Provably Secure Encryption Algorithm based on Feistel Structure

Ahmed M. Rayan
Master Student, Elec. & Comm. Dep., Faculty of Engineering, ASU, Egypt

Ahmed A. Abdel-Hafez
Communications Department, Military Technical Collage, Egypt

Ismail Mohamed Hafez
Elec. & Comm. Department, Faculty of Engineering, ASU, Egypt

ABSTRACT

In 1997 The National Institute of Standards and Technology (NIST) started a process to select a symmetric-key encryption algorithm instead of DES. NIST determined the evaluation criteria that would be used to compare the candidate algorithms depending on the analyses and comments received, NIST selected five finalist algorithms (RC6, MARS, Rijndael, Serpent and Twofish). At the end, NIST selected Rijndael as the proposed Advanced Encryption Standard algorithm (AES). Although Twofish algorithm based on Feistel structure and possesses a large security margin, it has some drawbacks as The Twofish structure is not easy to analyses, the mixing of various operations makes it hard to give a clean analysis and forces us to use approximation techniques. Moreover, The use of key-dependent S-Boxes adds complexity and greatly increase the effort required to write automated tools to search for characteristics (differentials, linear, …) of the structure. In this paper a proposal of a new Secure Symmetric-key Encryption (SSE) algorithm based on Feistel structure is produced to overcome the previous drawbacks and produce a provable secure algorithm.

Keywords

Symmetric-key cryptography; Block Ciphers; Substitution-Box; Diffusive Components; MDS; branch number.

1. INTRODUCTION

Symmetric-key cryptography is the most prominent and an important element in many cryptographic systems comes in two flavors, stream ciphers and block ciphers. The later type of Symmetric-key cryptography is more powerful object, as it can be used in more ways, to encrypt and authenticate, provide integrity, protection and confidentiality. It provides high diffusion (information from one plain text symbol is diffused into several cipher text symbols), and also can be employed in many modes (CTR –OFB - …) to gives stream cipher algorithm. Block cipher is easier to be implemented in software, as it avoids time consuming bit manipulations, operates on data in computer-sized block. This paper is organized as follows: Section 2 provides an overview of block cipher design principles. Section 3, briefly explain Twofish cryptographic algorithm. Section 4, describes the (SSE) algorithm. Section 5 proves the security of (SSE) algorithm. Finally, conclusion and future work will be in Section 6.

2. BLOCK CIPHERS DESIGN PRINCIPLES

Block ciphers are the most widely primitives for ensuring data confidentiality. Let n and k be two positive integers, a block cipher with block–size n and key–size k is a family of 2^k permutations $\{E_k : \{1,0\}^n$ to $\{1,0\}^n\}_{k\,\epsilon\{1,0\}^n}$ on bitstrings of length n. For implementation reasons, all classical block ciphers are composed of several round-permutations F_i (iterated ciphers) of finite field $GF(2^n)$, where each $F_i, 1 \leq i \leq r$, is parametrized by a secret quantity k_i named the round key, which is derived from the master key K as shown in figure 1.

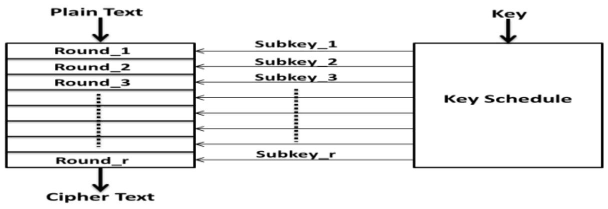

Fig 1: Block Cipher

Parameter r is the number of rounds in the cipher. In each iterated cipher, the r round-permutations F_i are chosen to be very similar for two reasons. First, the implementation cost of the iterated cipher in hardware. Moreover, the type of design provides some simple security arguments. However, the rounds should be slightly different in order to resist some structural attacks such as slide attacks [1].This difference may be introduced by key schedule(i.e. identical round permutations with different rounds-keys), or the round permutation may be slightly different. The main basic constructions for the round permutation: substitution-permutation network (SPN) as (Rijndael, Square,…), Lai-Massey scheme as (Proposed Encryption Standard (PES), …) and Feistel network as (Data Encryption Standard (DES), Twofish, ….). The later type as shown in figure 2 relies on an inner function F_k operating on the half of the block size.

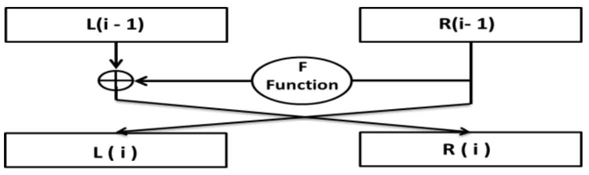

Fig 2: Feistel network

This structure used in many encryption algorithms and presents several advantages as the encryption and decryption operations hardly need separate implementations, and consequently low implementations cost, it has been widely studied from the theoretical point of view [2]. Some of the Feistel structure algorithms reversible with changes only in the key schedule (e.g., DES, and Blowfish), while others uses round functions slightly different, but are built from the same blocks as Twofish algorithm. The design principles for the round permutation follow the principles introduced by Shannon [3]:

(1) Confusion: means making "*the relation between the simple statics of the cipher text and the simple decryption of the key is very complex and involved one*". This implies for instance that any algebraic

relation between these quantities must have a high degree and a large number of terms.

