Efficient Implementation of Aes By Modifying S-Box

Vijay L Hallappanavar¹, Basavaraj P Halagali², Veena V Desai³
¹KLES’s College of Engineering & Technology, Chikodi, Karnataka
²V S M Institute of Technology, Nipani, Karnataka
³Gogte Institute of Technology, Belgaum, Karnataka

ABSTRACT: Security of data and messages are an imperative issue because of fast evolution of digital data exchanges over unsecured network. Data security is achieved by methods of cryptography, which deals with encryption of data. Standard symmetric encryption algorithms provide better security for the multimedia data. In the existing AES algorithm we are modifying S-Box based on some of the reference papers which we have gone through. After implementing modification the output is compared to original output, in terms of timing analysis, Hamming distance, Balanced output and Avalanche effect and so on. Hence the efficiency of modified AES is verified after the comparison.

Keywords: Avalanche effect, Hamming distance, Polynomial for S-box, Symmetric encryption, Swapping words in S-box

I. Introduction

AES belongs to secret key cryptographic algorithm. With secret key cryptography, a single key is used for both encryption and decryption. The sender uses the key (or some set of rules) to encrypt the plaintext and sends the cipher text to the receiver. The receiver applies the same key (or rule set) to decrypt the message and recover the plaintext. Because a single key is used for both functions, secret key cryptography is also called symmetric encryption. With this form of cryptography, it is obvious that the key must be known to both the sender and the receiver.

AES is a block cipher with a block length of 128 bits. AES allows for three different key lengths: 128, 192, or 256 bits. In this paper we have selected the key length of 128 bits. Encryption consists of 10 rounds of processing for 128-bit key, 12 rounds for 192-bit keys, and 14 rounds for 256-bit keys.

Except for the last round in each case, all other rounds are identical. Each round of processing includes one single-byte based substitution step, a row-wise permutation step, a column-wise mixing step, and the addition of the round key. The order in which these four steps are executed is different for encryption and decryption. To appreciate the processing steps used in a single round, it is best to think of a 128-bit block as consisting of a 4×4 matrix of bytes, arranged as follows:

\[
\begin{bmatrix}
\text{Byte0} & \text{Byte4} & \text{Byte8} & \text{Byte12} \\
\text{Byte1} & \text{Byte5} & \text{Byte9} & \text{Byte13} \\
\text{Byte2} & \text{Byte6} & \text{Byte10} & \text{Byte14} \\
\text{Byte3} & \text{Byte7} & \text{Byte11} & \text{Byte15}
\end{bmatrix}
\]

Therefore, the first four bytes of a 128-bit input block occupy the first column in the 4×4 matrix of bytes. The next four bytes occupy the second column, and so on. The 4×4 matrix of bytes is referred to as the state array. AES also has the notion of a word. A word consists of four bytes that is 32 bits. Therefore, each column of the state array is a word, as is each row. Each round of processing works on the input state array and produces an output state array. The output state array produced by the last round is rearranged into a 128-bit output block.

The Advanced Encryption Standard comprises three block ciphers, AES-128. AES has a fixed block size of 128 bits and a key size of 128,192 or 256 bits. Block size has a maximum of 256 bits, but the key size has no theoretical maximum. Encryption consists of 10 rounds of processing for 128 bit keys, 12 rounds of 192 bit keys and 14 rounds of 256 bit keys.
The cipher uses number of encryption rounds which converts plain text to cipher text. The output of each round is the input to the next round. The output of the final round is the encrypted plaintext known as cipher text. The input given by the user is entered in a matrix known as State Matrix. Following are the four steps used for encryption:

1.1 SubBytes Step
SubBytes, also known as Byte substitution are the first iterative step of the algorithm in each round. In the Sub Byte step, each byte in the matrix is recognized using an 8-bit substitution box. This substitution box is called Rijndael S-box. This operation provides the non linearity in cipher. The S-box used is derived from multiplicative inverse over GF(2^8), known to have good non-linearity properties. To avoid attacks based on simple algebraic properties, the S-box is constructed by combining the inverse function with an invertible affine transformation. The S-box is also chosen to avoid any fixed points (and so is a derangement), and also any opposite fixed points.

1.2 Shift Rows Step
The shift row step is performed on the rows of the state matrix. It cyclically shifts the bytes in each row by a certain offset. The first remains unchanged. Each byte of the second row is shifted one position to the left. Similarly, the third and fourth rows are shifted by two positions and three positions respectively. The shifting pattern for block of size 128 bits and 192 bits in the same.

