except for the last round, which is significantly different from the rest, the rounds are identical.

There are four steps in each processing round:
1. Substitute bytes - This method substitutes each byte in the block one at a time using an S-box.
2. Shift rows: A straightforward permutation.
3. Mix column - This substitution technique multiplies each column's data from the shift-row step by the algorithm's matrix.
4. Add round key - The data is XORed with the key for the processing round.

The S-box's responsibility is to reduce the algorithm's vulnerability to algebraic and differential attacks as well as to linear and differential cryptanalysis techniques. The S-box function must be invertible, have no fixed points, and meet the complexity condition. It must also execute quickly and be simple to implement. Examples of this include complimentary fixed points S(a)=a. All 256 8-bit values that could be used are permuted in S-box[7]. The following is the mapping of each unique state byte into a new byte: The byte's four leftmost bits are used as a row value and its four rightmost bits as a column value. These column and row values act as indices in the S-box to choose a specific 8-bit output value. By applying the multiplicative inverse to the plain text in GF (28) and then applying an affine transformation to it, the ByteSub & InvByteSub transformation is calculated. The inverse affine transformation is applied to the 8-bit output cipher text before processing the multiplicative inverse, to calculate the inverse substitution byte.

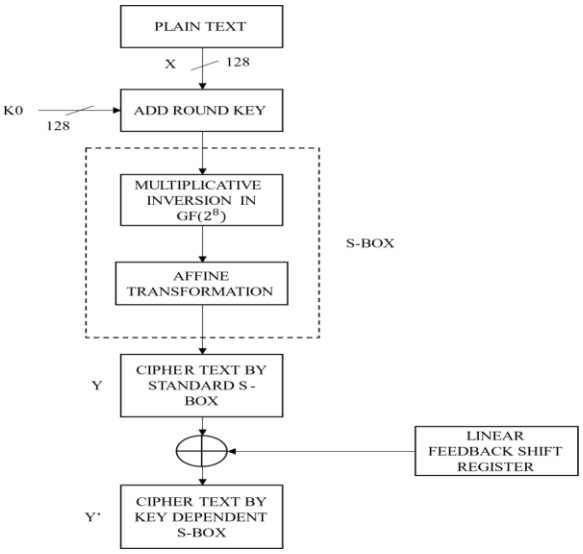


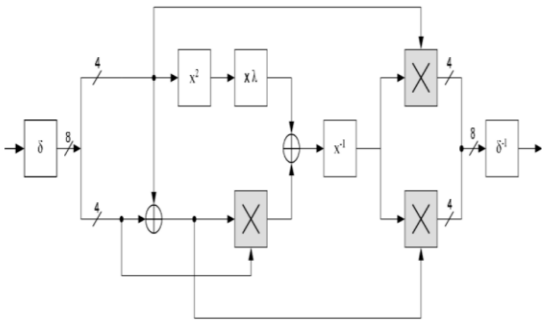
Fig.1 Flowchart of proposed S-box

II. SUBSTITUTION BOX

The S-box operation is a non-linear replacement step carried out during the encryption process in AES (Advanced Encryption Standard). A fixed 16x16 lookup table called the S-box accepts an 8-bit input and produces an 8-bit value. According to the settings in the S-box, each input value is replaced with an equivalent output value. The S-box is intended to create confusion and diffusion, making it challenging to understand how the input and output numbers relate to one another. The AES SubBytes transformation, which is applied to 8-bit of the input block during the encryption process, which uses of the S-box operation. The 8-bit input is replaced with a corresponding 8-bit from the S-box lookup table during the SubBytes transformation[4]. Using the inverse S-box operation during the decryption process, this step is reversible. The multiplicative inverse and affine transformation are two mathematical methods used in the design of the S-box in AES to guarantee its cryptographic features. The AES algorithm's S-box is a crucial part, and the encryption's security depends on how it is designed. The s-box involves 2 process namely: Multiplicative inverse and Affine transformation.

