The Euphrates Cipher

Omar A. Dawood[1], Abdul Monem S. Rahma[2] and Abdul Mohssen J. Abdul Hossen [3]

[1] Assistant instructor, College of Education for Humanities Science English Department, Anbar University and Ph.D. student At University of Technology, Baghdad, Iraq

[2] Professor, Computer Science Department University of Technology, Baghdad, Iraq

[3]Assistant Professor, Computer Science Department, University of Technology Baghdad, Iraq

Abstract

In the present paper we give a complete description for the proposed new variant of AES cipher with high level of security and coherent algebraic aspects that is called the EUPHRATES cipher. The EUPHRATES name has been derived from one of the two famous rivers in Iraq. It is a revision of the TIGRIS cipher which we have already proposed before. It works with block size 128-bits and three variable key lengths 128-bits, 192-bits and 256-bits similar to the AES standard. The EUPHRATES cipher uses the Galois Field GF (2^8) as the mathematical representation and provides a good insight to the block cipher design.

Keywords*: Block Cipher, Advance Encryption Standard (AES), Substitution and Permutation Network (SPN), Maximum Distance Separable (MDS), Feistel Structure (FS).*

1. Introduction

Most of the modern block ciphers are constructed as a cascade of repeated keyed components, called rounds or (round functions). The security of such ciphers relies on applying the round function sufficiently many times [1]. Modern cryptography has successfully developed increasingly strong notions of security, providing secrecy in highly adversarial settings. Still, all these strong notions of security guarantee secrecy as long as the encrypted messages are confused and diffused with a high randomness [2]. Systems ciphers design are classified into two general categories Feistel and SPN that are the two main structures in block cipher design; these two structures do not refer to certain block cipher, but they represent as a description for the structure of block cipher. Feistel has a symmetric structure that uses the same structure for encryption and decryption such as Twofish, RC6 and MARS cipher, unlike SPN that uses asymmetric structure such as Serpent and Rijndael cipher [3]. This paper is organized as follows: Section 2: gives the outlines and brief description about the AES cipher. Section 3: indicates the preliminaries and the mathematical notations. The feature and the architecture of the EUPHRATES cipher are presented in Section 4, while the fundamental operations of the round transformation are introduced in Section 5. Security analysis and the implementation issues submitted in Section 6. The Conclusions are drawn in Section 7.

2. Outlines of Advance Encryption Standard (AES)

The Advanced Encryption Standard (AES) is a block cipher standard selected in 2000 after an open submission and evaluation process. The AES is the SPN Rijndael, designed by Joan Daemen and Vincent Rijmen two scientist from Belgian. The key length and the block length are variable and they equal to 128 bits, 192 bits or 256 bits. The block of cipher text is represented by a matrix of bytes [4]. So the main round operations can be summarized as follow:

A. SubBytes: a byte wise transformation that applies on each byte of the current block of 8-bits to 8-bits non-linear S-box. Composed by the inversion in the Galois Field GF (256) and by an affine transformation to obtain the output.

B. ShiftRows: The rows of the temporary result are cyclically shifted over different offsets Row 0 is not shifted, Row 1 is shifted over 1 byte, Row 2 is shifted over 2 bytes and Row 3 is shifted over 3 bytes.

C. MixColumn: another linear mapping represented by 4×4 matrix chosen for its good properties of diffusion. Each column of the input matrix is multiplied by the MixColumns matrix in the mathematical form of GF (256) that provides the corresponding column of the output matrix. We denote by $a_{i\ j}$ for *i* and *j* from 0 to 3, the coefficients of the MixColumns matrix.

D. AddRoundKey: a simple Xor operation between the input matrix and the sub-key of the current round denoted by *Ki* [5].

3. Preliminaries

The Maximum distance separable (MDS) is a class of array codes of n×n size which is convolutional codes whose free distance achieves the generalized Singleton bound. Strongly MDS codes are an interesting subclass of MDS codes. These codes are characterized by the property that their column distances reach the generalized Singleton bound at the earliest possible time step [6]. The code of length n is a set of n-tuples (called code words) of a set (called the alphabet). The distance between two code words is the number of coordinates in which they differ, i.e. if x = (xi) and y = (yi) then the distance between them is

$$d(x, y) = /\{i\ /xi \neq yi\}/$$

The minimum distance of a code C is the minimum value of d(x, y) where x and y are any two distinct code words of C. Code with minimum distance at least 2e+1 can correct up to e errors. So if we receive a code word that has been distorted in at most e entries, then we can correctly deduce which code word was sent. We say that the code is an e-error correcting code [7].