(2) Diffusion: means "*dissipating the static structure of the plaintext into long range statistics*". This implies that all plaintext bits and key bits must influence all ciphertext bits. Then, the key idea behind the Feistel structure is to decompose the round function into two distinct steps: a nonlinear substitution function for providing confusion called Substitution- box, and linear permutation for providing diffusion.

2.1 Substitution- box

The strength of most block ciphers (more specifically their resistance against linear and differential cryptanalysis) is inevitably tied to the strength of their S-Boxes, which is usually their sole non-linear component. An n -bit to m -bit S-Box defines simply a substitution, i.e. to each n bits input is mapped a corresponding m bits output value (which has not necessarily to be the same length as the input). S-Boxes are responsible for bringing confusion in the data processing. This means that they should hide any mathematical relationship between the plaintext, the ciphertext and the key [2]. It is possible to identify three different strategies to build S-Boxes:

a. (1) Random choice: choose the contents completely at random. A way to choose random S-Boxes is to make them key- dependent, there are at least two disadvantages, which can be traded off against each other. One is that generating the S-Boxes has a cost. The other is that the generated S-Boxes are not optimized and may even be weak. On the other hand, generating cryptographically strong S-Boxes at run time are impractical [4].

b. (2) Random choice followed by filtering: generate random ones and to check if they have the desired properties until a good one is found but it is a very heavy and computation-intensive process [2].

c. (3) Algebraic constructions: using algebraic methods to offer good non-linearity properties:

d. *Mixing non-isomorphic operations (XOR and addition modulo 2^{32} for 32 -bit vectors, for instance).

- Using algebraic operations known as Mixing of addition in $\mathrm{GF}(2^n)$and in Z_n or Power function in $\mathrm{GF}(2^n)$.
- Combination of an inverse function $\mathrm{X} \rightarrow \mathrm{X}^{\mathrm{e}}$in $\mathrm{GF}(2^{\mathrm{n}})$ and an affine transformation over some other incompatible algebraic structure.
- Combination of a power function $\mathrm{X} \rightarrow \mathrm{X}^{\mathrm{e}}$ in $\mathrm{GF}(2^n)$ and an affine transformation over Z_n.

On the other hand, these constructions are helpful to Courtois-Pieprzyk algebraic attacks [2].

In his introduction to the Biham and Biryukov work on DES with permuted S-Boxes, Schneider summarizes the usefulness of randomly-generated S-Boxes [5], "*Linear and differential cryptanalysis work only if the analyst knows the composition of the S-Boxes. If the S-Boxes are key-dependent and chosen by a cryptographically strong method, then linear and differential cryptanalysis are much more difficult. Remember, though, that randomly-generated S-Boxes have very poor differential and linear characteristics, even if they are secret*". To overcome the drawbacks of the previous strategies, by using strong Algebraic S-Boxes that are key-dependent but are not randomly generated to get the benefits of strong S-Boxes and also of key-dependent S-Boxes by applying the operations before encryption begins and use the modified S-Boxes for the actual encryption, so the overhead is exclusively in the set-up phase. There is no increase in the per-block encryption cost. There are several classes of operations may be used [4].

1. Permuting S-Box columns: It can be achieved by permuting each row in a key-dependent way.
2. Adding affine functions to S-Box columns: The addition of affine functions can be done by XORing a constant into all rows, this constant may be the XORing of all bytes of the round subkey.
3. Permuting S-Box Inputs: Rearranging the order of the S-Box rows in a key-dependent way.
4. Adding Affine Functions to S-Box Inputs: Adding selected affine functions to S-Box inputs by XORing a constant binary vector into the input and then use the output as an input to the S-Boxes.

2.2 Diffusive components

The purpose of a diffusive construction is to provide an avalanche effect, both in the context of differential and linear approximations. In the linear context, this means that there should be no correlations between linear combinations of a small set of inputs and linear combinations of a small set of outputs. In the differential context, small input changes should cause large output changes, and conversely [6]. Maximum Distance Separable matrix (MDS) is a very popular tool to achieve diffusion. The concept of MDS is taken from linear coding that is defined by three variables:

1. n: The length of codeword (the sum of number of bytes input and output).
2. k: The dimension of the codeword (Dimension of matrix).
3. d: The minimum number of positions. In which any 2 codewords differ (number of positions where the two codewords differ).

A linear code is called MDS if $d = \mathrm{n} - \mathrm{k} + 1$. In simple words, if two inputs are applied with particular number of difference to MDS matrix, then at least a certain number of differences in the output are got. If the total number of differences in input bytes are denoted as Δ_{in} and total number of differences in output as Δ_{out} , then for MDS matrix $\Delta_{in} + \Delta_{out} \geq B$, where B is called branch number which gives a tighter bounds for the security of the cipher (the number of outputs that will change if one byte of input for single round is changed). For the matrix used in Twofish the branch number is 5, as it has input of length 4 bytes and output of 4 bytes (length of codeword n is 8 bytes) and dimension of matrix k is 4, so $d = 8 - 4 + 1 = 5$.

