INTERNATIONAL JOURNAL OF INNOVATIVE RESEARCH & DEVELOPMENT

ISSN 2278 – 0211 (Online)

Multiple Look-Up Based AES Encryption and Pair-Based Authentication Technique

Vinay Rathod
Department of Computer Engineering, PVG's COET, Pune, India
Yogesh Shete
Department of Computer Engineering, PVG's COET, Pune, India
Ritesh Divekar
Department of Computer Engineering, PVG's COET, Pune, India
Utkarsh Dhadge
Department of Computer Engineering, PVG's COET, Pune, India

Abstract:
The Advanced Encryption Standard (AES) and Authentication are using in a large scale of applications that need to protect their data and information. A normal implementation of AES may be vulnerable to a timing attack. To counter this attack and to increase processing efficiency at the cost of some storage we proposed a multiple lookup table (MLT) based AES implementation. Along with this we proposed a pair-Based authentication scheme which generates session password for every session and have resistance to various attacks.

Key words: *MLT-Multiple Lookup Table*

1. Introduction

Information Security is a primary concern for every communication system. Encryption and Authentication has an important role in data protection. The aging Data Encryption Standard (DES) can be broken with little effort. Therefore, the National Institute of Standards and Technology (NIST) published Advanced Encryption Standard (AES). AES is a symmetric block cipher which is based on the principle known as Substitution Permutation network (SP-network) which means there will be a series of linked mathematical operations in the block cipher algorithm.AES encrypts a data block of 128-bits which is fixed with three different key sizes 128,192,256 bits. But a normal implementation of AES may be vulnerable to a timing attack. To counter this attack and to increase processing efficiency at the cost of some storage, multiple lookup table (MLT) based AES implementation [1] can be used. In this proposed implementation 128-bit AES algorithm is implemented using four lookup tables, which are pre-generated from substitution box (s-box).Due to increase in processing efficiency, this implementation is also useful for encryption in mobile device where battery is the most critical resource.

At other part Authentication is serious issue for data protection. The common method used for authentication is the text password method. But it is vulnerable to various attacks like dictionary attacks, shoulder surfing, brute force and social engineering attack. Another technology for authentication is a graphical password, biometrics but both are having different disadvantages. There are number of methods for biometrics such as finger print, iris scan, facial recognition and signature. But major disadvantage of biometrics is that these systems can be costly. We proposed a new authentication scheme for authentication. This scheme authenticates the user by session passwords. Session passwords are valid for only one session because, for every session new session password is generated. This authentication scheme provides better security against dictionary and brute force attacks as password changes for every session.

2. A General AES implementation

The basic unit for processing in the AES algorithm is a byte, so the input bit sequence is first transformed into byte sequence. In the next step a 2-dimensional array of bytes (called the State) is built. The input and output for the AES algorithm each consist of sequences of 128 bits. The Secret Key for the AES algorithm is a sequence of 128, 192 or 256 bits. The 128-bit AES algorithm consists of ten rounds of encryption as well as decryption First the 128-bit key is expanded into eleven so-called round keys, each of

them 128 bits in size. Each round includes a transformation using the corresponding cipher key to ensure the security of the encryption.
After an initial round, during which the first round key is XO Red to the plain text (Add round key operation), nine equally structured rounds follow. Each round consists of the following operations:

- Substitute bytes: Uses an S-box to perform a byte-by-byte substitution of the block
- Shift Rows: A simple permutation
- Mix Columns: A substitution that makes use of arithmetic over GF (2^8).
- Add Round Key**:** A simple bitwise XOR of the current block with a portion of the expanded key.

The tenth round is similar to rounds one to nine, but the Mix columns step is omitted. Each stage is easily reversible. For the Substitute Byte, Shift Rows, and Mix Columns stages, an inverse function is used in the decryption algorithm. For Substitute byte operation inverse of S-box is used.

3. Multiple Lookup Table Based AES Encryption

The normal AES implementations have four operations i.e. Substitute bytes, Shift rows, Mix columns and Add round key. But the Multiple Lookup table based AES encryption consists only add round key operation.

3.1. Encryption

In 1 to 10 round Add round key operation is performed using four lookup table TE0, TE1, TE2, TE3. The Pseudo code for this implementation is given below,

AES encrypt (byte plaintext [16], byte cipher text [16], word RK [44])
Begin
Word W0, W1, W2, W3
W0 = RK [0] XOR plaintext [0, 3]
W1 = RK [1] XOR plaintext [4, 7]
W2 = RK [2] XOR plaintext [8, 11]
W3 = RK [3] XOR plaintext [12, 15]
K = 4
For round = 1 step 1 to 9
S [0, 3] = W0
S [4, 7] = W1
S [8, 11] = W2
S [12, 15] = W3
W0=TE0(S (0)) XOR TE1(S (5)) XOR TE2(S (10)) XOR TE3(S (15)) XOR RK (k + 0)
W1=TE0(S (4)) XOR TE1(S (9)) XORTE2(S (14)) XOR TE3(S (3)) XOR RK (k + 1)
W2=TE0(S (8)) XOR TE1(S (13)) XOR TE2(S (2)) XOR TE3(S (7)) XOR RK (k + 2)
W3=TE0(S (12)) XOR TE1(S (1)) XOR TE2(S (6)) XOR TE3(S (11)) XOR RK (k + 3)
K = K + 4
end for
S[0, 3] = W0
S[4, 7] = W1
S[8, 11] = W2
S[12, 15]= W3
ciphertext[0,3]=(TE2(S(0)) & 0xff000000) XOR (TE3(S(5)) & 0xff0000) XOR (TE0(S(10)) & 0xff00) XOR (TE1(S(15)) & 0xff) XOR RK[40]
ciphertext[4,7]=(TE2(S(4)) & 0xff000000) XOR (TE3(S(9)) & 0xff0000) XOR (TE0(S(14)) & 0xff00) XOR (TE1(S(3)) & 0xff) XOR RK[41]
ciphertext[8,11]=(TE2(S(8)) & 0xff000000) XOR (TE3(S(13)) & 0xff0000) XOR (TE0(S(2)) & 0xff00) XOR (TE1(S(7)) & 0xff) XOR RK[42]
ciphertext[12,15]=(TE2(S(12))&0xff000000) XOR (TE3(S(1)) & 0xff0000) XOR (TE0(S(6)) &0xff00) XOR (TE1(S(11)) & 0xff) XOR RK[43]