Definition1: A linear **[n, k, d)** code over **GF (2^P)** is a k-dimensional sub space of the vector space **$GF(2^P)^t$** where any two different vectors of the sub space have a Hamming distance of at least **d** and **d** is the largest number with this property.

Theorem 1: (The Singleton bound). If **C** is an **[n, k, d]** code then **$d \leq n - k + 1$**.Code that meets the Singleton bound is called MDS code. The following theorems relate the distance of a code to properties of the generator matrix **G.**

Theorem 2: A linear code **C** has distance **d** if every **d - 1** columns of the parity check matrix **H** are linearly independent and there exists some set of **d** columns that are linearly dependent.

Theorem 3: By definition, an MDS-code has distance n - k + 1. Hence, every set of n – k columns of the parity-check matrix are linearly independent. This property can be translated to a requirement for the matrix **A.**

Theorem 4: An [n, k, d) code with generator matrix **$G = [I_{k*k}\ A_{k*(n-k)}]$**
Is an MDS code if every square sub matrix of **A** is non-singular. **A** is a well-known class of MDS codes that formed by the Reed-Solomon codes, for which efficient construction algorithms are known [8].

4. The Euphrates Cipher

We have studied and devised a new cipher design to improve its strength and resistances against the rapid increasing capability and strength of computers in addition to withstand athwart known and unknown attacks. The Round function of the EUPHRATES cipher is composed of the four main transformations that repeated recursively: **SubBytes**, **RevisibleShiftColumns**, **ShiftingMixcolumn** and a bit-per-bit XOR with a round key **(Round key Addition)**. The Final Round of the EUPHRATES is composed of the same functions as a classical Round except that it does not include the **ShiftingMixcolumn** transformation. It uses the whitening concept that the key material before the first round and after the last round. Our emphasis is on the clarification of fundamental concepts and on demonstrating the feasibility of solving several central cryptographic problems related to the malicious attacks and security aspects. The proposed cipher follows the Wide Trail Strategy (such as AES) for resource-constrained environments and implementations. The EUPHRATES cipher uses a compact S-box which is the source of nonlinearity in the algorithm and the equations described it will be the main obstacle that prevents the system from being easily solved. It is similar to the AES S-box on the one hand of the work, but at the construction, it uses a different affine transform with a different constant vector and a different irreducible polynomial. The second step of the EUPHRATES is the linearity which is provided by the shifts the columns of the state matrix by a certain circular shifting which are reversible and symmetric in forward and backward. The third step of the round transformation is the shifting for the coefficient of the linear equation to generate a new equation by reordering arrangement for the same values. The mystery for this step is in the backward linear equation, since it is not used in the decryption process, because it uses the forward linear equation Xored with the complement values or in other words (differences values) that makes the forward linear equation coefficients equal to the backward linear equation coefficients as the alternative in decryption process. This means the ShiftingMixColumn step does not use the inverse linear equation, but it uses the same forward linear equation in encryption and decryption except it Xored with complement or difference values in decryption process.

This is the prominent change and the main difference between the TIGRIS and EUPHERATES cipher. The last step of the algorithm is the Xored with the ciphering Subkey via the state matrix. The round key Generation of Subkeys is generates from the input key during the operation of the algorithm that aids in thwarting cryptanalysis attacks. For more details see the main structure of the Euphrates cipher as stated in Figure (1) at the end of paper.

5. Fundamental Operations

In this section we try to describe the round transformation of the EUPHRATES cipher in detail that is also similar to the AES cipher which consists of four main stages that recur repeatedly with the number of rounds according to the key length:

5.1 SubBytes

One of the most critical and expensive operations in any cipher design is the substitution. The substitution box (S-box) which used in EUPHRATES is so far similar to the one used in AES on one hand of the design method and the mathematical representation on another. The S-box is a non-linear permutation over bytes, which is composed of two mappings: The first map treats the 8-bit quantity as an element of GF (2^8), and takes the multiplicative inverse if the element is non-zero, and otherwise just maps to 00. The second map, treats the resulting GF(2^8) element as an 8-bit vector and performs the following affine transformation f(x) over GF(2). Regarding 8-bit (byte) as elements in GF (2^8), where:

$$f(x) = \begin{cases} x^{-1}, & x \neq 0 \\ 0, & x = 0 \end{cases} \qquad (1)$$

The multiplicative inverse modulo the irreducible polynomial $\mathbf{X^8 + X^4 + X^3 + X^2 + 1}$ and the result Xored with the constant vector represented by the value **(3D)**.

