Modification Advanced Encryption Standard for Design Lightweight Algorithms

Mustafa M. Abd Zaid
University Of Technology
Baghdad, Iraq
mustafamajeed2014@gmail.com

Soukaena Hassan
University Of Technology
Baghdad, Iraq
Soukaena.Hassan@Yahoo.Com

Received Sep. 20, 2018. Accepted for publication Dec. 31, 2018

DOI : *http://dx.doi.org/10.31642/JoKMC/2018/060104*

Abstract

The acknowledgment of IoT requires a colossal measure of sensor hubs which have limited battery power, memory, computational latency and communication bandwidth to obtain contributions from the associated objects. In the current developments of the resource constraint environments, the trend is shifted towards lightweight cryptographic algorithm. Many lightweight cryptographic algorithms have been developed and also existing algorithms are modified in terms of resource constraint environment.

In this paper we analyzed a very popular block cipher AES-128 and tried to make it lightweight regarding energy consumption. In modified AES algorithm an execution of the AES mix columns operation is proposed, combine the add round-key operation with mix-columns to perform in one cycle, the shift-row operation has been modified to shift-rows and shift-columns and reduction in number of rounds to 6 rounds only for modified AES and compare the results of standard algorithm and modified algorithms. The proposed algorithms passed the statistical test suite, also the new algorithms faster than standard AES in term of implementation. In security term the modified algorithm for 6 round has more confusion and diffusion percent because of the modified in mix-columns and shift-rows operations.

1. Introduction

In the present time, the request of security is expanding quickly as the result of numerous users are sharing public and private data over web and the utilization of web is expanding at higher rate. As the information and data is exceptionally delicate as its transmission is required constantly, this offers ascend to the need of security. Encryption strategy is a standout amongst the most imperative viewpoints which are exceptionally helpful to anchor private data [1].

In 2001, NIST (National Institute of Standards and Technology) determined (Advanced Encryption Standard (AES)), likewise known by its unique name for the encryption of electronic information.

NIST determined AES algorithm to replace DES (Data Encryption Standard). AES must be a block cipher fit for taking care of 128-bit blocks, utilizing keys estimated at 128, 192, and 256 bits.

The symmetric block cipher (AES) can encode (encrypt) and decode (decrypt) data. The information is changed to an incomprehensible form called ciphertext by "the Encryption"; changing the information again into its original form called plaintext by "the Decryption" [2].

The number of rounds in AES relies upon the length of the key therefore it is variable comparing to DES. For 128-bit keys AES utilizes 10 rounds, 192-bit keys AES utilizes 12 rounds and 256-bit keys AES utilizes 14 rounds. An alternate 128-bit round key is used by every one of these rounds. The 128-bit round key is figured from the standard AES key.

The computing devices utilized as a part of an extensive class of remote correspondence systems, for example, cell phones, remote sensor systems (WSNs), vehicular ad hoc networks (VANETs), mobile ad hoc networks (MANETs), Internet of Things (IoT), body area networks (BANs) and so on, are little and asset compelled. In the current developments of the resource constraint environments, the trend is shifted towards

lightweight cryptographic algorithm. Many lightweight cryptographic algorithms have been developed and also existed algorithms are modified in terms of resource constraint environments.

Eslam et al[3] produced for both forward and reverse MixColumns steps an elective lightweight design which is required in the AES hardware usage. The examinations showed that the suggested MixColumn design has less complexity than past important work in door estimate and number of clock cycles. As per [3] that compact design can help in executing AES for RFID Tags, smart cards, and remote sensors.

According to Nada et al[5], the functions ShiftRow has been modified in original AES to accomplish higher many-sided quality keeping the required time for encryption and decryption forms close to the same. The adjusted AES utilizing five diverse keys in each round for the two activities, while in standard AES the ShiftRow utilizes a solitary key for encryption and unscrambling process.

Puneet et al [1], watched that by expanding the number rounds to 16 made the system less inclined to the attackers and more secure. The hacker will require more computational time to break the system because the expansion in number of rounds. The generation of key has been finished with the assistance of the Polybius square. Thus, the security of the system has been enhanced.