2.3 Efficient MDS matrix generation

The square matrix A is an MDS matrix if every square submatrices of A are nonsingular or matrix A is a full rank matrix with the inverse matrix having all entries non zero and all of its 2×2 submatrices are full rank. There are many strategies to generate MDS matrix:

(1) Use of circling MDS matrix: $M_{k*k}(GF(P))$ matrix is said to be circling and is noted $C(\alpha_1,..,\alpha_k)$ if it is of the form:

$$\begin{pmatrix} \alpha_1 & \alpha_2 & \ldots\ldots & \alpha_k \\ \alpha_k & \alpha_1 & \ldots\ldots & \alpha_{k-1} \\ . & . & \ldots\ldots & . \\ \alpha_2 & \alpha_3 & \ldots\ldots & \alpha_1 \end{pmatrix}$$

Where $\alpha_1,...,\ \alpha_k \in GF(q)$.With this construction, the number of distinct coefficients will be minimized to optimize the number of precomputed tables, and maximize the number of "1" coefficients.

(2) Use of Hadamard Matrix Second: Hadamard Matrices are matrices of the form

$$\begin{pmatrix} H1 & H2 \\ H2 & H1 \end{pmatrix}$$

Where H1 and H2 are Hadamard matrices. An interesting fact is that Hadamard matrices are entirely defined by their first line. Therefore only k distinct coefficients are necessary. Another property is that $H \times H = C^2 . I$, where C is the sum of element of the first row. By setting C to 1, $H = H^{-1}$: H is involuntary. This reduces the number of coefficients for ciphering, deciphering to k instead of 2k. The matrix of the cipher ANUBIS [7] is such an example.

$$\begin{pmatrix} 1 & \alpha & \alpha_2 & \alpha+\alpha_2 \\ \alpha & 1 & \alpha+\alpha_2 & \alpha_2 \\ \alpha_2 & \alpha+\alpha_2 & 1 & \alpha \\ \alpha+\alpha_2 & \alpha_2 & \alpha & 1 \end{pmatrix}$$

(3) Use of Algebraic method: It is possible to generate matrices that are MDS by construction using some code theory. A Reed-Solomon code has a generating matrix of the form:

$$G(\alpha) = \begin{pmatrix} 1 & 1 & 1 & \ldots\ldots & 1 \\ 1 & \alpha & \alpha^2 & \ldots\ldots & \alpha^{(n-1)} \\ 1 & \alpha^2 & \alpha^4 & \ldots\ldots & \alpha^{(n-1)^2} \\ . & . & . & \ldots\ldots & . \\ 1 & \alpha^{(k-1)} & \alpha^{2(k-1)} & \ldots\ldots & \alpha^{(n-1)(k-1)} \end{pmatrix}$$

And any sub matrix $k \times k$ of $G(\alpha)$ is MDS matrix.

3. TWOFISH CRYPTOGRAPHIC ALGORITHM

Twofish as shown in figure 3 is a 128-bit block cipher that accepts a variable length key. It is a 16-round Feistel network with additional whitening of the input and output. Its encryption and decryption round functions are slightly different, but are built from the same blocks. That is, it is simple to build a hardware or software module that perform both encryption and decryption without duplicating much functionality, but the same module cannot perform both encrypt and decrypt [8].

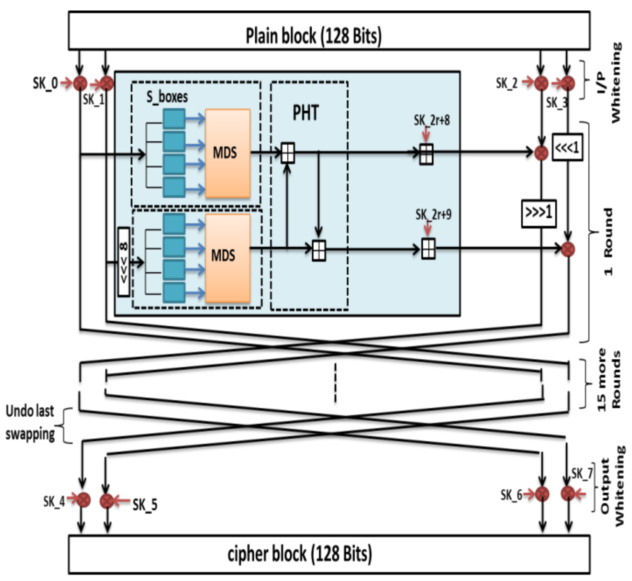

Fig 3: Twofish Algorithm

3.1 Input and output whitening

128 bits plaintext is divided into four words of each 32 bits. Each word passes through input whitening process that is XOR four units of 32 bits subkey and 128 bits plaintext.

3.2 F- function

Two words of the left side are used as inputs of two g-function inside the F- function in each round. One input word passes through 8 bits left circulation. A g- function is composed of MDS matrix multiplier and 4 S-Box. Outputs of two g- function combined to use PHT (Pseudo-Hadamard Transform), and two subkeys are added by modulo-2 addition.

3.3 Swapping

Two outputs of the F function exchange a position for the following round. The results of the last round exchange a position again after 16th Round and then passes through output whitening to create 128 bits ciphertext.