$$\begin{bmatrix} b'_0 \\ b'_1 \\ b'_2 \\ b'_3 \\ b'_4 \\ b'_5 \\ b'_6 \\ b'_7 \end{bmatrix} = \begin{bmatrix} 1 & 1 & 1 & 0 & 0 & 0 & 1 & 1 \\ 1 & 1 & 1 & 1 & 0 & 0 & 0 & 1 \\ 1 & 1 & 1 & 1 & 1 & 0 & 0 & 0 \\ 0 & 1 & 1 & 1 & 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 & 1 & 1 & 1 & 0 \\ 0 & 0 & 0 & 1 & 1 & 1 & 1 & 1 \\ 1 & 0 & 0 & 0 & 1 & 1 & 1 & 1 \\ 1 & 1 & 0 & 0 & 0 & 1 & 1 & 1 \end{bmatrix} \begin{bmatrix} b_0 \\ b_1 \\ b_2 \\ b_3 \\ b_4 \\ b_5 \\ b_6 \\ b_7 \end{bmatrix} \oplus \begin{bmatrix} 1 \\ 0 \\ 1 \\ 1 \\ 1 \\ 1 \\ 0 \\ 0 \end{bmatrix} \qquad (2)$$

Table 1: Forward S-Box

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0	3D	FA	70	B4	9B	71	F9	EB	C0	44	B5	9C	5F	97	56	32
1	6D	C7	2F	8E	D7	17	ED	A4	A2	FD	68	DE	26	96	BA	74
2	15	31	40	77	34	20	E4	6F	48	53	28	7D	55	A8	F1	9A
3	F2	27	5D	6B	39	67	CC	1E	B0	4A	46	DF	50	1C	99	01
4	29	D9	95	8B	2D	62	B6	22	B9	E0	B3	EF	D1	63	14	EA
5	87	8D	0A	FF	19	F0	1D	F3	09	E8	59	7C	F5	93	EE	0B
6	DA	E7	9E	C6	A3	30	16	2A	3F	38	BE	12	C5	F6	AC	36
7	FB	94	86	AA	2E	DD	E2	B1	25	1F	AD	2C	C1	C9	05	8A
8	37	AF	E1	A7	69	0E	C8	5A	35	7E	92	23	F8	CD	B2	73
9	EF	3A	D3	A5	D4	7B	54	91	4B	72	BC	BD	07	C4	78	A9
A	60	51	CB	BF	A6	A0	5C	E5	81	CF	DB	98	83	03	F4	0D
B	89	0F	79	42	A1	24	33	4F	F7	2B	6A	3E	7A	13	88	CA
C	CE	49	FE	61	EC	64	6E	0C	DC	FC	BB	01	06	AE	18	08
D	3C	8F	11	3B	52	84	04	C3	41	00	D8	76	5B	8C	B8	65
E	5E	D0	47	AB	4E	9D	58	6C	1A	57	E3	45	D2	02	D5	85
F	9F	80	82	66	75	B7	1B	D6	43	4C	E9	90	21	C2	E6	4D

5.2 InvSubByte

The inverse of the proposed S-box is constructed by applying the inverse of the affine transformation followed by applying the multiplicative inverse in GF(2^8) and the result Xored with the constant vector represented by the value **(73)**.The rational for using constant vector is to increase the complexity of the S-box and to remove fixed point respectively.

$$\begin{bmatrix} b'_0 \\ b'_1 \\ b'_2 \\ b'_3 \\ b'_4 \\ b'_5 \\ b'_6 \\ b'_7 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 1 & 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 0 & 1 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 & 0 & 1 & 0 & 1 \\ 1 & 0 & 0 & 1 & 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 & 1 & 0 & 0 & 1 \\ 1 & 0 & 1 & 0 & 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 1 & 0 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 & 1 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} b_0 \\ b_1 \\ b_2 \\ b_3 \\ b_4 \\ b_5 \\ b_6 \\ b_7 \end{bmatrix} \oplus \begin{bmatrix} 1 \\ 1 \\ 0 \\ 0 \\ 1 \\ 1 \\ 1 \\ 0 \end{bmatrix} \qquad (3)$$