Ahmed et al[4], focused on convert the sequence steps in AES classic algorithm to random steps inside modified AES algorithm by depended on some parts of keys. Step sequence in Shift Row is shift left second, third and fourth row by constant value, in modified the Shift Row the number of left shift is unknown because the number of shift depended on part of key value. Also step sequence in Add Round Key is XOR between data matrix and sequence sub key is constant (11 sub key), in modified the sequence sub key is random because sub key depended on part of key value. This random steps increase the confusion and diffusion in data encryption.

According to Rizky et al [6], an optimization has been successful done for AES. Enhancement has been made by constringes shiftrow circular process and SBox change for MixColumn transformation. Combination of MixColumn with modified SBox cause SubByte process is unnecessary. The examinations showed that percentage improvement of encryption/decryption processes. In any case, the strategy need to devour greater memory to store two modified SBox map and array ShiftRow map.

For A.Pratthiba et al[7], The essential target of this work was to design a lightweight, secure ideal SBox that suits IoT applications. The combinational design in the finite field for the hardware usage of the 4×4 SBox is exhibited.

Ming et al[8], presented a detailed overview of the various design of AES S-boxes, which are classified as the hardcoded LUT S-box, the pure combinatorial S-box using composite field arithmetic (CFA), the pipelined version of CFA S-Box, the AES S-box using direct computation and Linear Feedback Shift Register (LFSR) based S-Box.

2. The principle of the AES algorithm

Four fundamental operation blocks separate AES where information are dealt with at either byte or bit level. "State" is used to store out the variety of bytes as a 4*4 array, the following four steps are the essential steps in AES algorithm:

STEP 1: For byte-by-byte substitution amid the forward procedure, this step is known as SubBytes. InvSubBytes is the comparing substitution step utilized amid decryption.

- This process uses for discover a substitution byte for a given byte in the input state, utilizing a 16 × 16 query table.
- The sections in the query table are made by utilizing the thoughts of multiplicative inverses in GF(2^8) and bit mix to annihilate the bit-level relationships inside each byte. Figure 1 shows the substation byte operation

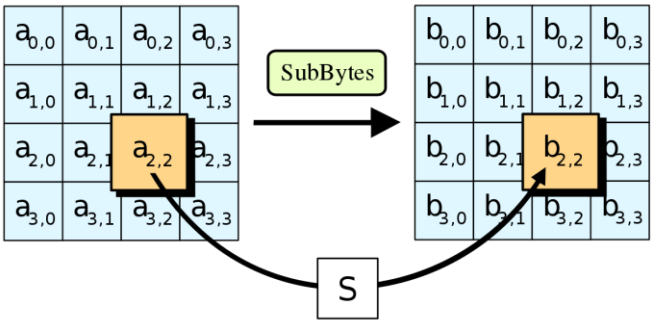

Fig1: SubByte operation

STEP 2: known ShiftRows (moving the rows of the state array amid the forward procedure). InvShiftRow (The relating change amid backward process).

Figure 2 show shiftROW operation.

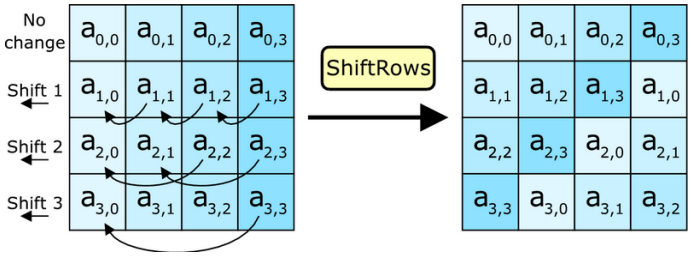

Fig2: shiftROW operation.

- To scramble the byte arrange inside each 128-bit block is the objective of this change.

STEP 3: known as MixColumns for stirring up of the bytes in every column independently amid the forward step. InvMixColumns is The corresponding transformation during decryption step.

The objective is to additionally mix up the 128-bit input block. each bit of the ciphertext is relied upon on every bit of the plaintext after 10 rounds of preparing by the ShifRows process along with the MixColumn process.