3.4 Key schedule

Twofish is defined for keys of length N = 128, N = 192, and N = 256.The global key in our case is 128. The key schedule uses the same primitives as the round function, and provides two sets of subkeys:

1) K Subkeys: 40 words of expanded key K_0 .. K_{39}

2) word of keys in each round (2* 16=32 words), 4 words for input whitening and 4 word of output whitening, the total is 32 +8 = 40 words). The generation of K set is done by dividing the master key (4 words) into two sets M_{even} (2 word) and M_{odd} (2word), then Apply M_0,M_1 , M_2 , M_3 to function h as shown in figure 4 to generate K_0, .., K_{39} [8].

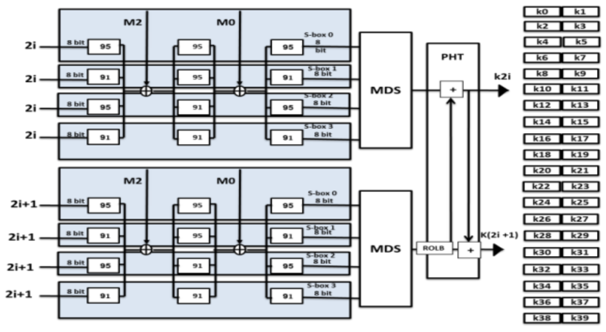
M2
M0
2i
2i+1
8 bit
S-box 0
S-box 1
S-box 2
S-box 3
MDS
PHT
ROL8
k2i
K(2i +1)
k0 k1 k2 k3 k4 k5 k6 k7 k8 k9 k10 k11 k12 k13 k14 k15 k16 k17 k18 k19 k20 k21 k22 k23 k24 k25 k26 k27 k28 k29 k30 k31 k32 k33 k34 k35 k36 k37 k38 k39

Fig 4: K subkeys generation

(2) S Subkeys: It contains S_0, S_1 that is used in S-Box, and they are fixed during the entire encryption and decryption process. The generation of S subkeys is done by taking the key bytes in groups of 8, interpreting them as a vector over $GF(2^8)$ with the primitive polynomial $x^8 + x^6 + x^5 + x^3 + x^2 + 1$, and multiplying them by a 4×8 matrix derived from an RS code.

4. (SSE) ALGORITHM

As mentioned before, Although Twofish algorithm possesses a large security margin, it has some drawbacks as the analysis of its Key-dependent S-Boxes is complicated and the overall complexity of design has drawn some concern [9]. Moreover, it had the following observations [10]:

1) It is not clear whether key-dependent S-Boxes used in Twofish algorithm necessarily offer any additional security over strong fixed S-Boxes. The flexibility of key-dependent S-Boxes can actually an advantage to the attacker." *Instead of choosing the characteristic to fit the S-Box, we choose the S-Box to fit the characteristic*".

2) The designers did not produce any significant reason for adding fixed rotations by one bit position in the algorithm except "*They believe that the one-bit rotations make cryptanalysis harder, if they have any effect at all*". But fixed rotations can be used by the cryptanalyst to reduce the number of active S-Boxes in a characteristic. Maybe this is the reason that one of the Twofish designers said "*We have no reason to believe that the 1-bit rotations make Twofish stronger against differential attack*" [11].

3) The fixed rotation by eight bits is intended to lead to conflicts that the cryptanalyst will find that is hard to resolve. However the use of S-Boxes that is changed with the key means that there may be some keys that will resolve any potential conflict.

The proposal symmetric-key encryption algorithm (SSE) as shown in figure 5 uses a 16-round reversible Feistel structure with additional whitening of the input and output is presented. It overcomes these drawbacks and observations of Twofish algorithm by providing a proven security for each component, besides construct a new key schedule that is fast and secure.

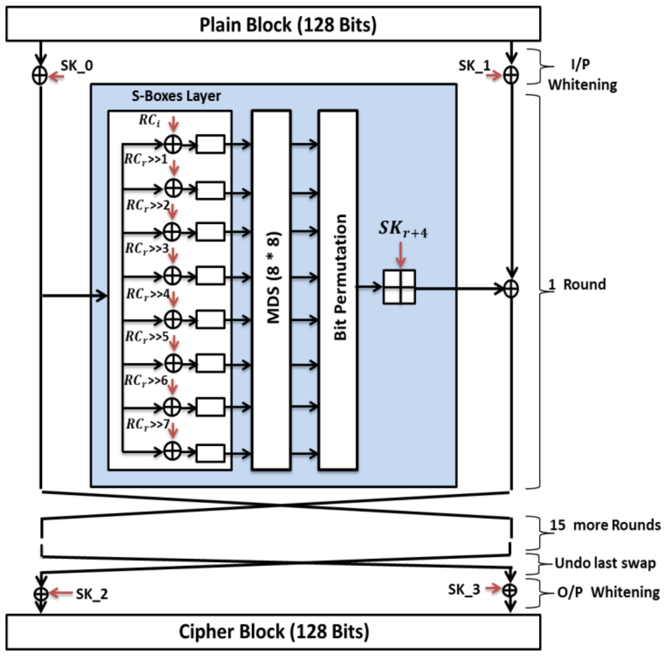

Fig 5: (SSE) Algorithm

4.1 S-Boxes layer

Our aim is to build strong Key dependent S-Boxes layer to overcome the drawbacks (differential cryptanalysis – linear cryptanalysis) of fixed S-Boxes. The building of this S-Box layer will be done in two steps:

1) Building offline a random balanced vectorial function over $GF(2^8)$ with the primitive polynomial $x^8 + x^4 + x^3 + x^2 + 1$ bits that is satisfies good linear properties (nonlinearity - algebraic degree - immunity order - ...) and differential properties (propagation criteria – max. autocorrelation - ...) Compared to S-Boxes of the AES algorithm.