Table 2: Backward S-Box

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0	D9	CB	ED	AD	D6	7E	CC	9C	CF	58	52	5F	C7	AF	85	B1
1	3F	D2	6B	BD	4E	20	66	15	CE	54	E8	F6	3D	56	37	79
2	25	FC	47	8B	B5	78	1C	31	2A	40	67	B9	7B	44	74	12
3	65	21	0F	B6	24	88	6F	80	69	34	91	D3	D0	00	BB	68
4	22	D8	B3	F8	09	EB	3A	E2	28	C1	39	98	F9	FF	E4	B7
5	3C	A1	29	96	2C	96	0E	E9	E6	5A	87	DC	A6	32	E0	0C
6	A0	C3	45	4D	C5	DF	F3	35	1A	84	BA	33	E7	10	C6	27
7	02	05	99	8F	1F	F4	DB	23	9E	B2	BC	95	5B	2B	89	90
8	F1	A8	F2	AC	D5	EF	72	50	BE	B0	7F	43	DD	51	13	D1
9	FB	97	8A	5D	71	42	1D	0D	AB	3E	2F	04	0B	E5	62	F0
A	A5	B4	18	64	17	93	A4	83	2D	9F	73	E3	6E	7A	CD	81
B	38	77	8E	4A	03	0A	46	F5	DE	48	1E	CA	9A	9B	6A	A3
C	08	7C	FC	D7	9D	6C	63	11	86	7D	BF	A2	36	8D	C0	A9
D	E1	4C	EC	92	94	EE	F7	14	DA	41	60	AA	C8	75	1B	3B
E	49	82	76	EA	26	A7	FE	61	59	FA	4F	07	C4	16	5E	4B
F	55	2E	30	57	AE	5C	6D	B8	8C	06	01	70	C9	19	C2	53

5.3 RevisibleShiftColumns

RevisibleShiftColumns is actually transposition cipher, its only rearrange the positions of the elements without changing their identities for the encryption process the 1st column is shifted 2 byte to the bottom, 2nd column is 2 byte the first one to the bottom and the last to up, 3rd column is 2 byte the first one to the last and the second to the third byte and the 4th column remain unchanged,. For the decryption process, the operation is similar exactly to that for encryption. For decryption, the corresponding step shifts the rows in exactly implemented in the same direction without any change as a symmetric shifting. This is a good method to obtain an optimal diffusion with low cost on hardware and high fast implementation in addition to gain resistance against truncated differential attacks and saturation attacks. As shown in below figure (2).

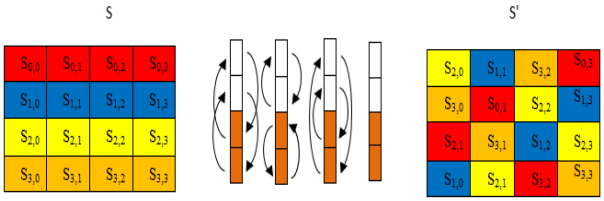

Fig. 2 RevisibleShiftColumns

5.4 ShiftingMixcolumn Transformation

Similarly to the AES round function, the ShiftingMixcolumn transformation in EUPHRATES takes a 4×4 matrix of bytes passes each byte that applies some linear transformations to the result. This linear transformation is called the "ShiftingMixcolumn". The ShiftingMixcolumn provides better diffusion and better protection against differential attacks. The shifting mixcolumn in this model uses the same equation of the original cipher except it makes a shifting to the one vector of the matrix. The forward ShiftingMixcolumn depends on row shifting, for each last row of matrix becomes the first row in the next new matrix and so on. Each four rounds repeat these operations. This step involves shifting left and exclusive-ORing bits with themselves. These operations provide both confusion and diffusion.