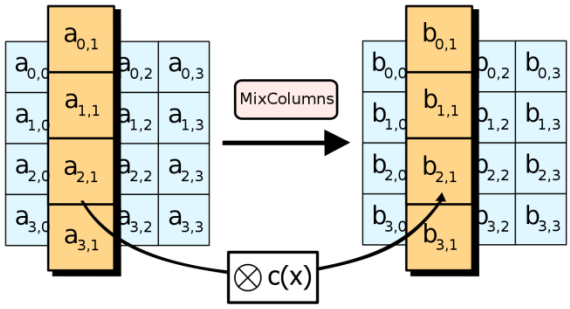

Fig3:MixColumns operation

STEP 4: known as (AddRoundKey):

Key scheduling or round key expansion is a method used for producing round keys. XOR the two previous columns are used to produce the sub keys which is gotten from the first key. The AES uses nine iterations for 128-bit round key, barring the first and the last round. Every bytes of the array (regard GF (2) is added to a byte of the comparing array of round sub keys[3]. Figure4 shows AddRoundKey operation.

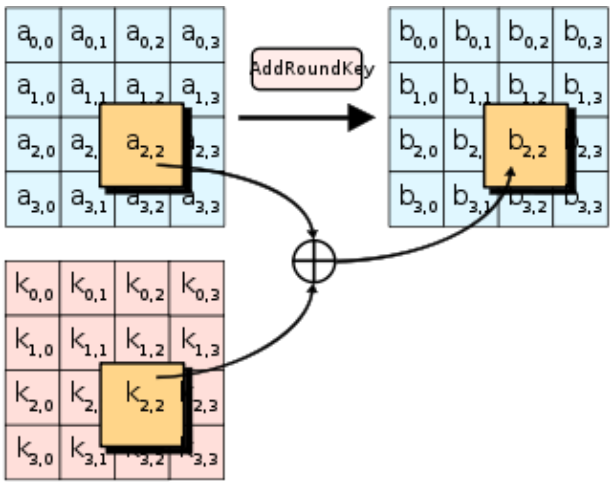

Fig4: AddRoundKey operation

The block diagram of the standard AES system is shown below figure 5.

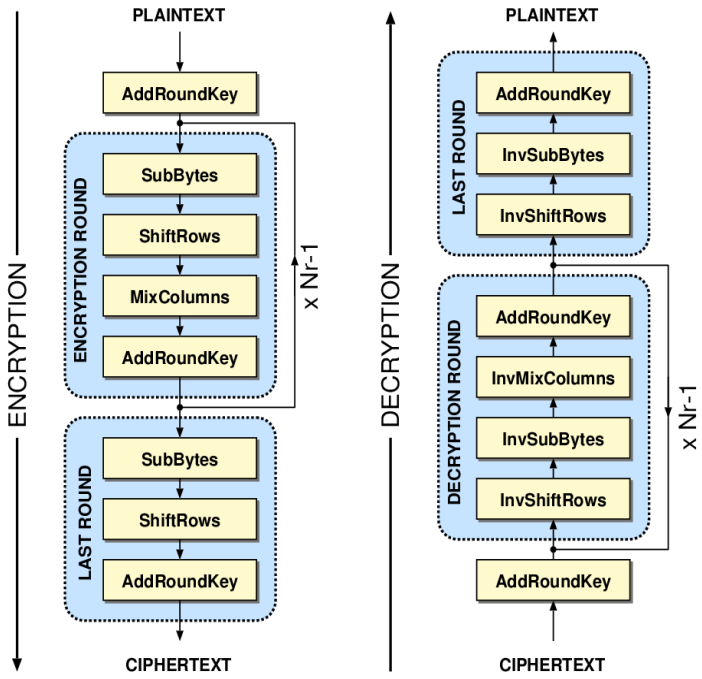

Fig5: Flowchart diagram for AES algorithm.