2) Conceal the input of the S-box by adding Affine Function to S-Box Inputs (XORing the round constant into all S-Box input) as described by Algorithm 1. 16 different round constants given from the key schedule will achieve a different S-Box for each round, and the right circular shift for each constant inside the round function resulting 8 different S-Boxes output in each round as shown in figure 6.

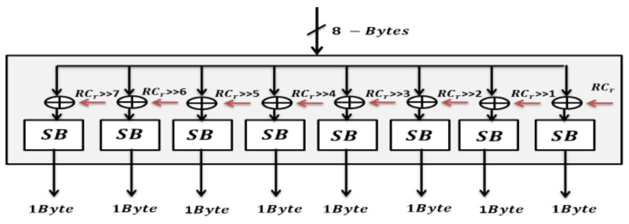

Fig 6: S-Boxes layer

So, the output of the S-Boxes Layer is dependent on the key constant driven from key schedule algorithm, and this output will be different even if the input for S-Box is the same.

Algorithm 1: Key dependent S-Boxes

```
// Input:
// 8 byte input to Key dependent S-Boxes layer
// 16 of 1 byte round constant $RC_0 \dots RC_{15}$
for all r from 0 to 15 do
    for all j from 0 to 8 do
        S-box_1[r][ j] ⟵ S-box[i][j ⊕ (RC[r]>>> j)]
    end for
end for
// Output:    // 8 byte output Key dependent S-Boxes layer
```

4.2 Optimal MDS

Our aim was to build MDS matrix with high branch number, by implementing an algorithm that generates a random MDS matrix M ϵ M_{k*k} (GF (q)). For efficiency the number of distinct coefficients is minimized to optimize the number of precomputed tables, and maximize the number of « 1 » coefficients leading to a simple and efficient implementation in software. Circling MDS matrix strategy is used. To fit the condition of minimizing the number of distinct coefficient, efficient circling matrix C(α) with $\alpha_{2^i} = 1$ for all i with $0 \le i \le [\log_2 (k+1)] - 1$, (i.e. $1 \le 2^i \le [k/2]$) is generated.

$$C(\alpha) = \begin{pmatrix} 01 & 01 & 04 & 01 & 08 & 05 & 02 & 09 \\ 09 & 01 & 01 & 04 & 01 & 08 & 05 & 02 \\ 02 & 09 & 01 & 01 & 04 & 01 & 08 & 05 \\ 05 & 02 & 09 & 01 & 01 & 04 & 01 & 08 \\ 08 & 05 & 02 & 09 & 01 & 01 & 04 & 01 \\ 01 & 08 & 05 & 02 & 09 & 01 & 01 & 04 \\ 04 & 01 & 08 & 05 & 02 & 09 & 01 & 01 \\ 01 & 04 & 01 & 08 & 05 & 02 & 09 & 01 \end{pmatrix}$$

4.3 Bit permutation

A regular bit-permutation is used. This bit-permutation shown in figure 7 can be written in the following way:

$$P(i) = \begin{cases} i * 8 \bmod 63 & i \in \{0,\dots,62\}. \\ 63 & i = 63. \end{cases}$$

This bit-permutation described by Algorithm 2 will satisfy three important features to the algorithm:

(1) If there is change in only one byte, this will satisfy that all output bytes (8 bytes) will change and the number of active S-Boxes will increase.

(2) Helps to make a clear security analysis for the algorithm.

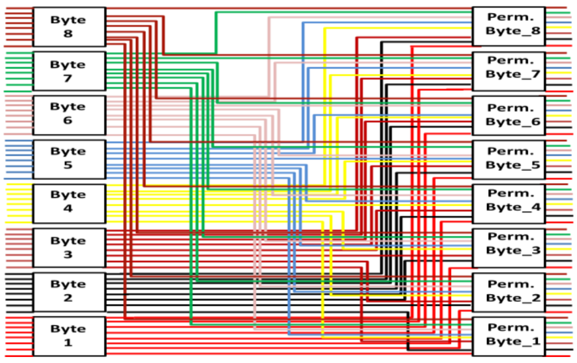

Fig 7: Bit Permutation

Algorithm 2: bit-permutation

```
// Input:    Eight bytes  $P\_I_0, \dots, P\_I_7$
Bit-Permutation (P_O[i][ j] , P_I[j][i])
{
    for all i from 0 to 7 do
        for all j from 0 to 7 do
            P_O[i][ j]  ⟵  P_I[j][i]
        end for
    end for
}
// Output:    Eight bytes  $P\_O_0, \dots, P\_O_7$
```

4.4 Key schedule

As any cipher can be broken due to a bad key-schedule design, so many concepts are achieved in our key-schedule design:

(1) Reuse the Same Primitives that is used in the encryption algorithm (S-Boxes – MDS - …).
(2) The design is secure and simple for analysis.
(3) Using constant (IV) to avoid related subkey attack.
(4) Using Key dependent S-Boxes layer to avoid linear and differential cryptanalysis.