$$\mathbf{a(x) = \{05\}\, x^3 + \{06\}\, x^2 + \{06\}\, x + \{04\}} \quad (4)$$

$$\mathbf{b(x) = \{0a\}\, x^3 + \{0c\}\, x^2 + \{09\}\, x + \{0e\}} \quad (5)$$

$$\mathbf{a(x) = b(x) \mod (x^4 + 1)} \quad (6)$$

$$\begin{bmatrix} 04 & 05 & 06 & 06 \\ 06 & 04 & 05 & 06 \\ 06 & 06 & 04 & 05 \\ 05 & 06 & 06 & 04 \end{bmatrix} \times \begin{bmatrix} 0e & 0a & 0c & 09 \\ 09 & 0e & 0a & 0c \\ 0c & 09 & 0e & 0a \\ 0a & 0c & 09 & 0e \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$\begin{bmatrix} S'_{0,c} \\ S'_{1,c} \\ S'_{2,c} \\ S'_{3,c} \end{bmatrix} \otimes \begin{bmatrix} 04 & 05 & 06 & 06 \\ 06 & 04 & 05 & 06 \\ 06 & 06 & 04 & 05 \\ 05 & 06 & 06 & 04 \end{bmatrix} \oplus \begin{bmatrix} S_{0,c} \\ S_{1,c} \\ S_{2,c} \\ S_{3,c} \end{bmatrix} \otimes \begin{bmatrix} 0a & 05 & 06 & 03 \\ 03 & 0a & 05 & 06 \\ 06 & 03 & 0a & 05 \\ 05 & 06 & 03 & 0a \end{bmatrix} = \begin{bmatrix} S'_{0,c} \\ S'_{1,c} \\ S'_{2,c} \\ S'_{3,c} \end{bmatrix} \otimes \begin{bmatrix} 0e & 0a & 0c & 09 \\ 09 & 0e & 0a & 0c \\ 0c & 09 & 0e & 0a \\ 0a & 0c & 09 & 0e \end{bmatrix}$$

$$\begin{bmatrix} S'_{0,c} \\ S'_{1,c} \\ S'_{2,c} \\ S'_{3,c} \end{bmatrix} \otimes \begin{bmatrix} 05 & 06 & 06 & 04 \\ 04 & 05 & 06 & 06 \\ 06 & 04 & 05 & 06 \\ 06 & 06 & 04 & 05 \end{bmatrix} \oplus \begin{bmatrix} S_{0,c} \\ S_{1,c} \\ S_{2,c} \\ S_{3,c} \end{bmatrix} \otimes \begin{bmatrix} 04 & 08 & 04 & 0a \\ 0a & 04 & 08 & 04 \\ 04 & 0a & 04 & 08 \\ 08 & 04 & 0a & 04 \end{bmatrix} = \begin{bmatrix} S'_{0,c} \\ S'_{1,c} \\ S'_{2,c} \\ S'_{3,c} \end{bmatrix} \otimes \begin{bmatrix} 09 & 0e & 0a & 0c \\ 0c & 09 & 0e & 0a \\ 0a & 0c & 09 & 0e \\ 0e & 0a & 0c & 09 \end{bmatrix}$$

$$\begin{bmatrix} S'_{0,c} \\ S'_{1,c} \\ S'_{2,c} \\ S'_{3,c} \end{bmatrix} \otimes \begin{bmatrix} 06 & 06 & 04 & 05 \\ 05 & 06 & 06 & 04 \\ 04 & 05 & 06 & 06 \\ 06 & 04 & 05 & 06 \end{bmatrix} \oplus \begin{bmatrix} S_{0,c} \\ S_{1,c} \\ S_{2,c} \\ S_{3,c} \end{bmatrix} \otimes \begin{bmatrix} 06 & 03 & 0a & 05 \\ 05 & 06 & 03 & 0a \\ 0a & 05 & 06 & 03 \\ 03 & 0a & 05 & 06 \end{bmatrix} = \begin{bmatrix} S'_{0,c} \\ S'_{1,c} \\ S'_{2,c} \\ S'_{3,c} \end{bmatrix} \otimes \begin{bmatrix} 0c & 09 & 0e & 0a \\ 0a & 0c & 09 & 0e \\ 0e & 0a & 0c & 09 \\ 09 & 0e & 0a & 0c \end{bmatrix}$$

$$\begin{bmatrix} S'_{0,c} \\ S'_{1,c} \\ S'_{2,c} \\ S'_{3,c} \end{bmatrix} \otimes \begin{bmatrix} 06 & 04 & 05 & 06 \\ 06 & 06 & 04 & 05 \\ 05 & 06 & 06 & 04 \\ 04 & 05 & 06 & 06 \end{bmatrix} \oplus \begin{bmatrix} S_{0,c} \\ S_{1,c} \\ S_{2,c} \\ S_{3,c} \end{bmatrix} \otimes \begin{bmatrix} 04 & 0a & 04 & 08 \\ 08 & 04 & 0a & 04 \\ 04 & 08 & 04 & 0a \\ 0a & 04 & 08 & 04 \end{bmatrix} = \begin{bmatrix} S'_{0,c} \\ S'_{1,c} \\ S'_{2,c} \\ S'_{3,c} \end{bmatrix} \otimes \begin{bmatrix} 0a & 0c & 09 & 0e \\ 0e & 0a & 0c & 09 \\ 09 & 0e & 0a & 0c \\ 0c & 09 & 0e & 0a \end{bmatrix}$$