3. METHODOLOGY

To defeat the issue of computational overhead and high calculation, we break down AES (Advanced Encryption Standard) and modify it, to diminish the algorithm's calculation. So we create and execute an altered AES based Algorithm for all sort of data. The essential plan to alter AES is to give better security for information and less calculation. The modified AES algorithms acclimate to give better encryption and decryption speed. The length of key and block are specified as standard AES algorithm. To defeat the issue of high computation we have proposed an alternative

lightweight design for both forward and backward MixColumns operation required in the AES execution and combination with AddRound key operation to reduce memory used and an altered proposed for shift row by combine(shift rows and shift columns).

3.1 Rounds of Modified-AES Algorithm

We proposed two modifying algorithms, the first one with 10 rounds for full encryption/decryption operations and the second use only 6 rounds for full encryption/decryption operations. The three steps that we use for modified algorithms included:

Modified AES Algorithms are:

- SubBytes
- Modified ShiftRows
- Mix columns & AddroundKey

Substitution Bytes, stay unaffected as it is in the standard AES.

3.1.1 Substitution bytes : it is a non-linear byte substitution operation. The two sub transformations are made out it; the first one is multiplicative inverse and the second is affine transformation. These two sub steps in many executions are joined into a solitary table query called SBox.

3.1.2 Modified ShiftRows: The reason for the ShiftRows is to distribute the bytes of every input column to different output columns. This progression is a linear diffusion process. In modified ShiftRow cyclical moving process for key matrix in each row and column, the arrays' rows and columns are rotated by a specific number of byte positions to expand diffusion more than standard AES. Figure 6 shows modified ShiftRows operation.

For few reiterations, the overhead could be expelled through and through by supplanting the loop with the code parts for that settled number of times. This technique is known as loop unrolling[9].

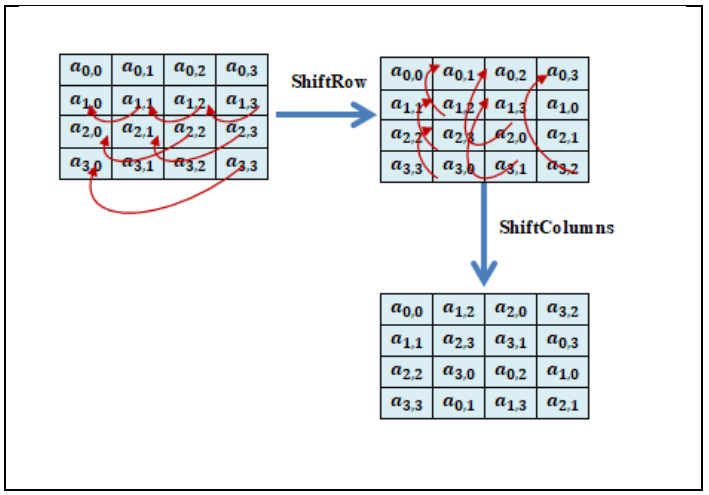

Fig6:Modified ShiftRow

Python Pseudocode of modified ShiftRow:

```
def shiftRows(state, isInv):
    s=list(state)
    if isInv:
        state[0], state[6], state[8], state[14] = s[0], s[1], s[2], s[3]
        state[5], state[11], state[13], state[3] = s[4], s[5], s[6], s[7]
        state[10], state[12], state[2], state[4] = s[8], s[9], s[10], s[11]
        state[15], state[1], state[7], state[9] = s[12], s[13], s[14],s[15]
    else:
        state[0], state[1], state[2], state[3] = s[0], s[6], s[8], s[14]
        state[4], state[5], state[6], state[7] = s[5], s[11], s[13], s[3]
        state[8], state[9], state[10], state[11] = s[10], s[12], s[2], s[4]
        state[12],state[13], state[14], state[15] = s[15], s[1], s[7], s[9]
    return state
```

3.1.2 MixColumns & AddroundKey:

In this step, we combine the modified MixColumns operation with AddroundKey operation. The bytes of column is mapped into another esteem in MixColoumn step that is an element of every one of the four bytes in that column. The following matrix transformation characterizes for each component in the

product matrix which is the sum of products of components of one row and one column.

$$\begin{bmatrix} r_0 \\ r_1 \\ r_2 \\ r_3 \end{bmatrix} = \begin{bmatrix} 2 & 3 & 1 & 1 \\ 1 & 2 & 3 & 1 \\ 1 & 1 & 2 & 3 \\ 3 & 1 & 1 & 2 \end{bmatrix} \begin{bmatrix} a_0 \\ a_1 \\ a_2 \\ a_3 \end{bmatrix}$$

Fig7:MixColumn

For this situation, the multiplications and individual additions are performed in GF (2^8)[3].