(5) Change in only one bit of master key gives influence to all subkeys.

(6) No weak keys.

The key schedule shown in figure 8 and described by Algorithm 3 with 128 bits global key, provided 40 subkey of expanded key SK_0, .., SK_{39} , and 16 round constant (1 byte) RC for each round to Xored to the input of S-Boxes.

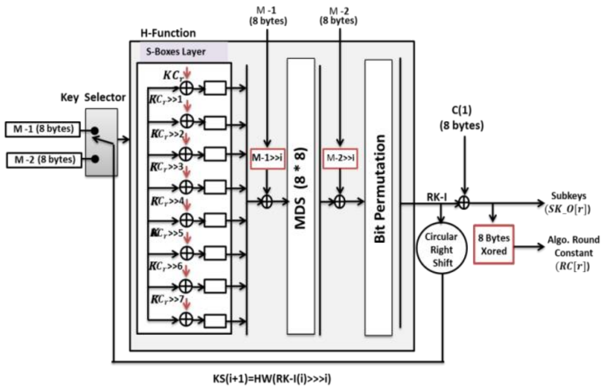

Fig 8: key schedule

Algorithm 3: key schedule Algorithm

```
// Input:
// Master Key 16 bytes $M_0 .. M_{15}$ , Constant IV ($C[0] = 8$ bytes) .
//Start Algorithm
// Divide the master key into two sets M_1, M_2
for all $i$ from 0 to 7 do
    $M\_1[i] \leftarrow M[i]$
    $M\_2[i] \leftarrow M[i+8]$
end for
// Start 20 iteration
for all $r$ from 0 to 19 do
    $Z$ = hamming weight ($C[r]$)
    If $Z$ is odd then
        // use M_1 as IP and ($KC_r$= Xored M_2 bytes as constant to xored with the input of S-Boxes)
        for all $i$ from 0 to 7 do
            $KC_r(i) \leftarrow KC_r(i) >>> i$
        end for
        for all $i$ from 0 to 7 do
            $Sbox\_in[i] \leftarrow M\_1[i] \oplus KC_r(i)$
        end for
        //Apply to S-Boxes
        for all $i$ from 0 to 7 do
            $S\_O[i] \leftarrow$ (S-Box[Sbox_in[$i$] ]) $\oplus$ ($M\_1[i] >>> i$)
        end for
        // Multiply by MDS matrix
        for all $i$ from 0 to 7 do
            $M\_O[i] \leftarrow$ (MDS[$i$][$j$]*$S\_O[i]$) $\oplus$ ( $M\_2$[i]>>> $i$)
        end for
        //Apply to Bit-Permutation
        Bit - Permutation ($SK\_I$,$M\_O$)
        // Generate the output subkeys ($SK\_O$)
        $SK\_O[r] \leftarrow SK\_I \oplus (C[r])$
        // Generate the algorithm rounds constant (RC)
        If $i > 3$ then
            for all $j$ from 0 to 7 do
                $RC \leftarrow Xored(SK\_O[j])$
            end for
        end if
        // Build the new constant (Circular shift of 64bits subkey)
        for all $i$ from 0 to 63 do
            $C[r+1] \leftarrow (SK\_I[r] >>> i)$
        end for
    else Z is even then
        Use $M\_1[i]$ as IP and ($RC_r$= Xored M_1 bytes as constant to xored with the input of S-Boxes).
    end if
end for
// End Algorithm
// Output:
// 20 Subkeys of 8 bytes $SK_0, ..., SK_{19}$
// 16 Round constants RC (1 byte)
```

5. (SSE) ALGORITHM

5.1 Brute force attack

As the key length of (SSE) algorithm is 128-bit, the key has a complexity of (2^{128}).To be attacked by Brute Force Attack. Let's used super computer (Tianhe-2, a supercomputer developed by China's National University of Defense Technology, has retained its position as the world's No. 1 system, according to the 45th edition of the twice-yearly TOP500 list of the world's most powerful supercomputers. with a performance of 33.86 petaflop/s (quadrillions of calculations per second) [12]) do 33.86 * (10^{15}) decryptions per second (33.86 * (10^{15}) quadrillions of calculations per second). That is mean (10.41 * (10^{23}) decryptions per year for one machine. About (2^{127}) decryptions on average are needed, so you would need $\frac{(2^{127})}{(10.41)*(10^{23})}$ = 16.34* (10^{13}) years. Or, need $\frac{(2^{127})}{(10.41)*(10^{23})}$ = 16.34 * (10^{13}) computers / year and this is costly and infeasible.

5.2 Linear and differential cryptanalysis

There are two approaches used to ensure the resistance of any block cipher to linear and differential cryptanalysis, either using key-dependent S-Boxes, or increase the number of active S-Boxes.