5.5 Inverse ShiftingMixcolumn

The backward ShiftingMixcolumn (Inverse ShiftingMixcolumn) relies on shifting column, for each last column of the matrix becomes the first column in the next new matrix and so on for each four rounds also will repeat these operations, in order to superpose the same sequence of forward matrices representative by shift rows. The same matrix of forward mixcolumn uses in backward mixcolumn except it Xored with the complement (difference values) matrix with shifting vector for each round as the inverse work.

5.6 The Round Key Addition

In this operation the Round Key is applied to the State matrix by a simple bitwise EXOR operation. The Round Key is derived from the cipher key by means of the key schedule. The transformation that consists of EXORing a Round Key to the State is denoted by: AddRoundKey (State, RoundKey) that occurs at each round.

5.7 Key Schedule

Key schedule is the main part of the key expansion that responsible for accepting a 128-bit input key and producing 128-bit ciphering key which represent the secret key. The key expansion specifies how expanded key derived from the cipher key, because the encryption and decryption requires one round of ciphering key for the initial key state and one that generated for each round. The

key expansion has been modified to be possible to execute the key schedule using a small amount of working memory with high key agility on a wide range of processors that implemented with two constants vectors to eliminate any symmetries or weak keys, represented by base natural algorithm and golden ratio in order to gives an efficient diffusion of cipher key differences into the expanded key. The key generation depends upon the novel technique that works in two directions to generate strong ciphering keys, as stated in below figure (3).

Note

Pw = base natural algorithm (b7e15163),
Qw = golden ratio (9e377969)
<< **8-bit** =right rotate over the vector

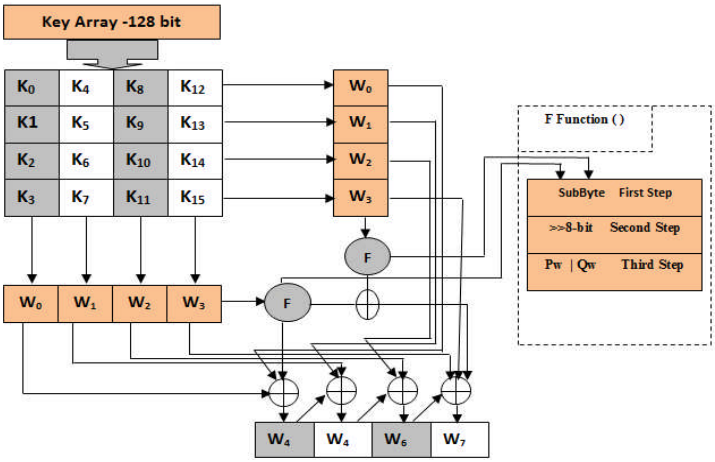

Fig. 3 EUPHRATES Key Expansion

6. Security Analysis

The proposed cipher does not base its security or part of it on obscure and not well understood interactions between arithmetic operations and algebraic notations. We increase the key complexity as well as the block encryption passes through the several stages in order to improve security and enhanced the resistance against the malignant actions. Therefore the linear and differential cryptanalysis require more time than Rijndael to break our proposed cipher. Our envisioned target application can be work efficiently on a constrained device, e.g. a low-cost passive RFID-tag or similar. By re-ordering the input and output bytes, or by make simple shift and extra Xoring operations to reduce the area significantly. We have studied the efficiency of the key avalanche and conclude that the complexity of exhaustive search if one tries to bypass one round it will be difficult to analyze. The proposed cipher focuses on increasing the algebraic complexity for the S-box and its inverse to defeat algebraic attacks and interpolation attack. This cipher is slower than the Rijndael cipher especially in the decryption process compared with encryption process, and the implementation require more time for the key scheduling. The internal algebraic operations constitute a form of security margin against known attacks by minimizing the correlation between linear transformations of input/output bits to face linear attack, and at the same time minimizing the difference propagation probability to defeat differential attack, the number of rounds remain a fixed similar to AES to defeats the square and related key attack that is possible for reduced rounds. The Euphrates cipher uses wise and precise techniques in generating Subkeys with the ability to change keys and with a minimum of resources (high key agility). Simple assessment for the time implementation between the AES and the EUPHRATES cipher explained in table (3). Written and Implemented by visual C# in dot Net environment using the Pentium4 of CPU 2.00GHz running Microsoft Windows7 ultimate version.