III. MULTIPLICATIVE INVERSE

The multiplicative inverse is utilised in the AES S-box operation to add non-linearity and complication to the substitution process. A mathematical concept called the multiplicative inverse is used to determine a number's inverse in a specific field. The 256 elements make up the Galois Field (GF) of 28, which is used to create the AES S-box. An 8-bit binary number can be used to represent each component of GF(28). In the S-box operation, an input byte in GF(28) is multiplicatively inversed, the resulting value is subjected to an affine transformation, and the result is subjected to a fixed permutation to yield the output byte. The Extended Euclidean Algorithm is used to calculate the multiplicative inverse of a given input byte in GF(28)[1]. The greatest common divisor of x and y is determined by this algorithm by finding two integers, a and b, such that a * x + b * y = gcd(x, y). Since the gcd in GF(28) is always 1, a * x + b * y = 1. The multiplicative inverse of x is then the value of a. An attacker would find it more challenging to decipher the encryption because of the multiplicative inverse's use in the AES S-box operation, which contributes to the replacement process's non-linearity and confusion. The S-box process is even more securely improved by the affine transformation and permutation phases[2]. As part of the AES S-box operation, the outcome of the multiplicative inverse computation is subjected to an affine transformation in order to increase the nonlinearity and confusion of the substitution process of x.



δ	Isomorphic mapping to composite fields
x^2	Squarer in GF(2^4)
x^{-1}	Multiplication inversion in GF(2^4)
δ^{-1}	Inverse isomorphic mapping to GF(2^8)
$x\lambda$	Multiplication with constant, in GF(2^4)
$\oplus$	Addition operation in GF(2^4)
$\times$	Multiplication operation in GF(2^4)

Fig.2 Block Diagram of Multiplicative Inverse

IV. AFFINE TRANSFORMATION

A linear transformation and the addition of a constant value make up the two steps of the affine transformation. The input byte is multiplied with a fixed 8x8 binary matrix over GF(2), referred to as the affine matrix, to produce the linear translation. The outcome of this multiplication is then XORed with the affine vector, a fixed 8-bit binary vector. The affine matrix and vector are carefully selected to guarantee that the output of the S-box operation has a high level of non-linearity and is resistant to attacks like linear and differential cryptanalysis. The precise selection of the affine matrix and vector is based on mathematical analysis and design concepts. In the AES S-box procedure, the bits in the output byte are fixedly shuffled after the affine transformation step. The S-box operation's confusion and diffusion features are further improved by this permutation step, making it more challenging for an attacker to ascertain the relationship between the input and output bytes. The AES S-box operation is a crucial phase in the AES encryption algorithm since it offers strong cryptographic features overall thanks to the multiplicative inverse, affine transformation, and permutation processes.

V. LINEAR SHIFT FEEDBACK REGISTER

The input bit of a shift register known as a seed is the result of a linear function of at least two of the states (or taps) it previously had. A shift register is a kind of sequential logic circuit used in digital circuits that is primarily for the storage of digital data[3]. It is linearly configured and contains inputs and outputs that are connected in such a way that the data is moved down the line when the circuit is activated. When a shift register's output bit is the result of a linear function of two or more of its previous states (taps), the shift register is said to be operating as a linear feedback shift register (LFSR). Each stage of an LFSR of length m may store one bit, and there is a clock that regulates data interchange between the stages. The stages are numbered 0, 1,..., m1. The shift register would be initialised with a vector containing the entries s0,...,sm1.

The actions listed below are carried out at time i.

1. s_i(the content of stage 0) forms part of the output;
2. The content of stage *i* is shifted to stage i−1, for 1≤i≤m−1;
3. The new content (the *feedback bit*) of the stage m−1 would be obtained by xor- ing a subset of the content of the *m* stages.

An LFSR's initial input is referred to as a seed. Any register must finally be periodic since each register has a limited amount of different states. However, a well-chosen feedback function and seed can enable an LFSR to generate a large-period sequence of bits that appears random (and has strong statistical features). In both software and hardware, LFSRs can be used to generate pseudo-random numbers, pseudo-noise sequences, rapid digital counters, whitening sequences, cryptography, etc.