A single column transformation (col(0≤ col≤ 3)) of state in the MixColumns can be defined as:-

$$\begin{aligned} col[0] &= (col[0]*\{02\}) \wedge (col[3]*\{03\}) \wedge col[2] \wedge col[1] \\ col[1] &= (col[1]*\{02\}) \wedge (col[0]*\{03\}) \wedge col[3] \wedge col[2] \\ col[2] &= (col[2]*\{02\}) \wedge (col[1]*\{03\}) \wedge col[0] \wedge col[3] \\ col[3] &= (col[3]*\{02\}) \wedge (col[2]*\{03\}) \wedge col[1] \wedge col[0] \end{aligned} \quad (1)$$

Utilizing the identity $\{03\} \cdot y = (\{02\} \cdot y)\, y$, now equation (1) can modify as follows:

$$\begin{aligned} tmp1 &= (col[0] \wedge col[1] \wedge col[2] \wedge col[3]) \\ col[0] &= (col[0] \wedge tmp1 \wedge (\{02\}* (col[0] \wedge col[1]))) \\ col[1] &= (col[1] \wedge tmp1 \wedge (\{02\}* (col[1] \wedge col[2]))) \\ col[2] &= (col[2] \wedge tmp1 \wedge (\{02\}* (col[2] \wedge col[3]))) \\ col[3] &= (col[3] \wedge tmp1 \wedge (\{02\}* (col[3] \wedge col[0]))) \end{aligned} \quad (2)$$

Finally modified mix columns can implement using = 116 XORs with 2 input[3].

We can combine modified mix columns with add round key to get a small number of repetitions using the following formula:

$$\begin{aligned} col[0] &= (col[0] \wedge tmp1 \wedge (\{02\}* (col[0] \wedge col[1])))\wedge roundkey[0] \\ col[1] &= (col[1] \wedge tmp1 \wedge (\{02\}* (col[1] \wedge col[2])))\wedge roundkey[1] \\ col[2] &= (col[2] \wedge tmp1 \wedge (\{02\}* (col[2] \wedge col[3])))\wedge roundkey[2] \\ col[3] &= (col[3] \wedge tmp1 \wedge (\{02\}* (col[3] \wedge col[0])))\wedge roundkey[3] \end{aligned} \quad (3)$$

In decryption process, the inverse MixColumn transformation uses in decryption process, is expressed matrix multiplication as following:

$$\begin{bmatrix} d_0 \\ d_1 \\ d_2 \\ d_3 \end{bmatrix} = \begin{bmatrix} 14 & 11 & 13 & 9 \\ 9 & 14 & 11 & 13 \\ 13 & 9 & 14 & 11 \\ 11 & 13 & 9 & 14 \end{bmatrix} \begin{bmatrix} b_0 \\ b_1 \\ b_2 \\ b_3 \end{bmatrix}$$

Fig8:Inverse MixColumn

the sum of products of components of one row and one column is forming each component in the product matrix. In this situation, the multiplications and individual additions are performed in GF (28). The MixColumns transformation on a single column col (0≤ col ≤ 3) of State can be defined as:-

$$\begin{aligned} col[0] &= col[0]*\{14\} \wedge col[3]*\{11\} \wedge col[2]*\{13\} \wedge col[1]*\{09\} \\ col[1] &= col[0]*\{09\} \wedge col[3]*\{14\} \wedge col[2]*\{11\} \wedge col[1]*\{13\} \\ col[2] &= col[0]*\{13\} \wedge col[3]*\{9\} \wedge col[2]*\{14\} \wedge col[1]*\{11\} \\ col[3] &= col[0]*\{11\} \wedge col[3]*\{13\} \wedge col[2]*\{09\} \wedge col[1]*\{14\} \end{aligned} \quad (4)$$