Table 3: Comparison Speed between EUPHRATES Cipher and the AES

Algorithms	Block Size	Key Size	Time of Encryption in MS
Original-AES (Rijndael)	128-bit	128-bit	0.0468
		192-bit	0.0476
		256-bit	0.0498
EUPHRATES Cipher	128-bit	128-bit	0.0501
		192-bit	0.0526
		256-bit	0.0543

7. Conclusions

A simple, efficient and secure block cipher has been proposed. The main goal of the research is to design a new algorithm that withstands adversarial behavior, and moreover it has realized high speed implementation on various platforms as well as high performances. So this paper has been conducted to design a practical and strong SPN cryptosystem. The strength of this cryptosystem depends on the strength of its elements which are designed according to the standard criteria and the wide trail strategy. The proposed cipher can provide a provable security against traditional attacks. EUPHRATES cipher designed with increasing in confusion and diffusion layers through the S-box stage and the ShiftingMixcolumn stage respectively in order to encounter the known and un-known attacks.

Acknowledgments

I would like to apply special thanks to Dr. Hazim Hakoosh for his great efforts in the revising of this paper from the grammatical terms that have improved this paper significantly.

References

[1] Devesh C. Jinwala, Dhiren R. Patel and Kankar S. Dasgupta, "Optimizing the Block Cipher Resource Overhead at the Link Layer Security Framework in the Wireless Sensor Networks", Proceedings of the World Congress on Engineering, Vol-I, 2008.
[2] Joan Daemen and Vincent Rijmen, "The design of Rijndael: AES the Advanced Encryption Standard", Springer-Verlag, 2002.
[3] Liam Keliher, "Refined Analysis of Bounds Related to Linear and Differential Cryptanalysis for the AES",Department of Mathematics and Computer Science, Mount Allison, University, Sackville, New Brunswick, Canada" Springer-Verlag Berlin Heidelberg 2005.
[4] Marine and Minier, "A Three Rounds Property of the AES", Universities' Paris 8 - INRIA, Project CODES, Springer-Verlag Berlin Heidelberg, 2005.
[5] Pawel Chodowiec, Po Khuon and Kris Gaj, "Fast implementations of secret-key block ciphers using mixed inner- and outer-round pipelining",FPGA'01, Monterey, CA, February 11-13, 2001.
[6] Ryan Hutchinson and Jochen Trumpf, "On Super regular Matrices", April 29, 2004.
[7] Simeon Ball and Zsuzsa Weiner, "An Introduction to Finite Geometry", 29 March 2007.
[8] Takeshi Sugawara, Naofumi Homma, and Takafumi et al, "ASIC Performance Comparison for the ISO Standard Block Ciphers", JWIS, 2007.

Omer Abdulrahman Dawood was born in Habanyah, Anbar, Iraq (1986), now live in Ramadi, Anbar. He obtained B.Sc. (2008), M.Sc. (2011) in Computer Science from the College of Computer, Anbar University. He is teaching staff member in English Department in College of Education for humanities sciences, Anbar University, and now he is a Ph.D. student at the Technology University-Baghdad. His research interests Data and Network Security, Coding, Number Theory and Cryptography.

Prof. Abdul Monem S. Rahma Ph.D Awarded his M.Sc. from Brunel University and his Ph.D. from Loughborough University of technology United Kingdom in 1982, 1984 respectively. He taught at Baghdad university department of computer science and the Military Collage of Engineering, computer engineering department from 1986 till 2003. He fills the position of Dean Asst. of the scientific affairs and works as a professor at the University of Technology computer Science Department. He published 88 Papers, 4 Books in the field of computer science, supervised 28 Ph.D. and 57 M.Sc. students. His research interests include Computer graphics image processing, Biometrics and Computer Security. And he attended and submitted in many scientific global conferences in Iraq and many other countries. From 2013 to Jan. 2015 he fills the position of Dean of the computer Science Department at the University of Technology.