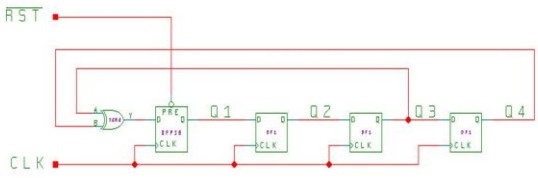


Fig.3.Block Diagram of 4-bit LFSR

VI. REVERSIBLE LOGIC GATES

Reversible logic is one of the options available in the EDA (Electronic Design Automation) industry to meet the power, speed, and space requirements, as these circuits have been theoretically proven to provide near-zero energy computation using Information Loss Prevention during operation.. Reversible logic is a computing paradigm in which logic gates are designed to be reversible, meaning that they can perform operations in both forward and reverse directions without losing information. In traditional computing, Despite of, irreversible operations such as deleting data cannot be undone, results in data loss. Reversible logic gates, on the other hand, are designed to have the property that every input has a unique output and vice versa[5]. This makes it possible to reverse the operation and recover the original input, without any data loss and it is also being studied for its potential in reducing power consumption in classical computing, which is a major concern due to the increasing energy costs and environmental impact of computing technology. In cryptography, an S-box (substitution box) is a component of a symmetric-key cipher that performs a substitution operation on a block of data. Reversible logic can be used in the design of S-boxes to improve their security and efficiency. To designing reversible S-boxes is to use quantum circuits, which can implement reversible gates such as the Toffoli gate and the Feynman gate. By using reversible gates, the S-box can be designed to have a one-to-one mapping between its input and output. We are using the Feynman and Toffoli gates and the quantum cost is relatively low.

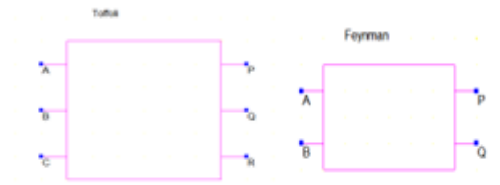


Fig.4. Toffoli & Feynman logic gates

The quantum cost of a logic gate is a measure of the resources (such as the number of quantum gates and the number of qubits) required to implement the gate in a quantum circuit. The Feynman gate is known as the quantum controlled-NOT (CNOT) gate, has a quantum cost of 1. This means that it can be implemented using a single CNOT gate, which requires two qubits. And the Toffoli gate is also known as the quantum CCNOT gate, has a quantum cost of 5. This means that it can be implemented using five CNOT gates and three qubits. It's worth noting that the quantum cost of a gate is not the only factor that determines its usefulness in a particular application. Other factors, such as the gate's error rate, its compatibility with other gates in a quantum circuit, and its ability to perform specific quantum operations, may also be important considerations.

VII. RESULTS & DISCUSSION

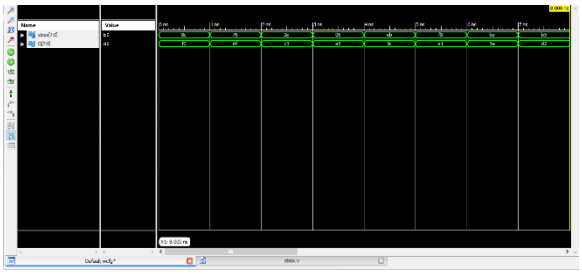


Fig.5. Timing Diagram of S-box without LFSR

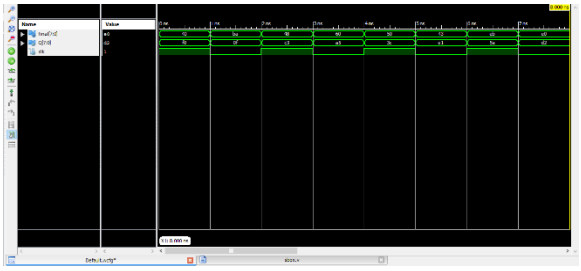


Fig.6. Timing Diagram of S-box with LFSR

In the fig.5, the variable ‘Q’ is the 8-bit input plain text and the variable ‘s-box’ is the 8-bit output cipher text. The output shown in the fig.5 are the values of existing static s-box. In the fig.6, the variable ‘Q’ is the input plain text and the variable ‘final’ is the 8-bit output cipher text. The output shown in the fig.6 are distinct in nature because of the use of LFSR. In this way, we are improving the security of the s-box. The below two figures are the RTL schematic of the s-box using irreversible and reversible logic gates respectively.