1.3 Mix column Step
In the Mix column step, the four bytes of each column of the state matrix are combined using an invertible linear transformation. A randomly generated polynomial is arranged in 4 x 4 matrix. The same polynomial is used during decryption. Each column of the state matrix is XOR-ed with the corresponding column of the polynomial matrix. The result is updated in the same column. The output matrix is the input to Add Round Key.
1.4 Add Round Key

A round key is generated by performing various operations on the cipher key. This round key is XORed with each byte of state matrix. For every round a new round key is generated using some operations on the cipher key.

For decryption, each round consists of following four steps: 1) Inverse shift rows, 2) Inverse substitute bytes, 3) Add round key, 4) Inverse mix column. The third step consists of XORing the output of previous two steps with four words from key schedule. Note the differences between the order in which substitution and shifting are carried out in a decryption round vis-a-vis the order in which similar operations are carried out in an encryption round. The last round for encryption does not involve the “Mix column” step. The last round for decryption does not involve “Inverse Mix column” step.

I. MODIFICATION IN AES

The advance encryption standard (AES) algorithm considered to be the leader of this innovation. A large variety of approaches for modifying AES have been appeared so as to satisfy the varying criteria of different applications. Effort spent on evaluating AES security strength has been very limited, it is significant that the majority of researches at AES dealt with performance evaluation rather than with security evaluation. The design and implementation of S-box are representing a key step in the AES algorithm. So many efforts were emulated to redesign, reconstruct or renew the implementation of the S-boxes.

2.1 New S-box Construction

Four irreducible polynomials shown in figure 3 can be used to create new substitution tables (S-boxes). Fig. 4 and 5 describe the S-box and its inverse for the polynomial ($X^8 + X^6 + X^4 + X^3 +X+1$).

<table>
<thead>
<tr>
<th>No.</th>
<th>Polynomial</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.</td>
<td>$X^8 + X^6 + X^4 + X+1$</td>
</tr>
<tr>
<td>2.</td>
<td>$X^8 + X^6 + X^4 + X^2 +X^3 +X+1$</td>
</tr>
<tr>
<td>3.</td>
<td>$X^8 + X^6 + X^4 + X^2 +X+1$</td>
</tr>
<tr>
<td>4.</td>
<td>$X^8 + X^6 + X^4 + X^3 +X+1$</td>
</tr>
</tbody>
</table>

Fig 3 : Different polynomial used to create new S-boxes

Fig 4: S-box generated from the polynomial ($X^8 + X^6 + X^4 + X^3 +X+1$) Figure 5 : Inverse S-box generated from the polynomial ($X^8 + X^6 + X^4 + X^3 +X+1$)

2.1 Modified AES S-box mapping Calculation

In AES we are constructing S-box and inverse S-box using new polynomial instead of standard polynomial and for modification in AES we are modifying the S-box and Inv S-box by swapping each word of S-box and Inv_s-box generated by new polynomial.

2.1.1 Modification In S_Box Using New Polynomial
In standard AES we are using $X^8+X^4+X^3+X+1$ for generating S-box. In the modification we used $X^8+X^4+X^3+X+1$ polynomial equation and each word will be swapped by own word.

### III. RESULT ANALYSIS

#### 3.1.1 Hamming Distance:
The Hamming distance between two strings of equal length is the number of positions at which the corresponding symbols are different. In another way, it measures the minimum number of substitutions required to change one string into the other, or the minimum number of errors that could have transformed one string into the other.

![Bar graph for Hamming Distance](image)

**Fig 8**: Bar graph for hamming distance output table

#### 3.1.2 Balanced Output:
Balanced output is the output which has the number of 1’s & 0’s should be nearly same.

![Bar graph for Balanced Output](image)

**Fig 9**: Bar graph for Balanced output table

#### 3.1.3 Avalanche Effect:
The avalanche effect is evident if, when an input is changed the output changes significantly. In the case of quality block ciphers, such a small change in either the key or the plain text should cause a drastic change in the cipher text.
IV. CONCLUSION

Usually lightweight encryption algorithms are very attractive for multimedia applications. Luckily we have achieved through our research a fast light weight encryption algorithm to secure our multimedia data from unauthorized access. For the security of multimedia data, we have proposed a Modified AES with change in S-Box with gives better results than the standard AES. Theoretical analysis and experimental results of the achievement makes it very suitable for high rate and less overhead on the data. For all these compensation it is suitable for any large scale text and image transfer. From the results we can conclude that the output from modified AES is better than standard AES in terms Hamming Distance, Balanced Output and Avalanche Effect.

REFERENCES