In order to achieve the first approach, S-Box layer started with carefully-prepared 8 different strong static S-Boxes with good linear and differential properties compared with AES and key-dependent operations are applied to the input of S-Boxes before using it to achieve best differential and linear characteristics. The goal was to introduce additional entropy so that attacks which depend on knowledge of the S-Boxes become impractical, without changing the properties which make the S-Boxes strong. Affine function is added to the input of S-Boxes (XORing a round constant to the input) before encryption begins and use the modified S-Boxes layer for the actual encryption. The addition of affine functions does nothing to degrade cryptographic security in the S-Boxes layer. However, such an operation, can make it significantly more difficult to construct characteristics in a differential cryptanalysis attack (because it cannot be computed in advance when the XOR of two given S-Box outputs will produce one value or another). Hence, this operation increases the security of the cipher by raising the computational complexity of mounting this attack [4].

For the second approach, the diffusion layers of the (SSE) Algorithm are MDS and bit permutation. As the replacement of permutation layer in SPN with a diffusive linear transformation improves the avalanche characteristics of the block cipher which increases the cipher's resistance to differential and linear cryptanalysis [13-14]. Thus the main application of MDS matrix in cryptography is in designing block ciphers that provide security against differential and linear cryptanalysis [6].

If there is one active byte at the input to the F function, there must be at least 8 active bytes at its output. For the next round, one active S-Box will appear at the output of MDS (Piling-Up Lemma) [15], but bit permutation layer increase this number to 8 bytes. The minimal Number of Active S-Boxes for (SSE) algorithm satisfied the relation (number of active S-Boxes = 8r +1, where r = 0:15), and shown in table 1 compared to the minimal Number of Active S-Boxes for AES shown in table 2. The total number of active S-Boxes increases faster.

Table 1 : Number of active S-Boxes in (SSE) algorithm

Round	1	2	3	4	5	6	7	8	9	10
Min.	1	9	17	25	33	41	49	57	65	73

Table 2: Number of active S-Boxes in AES

Round	1	2	3	4	5	6	7	8	9	10
Min.	1	5	9	25	26	30	34	50	51	55

5.3 Higher order differential cryptanalysis
It is another version of differential cryptanalysis, looks at higher order relations between pairs of plaintext and cipher texts, and is applicable to algorithms which can be represented as Boolean polynomials of a low algebraic degree [16][17], or algorithms with a few rounds and poor short-term diff usion. In order to avoid this attack, (SSE) algorithm S-Box has been generated using the multiplicative inverse procedure similar to AES with a randomly chosen primitive polynomial defining a Galois field. The nonlinearity of this S-Box is 110 and its nonlinear degree is 7, strong diffusion functions (MDS, bit permutation) and the number of rounds equal 16.

5.4 Interpolation attack
This attack is efective against ciphers with a very small number of rounds, or against ciphers whose rounds functions have very low algebraic degree [18]. (SSE) algorithm has two reasons make this attack very unlikely to succeed. The first reason, S-Boxes have large algebraic degree = 7, moreover, the combination of operations from diff erent algebraic groups (including both addition mod 2^{32} and Xor operations) increase the degree. The second reason, its number of rounds is large = 16.

5.5 Related-key attack and slide attack
These attacks focus on the key scheduling. In the related-key attack, attacker obtains the encryption of certain plaintexts under several keys having certain relationships with each other to reveal the secret key. The basic related-key attack is "*chosen key attack*" based on the observation that in many block ciphers, the key scheduling algorithm can be viewed as a set of algorithms each of which "*extracts one particular subkey from the subkeys of previous rounds*". If all the algorithms of extracting the subkeys of the various rounds are the same then for a given key all the subkeys can be shifted one round backwards and get a new set of valid subkeys which can be derived from some other keys[19]. It depends on slow diff usion or symmetry in the key schedule. (SSE) key schedule algorithm uses functions that have maximum diff usion (MDS, bit permutation) and strong Key dependent S-Boxes layer. The generation of each round subkey is independent on the other round subkeys but depends only on the master key. Although the structure of the key schedule algorithm seems the same during generation of each round subkey, but it differs in many things:

(1) the contents of S-Boxes layer is not the same in each round but depends on a constant that is delivered from the xored bytes from the previous round subkey after circulated shift right (not from previous round subkey directly).

(2) In each round, the generation of its subkey depend on half of master key M_1 or M_2 (independently any other subkeys). The selection between the two half depend on the hamming weight of the byte resulted from xored bytes from the previous round subkey after circulated shift right (even or odd).

(3) The changing in any bit of master key influences SK_i and Round constant RC_i directly ,as The values that is xored with the outputs of S-Boxes and MDS is differ depending on the round number (M_1>>> r ,M_2>>> r).

Slide attack can be viewed as a particular case of related-key attack in which the relation is between the key and itself. If the function F is weak enough, it permits to retrieve the key k, so the structure of H-function of key schedule algorithm (strong Key dependent S-Boxes, MDS, bit permutation) makes this attack very unlikely to succeed.

5.6 Related subkey attack
The idea of this attack depends on finding a fixed diff erence (relationship) between expanded keys (subkeys), but not to original keys. It is applied to 11 round of version. 256-bit AES since its key schedule is close to linear and therefore the subkeys can be viewed as a codeword of a linear code.