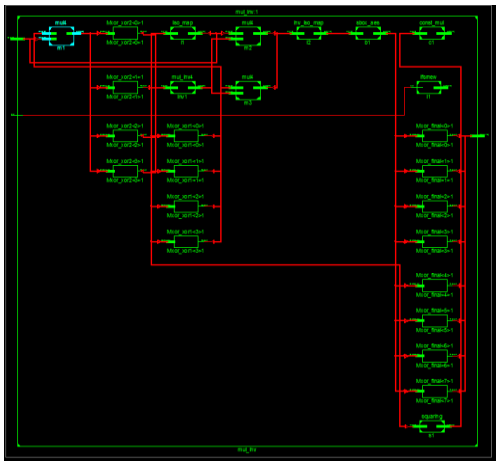


Fig.7. RTL schematic of s-box using irreversible logic gates

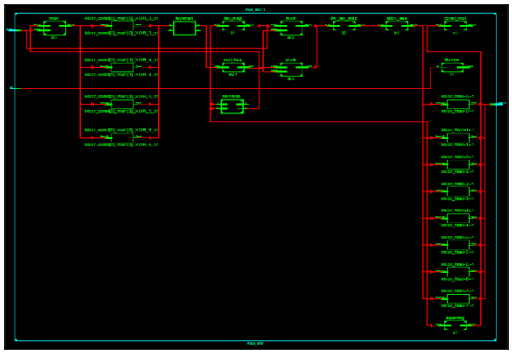


Fig.8. RTL schematic of s-box using reversible logic gates

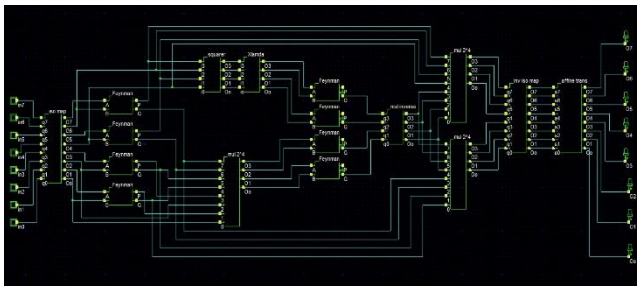


Fig.9. Architecture of proposed S-Box

VIII. CONCLUSION

This paper presents a new approach to generate a dynamic AES with key-dependent S-boxes. It was established that for any change of the secret key, the structure of the S-box will be changed essentially.The power analysis attack such as side channel attacksis decreased because of the use of reversible gates which theoretically consumes zero power, they are exploited to construct the AES algorithm in this work. The Toffoli & Feynman family of reversible gates are used in the proposed designs and the garbage values of the reversible logic gates are optimized as much as possible in order to increase the performance metrics in the proposed structures.

REFERENCES

[1] A. Barrera, C. -W. Cheng and S. Kumar, "A Fast Implementation of the Rijndael Substitution Box for Cryptographic AES," 3rd International Conference on Data Intelligence and Security (ICDIS), South Padre Island, TX, USA, pp. 20-25, 2020.

[2] W. I. El Sobky, A. A. Isamail, A. S. Mohra and A. M. Hassan, "Implementation Mini (Advanced Encryption Standard) by Substitution Box in Galois Field (24),“ International Telecommunications Conference (ITC-Egypt), Alexandria, Egypt, pp. 1-4, 2021.

[3] D. Lee and Y. Kim, "Design of a Light-Weight Key Scheduler for AES using LFSR for IoT Applications," IEEE International Conference on Consumer Electronics-Asia (ICCE-Asia), Gangwon, Korea, Republic of, pp. 1-2, 2021.

[4] D.K. Sushma and Dr. Manju Devi, “Design of S-box and INV S-box using Composite Field Arithmetic for AES Algorithm”, International Journal of Engineering Research & Technology, Vol 6, pp. 1-2, 2018.

[5] Bahram Rashidi and Bahman Rashidi, “Implementation of An Optimized and Pipelined Combinational Logic Rijndael S-Box on FPGA”, Computer Network And Information Security, pp. 41-48, 2018.

[6] Arash Reyhani-Masoleh, Senior Member, Mostafa Taha, Member IEEE, and Doaa Ashmawy, “ New Low-Area Designs for the AES forward, Inverse and Combined S-Boxes”, IEEE Transactions on Computers, Vol 69, No. 12, 2020.

[7] National Institute of Standard and Technology (NIST). (2001). Advanced Encryption Standard (AES),FIPS-197.

[8] J. J. Tay, M. L. Dennis, M. M. Wong, C. Zhang, and I. Hijazin,
“Construction of a low multiplicative complexity GF(24) inversion circuit for compact AES S-box,” in Proc. IEEE Region 10 Conf. TEN-CON, Oct. 2018, pp. 0540–0544.

[9] A. Reyhani-Masoleh, M. Taha, and D. Ashmawy, “New area record for the aes combined s-box/inverse s-box,” in Proc. IEEE 25th Symp. Comput. Arithmetic, 2018, pp. 145–152.