Equation set (4) is formulated to simplify:

$$\begin{aligned} tmp1 &= \{09\}*(cpy[0] \wedge cpy[1] \wedge cpy[2] \wedge cpy[3]) \\ col[0] &= cpy[0] \wedge tmp1 \wedge (\{02\}*(\{02\}* (cpy[0] \wedge cpy[2]))\wedge (\{02\}*(cpy[0] \wedge cpy[1]) \\ col[1] &= cpy[1] \wedge tmp1 \wedge (\{02\}*(\{02\}* (cpy[1] \wedge cpy[3]))\wedge (\{02\}*(cpy[1] \wedge cpy[2]) \\ col[2] &= cpy[2] \wedge tmp1 \wedge (\{02\}*(\{02\}* (cpy[0] \wedge cpy[2]))\wedge (\{02\}*(cpy[2] \wedge cpy[3]) \\ col[3] &= cpy[3] \wedge tmp1 \wedge (\{02\}*(\{02\}* (cpy[1] \wedge cpy[3]))\wedge (\{02\}*(cpy[3] \wedge cpy[0]) \end{aligned} \quad (5)$$

this modified form of inverse mix column uses 198 XOR gates in decryption face.
Finally we combine mix column operation with Add round key in the same cycle to reduce number of repetitions[3].

4. Experiment Result:

4.1 NIST test suite:

The statistical test suite (NIST) is the most broadly utilized one in the field of cryptography, which we have utilized for contrasting the standard AES and proposed AES Algorithms. In this paper, three tests namely Approximate entropy, Run test and Linear complexity in the statistical test Suit are used to measure the randomness of the cipher-text created from AES and proposed AESs.

In the wake of applying the NIST test suite, we use 10 random keys to test algorithms as showing in table1.

In this paper, the significance level, (p-value) is set to 0.01. The statistical tests suit (NIST) results indicate success of output (cipher-text) of all tested algorithms. In other words, all the test type of statistical test Suit are adequate and have good randomness for the all tested algorithms.

Table 1:NIST test suite:

KEY	Standard AES			Modified AES			Modified AES(6-R)		
	Approximate entropy	Run test	Linear complexity	Approximate entropy	Run test	Linear complexity	Approximate entropy	Run test	Linear complexity
9pGjSKZGXqm3cCL5	0.272	0.7045	0.42319	0.0421	0.5244	0.2854	0.4084	0.8193	0.7395
9GAFveP7fRsBV8a6	0.5709	0.2969	0.98	0.0822	0.3458	0.2513	0.9073	0.9972	0.3208
BBYWHL7v6Sz2FCtC	0.28	0.7447	0.3695	0.04058	0.4506	0.0374	0.7924	0.2568	0.7272
L7nw9Nje9G7DEEbm	0.6574	0.6017	0.097	0.2518	0.6775	0.8571	0.3115	0.6566	0.2983
mgD3D7uCQY4hLA9R	0.6574	0.6017	0.097	0.5199	0.192	0.6766	0.1944	0.4585	0.07181
pzLJd2R4teKeLs22	0.3425	0.8411	0.07	0.0612	0.5982	0.8542	0.1731	0.2464	0.8866
6c2XS5LpaF7kSDwE	0.0193	0.6948	0.979	0.5649	0.4016	0.8821	0.4712	0.6821	0.1413
pkPwfCEpt2AZjHvj	0.2318	0.8042	0.561	0.1861	0.1206	0.4966	0.5997	0.6133	0.93
GPtLySJcB4K48nyx	0.0417	0.0496	0.5121	0.4006	0.3883	0.7272	0.1217	0.1645	0.7272
KySFjuFdwkJPSP5v	0.621	0.8869	0.7572	0.2992	0.1162	0.8821	0.1866	0.6565	0.3919
Average	0.3694	0.62	0.484	0.33646	0.38	0.595	0.4166	0.55	0.523

4.2 Encryption/Decryption Time

We used different size of text files to test the speed of the proposed algorithms, and we compared the calculated time of both the Modified AES algorithms with original AES algorithm.