This fixed difference between expanded keys (subkeys) in the (SSE) key schedule algorithm is infeasible as:

(1)The generation of each round subkey is independent on the others round subkeys but depend only on the master key.

(2) The structure of key schedule algorithm is slightly different for each round (S-Boxes layer is not the same - The values M_1>>> r and M_2>>> r that xored with the outputs of S-Boxes and MDS is dependent on round number).

(3) Using a different constant to be xored with the output subkey in each round.

The previous reasons make this attack very unlikely to succeed.

6. CONCLUSION AND FUTURE WORK
This paper has proposed a new secure symmetric-Key encryption (SSE) Algorithm based on the well-studied Feistel Structure to overcome the drawback and observations of Twofish algorithm. Each component in (SSE) Algorithm was studied carefully; a proven security for these components is produced. Cipher structure can be used on platforms with limited resources, the identical function be used for encryption and decryption with changes only in the key schedule. New key dependent S-Boxes layer was designed, fully dependent on the master key, which improves cipher quality when it comes to linear and differential cryptanalysis. New efficient MDS matrix and bit permutation to increase the number of active S-Boxes compared to AES algorithm are used. The key schedule is secure and simple for analysis, uses the same components that are used in (SSE) Algorithm, has been considerably simplified in comparison to that of Twofish Algorithm.

A new cryptanalysis attacks as algebraic attack and fast algebraic attack on (SSE) Algorithm to evaluate it will be our future work.

7. REFERENCES

[1] A. Biryukov, D. Wagner. "Slide Attacks," Fast software Encryption (FSE'99), volume 1636, lecture notes in computer science, pp.245-259, springer, 1999.

[2] P. JUNOD, Statistical Cryptanalysis of Block Ciphers (Lausanne, EPFL, 2005).

[3] C. E. Shannon, "Communication theory of secrecy systems," Bell System Technical Journal, vol. 28, pp. 656–715, Oct. 1949.

[4] S. Harris1, C. Adams2, "Key-Dependent S-Box Manipulations" Selected Areas in Cryptography (SAC '99) Proceedings, LNCS 1556, Springer, 1999.

[5] M. Matsui, R. Zuccherato," Selected Areas in Cryptography," 10th Annual International Workshop, SAC 2003, Ottawa, Canada, August 2003.

[6] K. Gupta, I. Ghosh Ray, "On Constructions of MDS Matrices from Companion Matrices for Lightweight Cryptography," CD -ARES 2013 Workshops, MoCrySEn, pp. 29-43, Springer 2013.

[7] P. S. L. M. Barreto and V. Rijmen, "The ANUBIS block cipher," 1st NESSIE Workshop, Heverlee, Belgium, Nov. 2000.

[8] B. Schneier, J. Kelsey, D. Whiting, D. Wagner, C. Hall, N. Ferguson, "Twofish: A 128-bit Block Cipher," AES Round 1 Technical Evaluation CD-1: Documentation, National Institute of Standards and Technology, Aug 1998.

[9] J. Nechvatal, E. Barker, D. Dodson, M. Dworkin, J. Foti and E. Roback," Status report on the first round of the development of the advanced encryption standard," Journal of Research of the NIST, vol. 104, no 5, Nechvatal et al., Sep-Oct, 1999.

[10] S. Murphy, M. Robshaw," Differential Cryptanalysis, Key- Dependent S-Boxes and Twofish," Codes and Cryptography, Vol. 27, pp. 229-255, 2002.

[11] B. Schneier, J. Kelsey, D. Whiting, D. Wagner, C. Hall, N. Ferguson," Twofish: A 128-bit Block Cipher," Counterpane Systems, USA, AES submission, 15 June, 1998.

[12] Top500 List - June 2015. http://www.top500.org/list/2015/06/

[13] H. M. Heys, S. E. Tavares, "The Design of Substitution-Permutation Networks Resistant to Differential and Linear Cryptanalysis," Proceedings of 2nd ACM Conference on Computer and Communications Security, Fairfax, Virginia, pp. 148–155, 1994.

[14] H. M. Heys, S. E. Tavares, "Avalanche Characteristics of Substitution - Permutation Encryption Networks," IEEE Trans. Comp., Vol. 44, pp. 1131-1139, Sept 1995.

[15]] M. Matsui, " Linear cryptanalysis method for DES cipher," in Advances in Cryptology -EUROCRYPT'93, Lecture Notes in Computer Science 765, Springer-Verlag, pp. 386–397, 1994.

[16] X. Lai, "Higher order derivatives and differential cryptanalysis," Communications and Cryptology, pp.227-233, Kluwer Academic Publishers, 1994.

[17] L.R. Knudsen, "Truncated and Higher Order Diff erentials," Fast Software Encryption, 2nd International Workshop Proceedings, pp. 196– 211, Springer- Verlag, 1995.

[18] T. Jakobsen and L.R. Knudsen, "The interpolation attack on block ciphers," Fast Software Encryption, LNCS 1267, pp. 28-40, Springer- Verlag, 1997.

[19] G. Piret, M. Ciet, J. Quisquater, "Related key and slide attacks: Analysis, connections, and improvements," Proceedings of the 23rd Symposium on IT in Benelux, pp. 315-325, 2002.