Abdul Mohssen J. Abdul Hossen is an Associate Professor of applied mathematics, Computer Science Department, University of Technology, where he teaches undergraduate and graduate courses in mathematics. Abdul Hossen received the B. Sc. degree in mathematics from Mustansiriyah University, Iraq in1977, the M. Sc. degree in applied mathematics from Bagdad University, Iraq. in1980, the PH.D degree in applied mathematics from University of Technology, Iraq. In 2005. He is a member of the IEEE system, and Member of the editorial Journal.

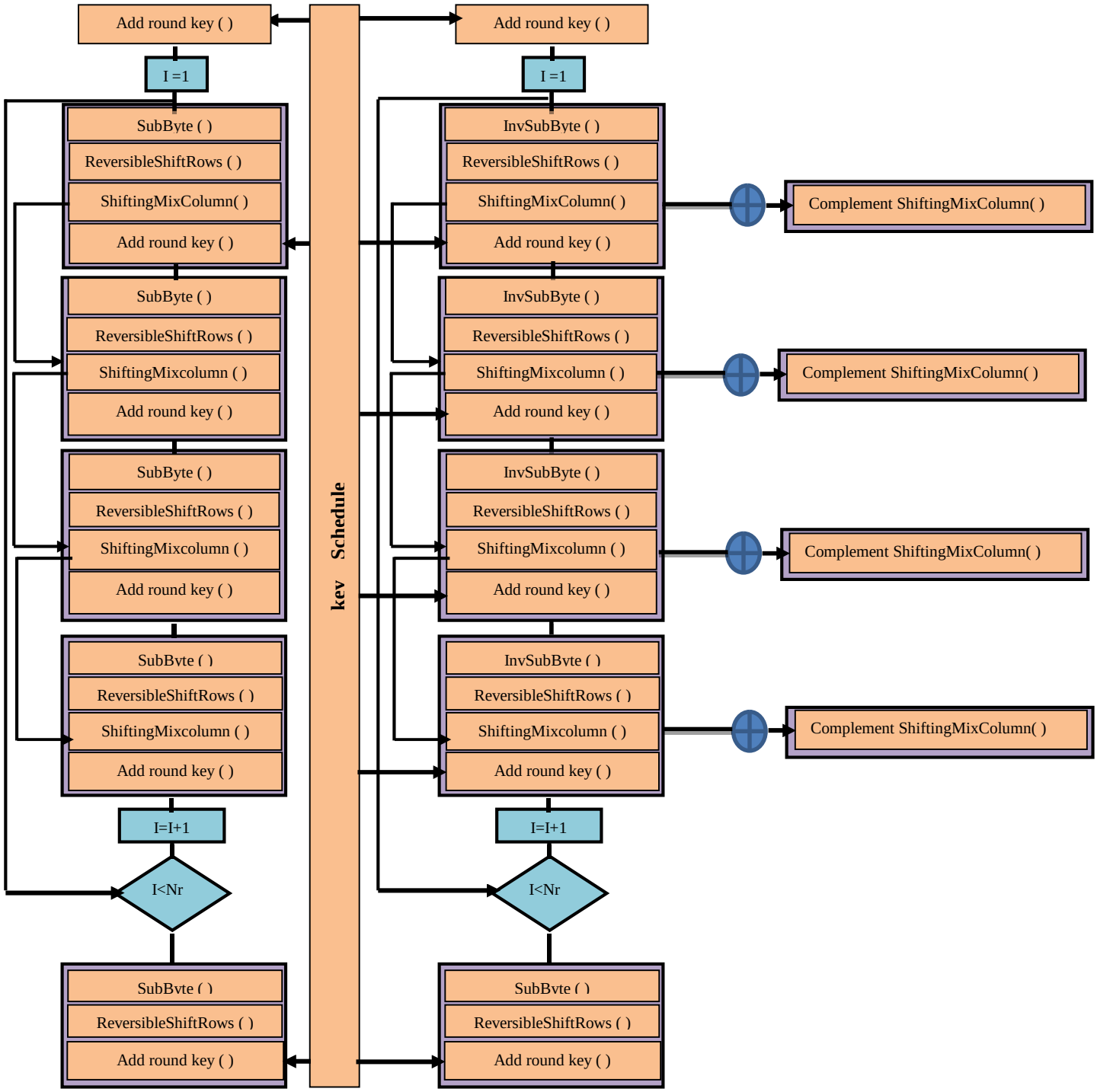

Fig. 1 The EUPHRATES Structure