Table 2:Encryption/Decryption Time

File Size	Standard AES		Modified AES		Modified AES (6 Round)	
	Encryption	Decryption	Encryption	Decryption	Encryption	Decryption
3 KB	2.128	2.26	0.753	1.77	0.47	0.98
21 KB	24.11	25.04	7.77	18.02	4.35	9.7
25 KB	27.7	27.85	8.79	22.2	5.22	11.85

In this evaluation step we have tested several files in order to prove that how fast the Modified AES algorithms than the standard AES.

According to this test, we can indicate that the modified AES algorithms are faster than AES algorithm.

4.3 confusion and diffusion

Confusion and diffusion are two techniques that symmetric ciphers should fulfill to upset cryptanalysis. In a block cipher with good diffusion, on the off chance that one bit of the plaintext digit is changed, at that point influences many ciphertext digits in a random mode. Cryptographic diffusion test is a sort of factual test that assesses a block cipher for diffusion.

Modified AES(with 6 rounds) shows confusion and diffusion is stronger than that happens in original AES and modified AES (with 10 rounds).

High confusion and diffusion is come from the modified in ShiftRow and MixColumns operations:

Table 3:Confusion/Diffusion test

	Standard AES/encryption	Modified AES/encryption	Modified AES (6 Round)/encryption
Diffusion	0.5094	0.4482	0.5363
Confusion	0.506	0.4444	0.5164

5.Conclusion

In this paper we have proposed a modification lightweight design for both forward ShiftRows and backward ShiftRows (inverse) and both forward and backward MixColumns (inverse) operation required in the AES which is combined with AddRound key operation to reduce time of encryption and decryption operations.

In form of security examination and experimental results our proposed encryption algorithms are fast and then again it gives great security and includes less overhead the information. The modified AES (with 6 rounds) is the faster between the modified algorithms.

Reference:

[1]: P.Kumar and S.B. Rana, "Development of modified AES algorithm for data security", Elsevier,2015.

[2]: P.Kawle, A.Hiwase, G.Bagde, E.Tekam and R.Kalbande, "Modified Advanced Encryption Standard", International Journal of Soft Computing and Engineering (IJSCE), ISSN: 2231-2307, Volume-4, Issue-1,pp.21-23, March 2014.

[3]: E.G.Ahmed, E.Shaaban and M.Hashem, "Lightweight Mix Columns Implementation for AES" Proceedings of the 9th WSEAS International Conference on APPLIED INFORMATICS AND COMMUNICATIONS (AIC '09), ISSN: 1790-5109, pp.253-258,2013.

[4]: A.T.Sadiq and F.H.Faisal, "Modification AES algorithm based on Extended Key and Plain Text" , Journal of Advanced Computer Science and Technology Research, ISSN: 2231-8852, Vol.5 No.4, , pp. 104-112,2015.

[5]: N.M. Ali, A.M.S. Rahma, A.M.Jaber and S.Yousef, "A Byte-Oriented Multi Keys Shift Rows Encryption and Decryption Cipher Processes in Modified AES", International Journal of Scientific & Engineering Research, Volume 5, Issue 4,ISSN 2229-5518,pp.953-955,2014.

[6]:R.Riyaldhi , Rojali and A.Kurniawan, "IMPROVEMENT OF ADVANCED ENCRYPTION STANDARD ALGORITHM WITH SHIFT ROW AND S.BOX MODIFICATION MAPPING IN MIX COLUMN", Elsevier, pp.401-407,2017.

[7]: A. Prathiba and V. S. K.Bhaaskaran, "Lightweight S-Box Architecture for Secure Internet of Things" , www.mdpi.com/journal/information, 2018.

[8]: M.M.Wong, M. L. D.Wong, C. Zhang and I.Hijazin, "Circuit and System Design for Optimal Lightweight AES Encryption on FPGA", IAENG International Journal of Computer Science, 45:1, IJCS_45_1_10, 2018.

[9]: R. Doomun, J.Doma and S.Tengur, "AES-CBC Software Execution Optimization", IEEE, **ISBN:** 978-1-4244-2327-9, 2008.