3.2. Decryption

In one to nine round Add Round Key operation is perform using lookup table TD0, TD1, TD2, TD3.
And in last round Add Round Key operation is perform using lookup table TD4.The Pseudo code for decryption is given below,

AES_encrypt (byte plaintext [16], byte cipher text [16], word RK [44])
Begin
Word W0, W1, W2, W3
W0 = RK [0] XOR ciphertext [0, 3]
W1 = RK [1] XOR ciphertext [4, 7]

W2 = RK [2] XOR ciphertext[8, 11]
W3 = RK [3] XOR ciphertext[12, 15]
K = 4
For round = 1 step 1 to 9
S [0, 3] = W0
S [4, 7] = W1
S [8, 11] = W2
S [12, 15] = W3
W0=TD0(S (0)) XOR TD1(S (5)) XOR TD2(S (10)) XOR TD3(S (15)) XOR RK (k + 0)
W1=TD0(S (4)) XOR TD1(S (9)) XORTD2(S (14)) XOR TD3(S (3)) XOR RK (k + 1)
W2=TD0(S (8)) XOR TD1(S (13)) XOR TD2(S (2)) XOR TD3(S (7)) XOR RK (k + 2)
W3=TD0(S (12)) XOR TD1(S (1)) XOR TD2(S (6)) XOR TD3(S (11)) XOR RK (k + 3)
K = K + 4
end for
S[0, 3] = W0
S[4, 7] = W1
S[8, 11] = W2
S[12, 15]= W3
plaintext[0,3]=(TD4(S(0)) & 0xff000000) XOR (TD4(S(13)) & 0xff0000) XOR (TD4(S(10)) &0xff00) XOR (TD4(S(7)) & 0xff) XOR RK[40]
plaintext[4,7]=(TD4(S(4)) & 0xff000000) XOR (TD4(S(1)) & 0xff0000) XOR (TD4(S(14)) &0xff00) XOR (TD4(S11)) & 0xff) XOR RK[41]
plaintext[8,11]=(TD4(S(8)) & 0xff000000) XOR (TD4(S(5)) & 0xff0000) XOR (TD4(S(2)) & 0xff00) XOR (TD4(S(15)) & 0xff) XOR RK[42]
plaintext[12,15]=(TD4(S(12))&0xff000000) XOR (TD4(S(9)) & 0xff0000) XOR (TD4(S(6)) &0xff00) XOR (TD4(S(3)) & 0xff) XOR RK[43]
The following figure shows MLT based 128-bit AES implementation.

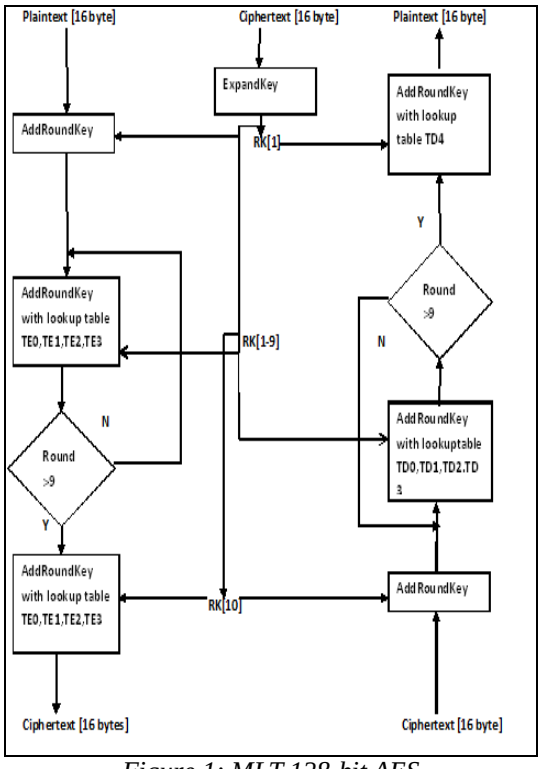

Figure 1: MLT 128-bit AES

4. Generation of Lookup Tables

Input**:**

State matrix consisting of elements ai,j and a round-key matrix consisting of elements ki,j.

State:

$$S[] = \begin{pmatrix} a0,0 & a0,1 & a0,2 & a0,3 \\ a1,0 & a1,1 & a1,2 & a1,3 \\ a2,0 & a2,1 & a2,2 & a2,3 \\ a3,0 & a3,1 & a3,2 & a3,3 \end{pmatrix}$$

Then the transformations can be expressed as follows.

Substitute Byte:bi,j= S[ai,j]

Shift Rows:

$$\begin{bmatrix} c_{0,j} \\ c_{1,j} \\ c_{2,j} \\ c_{3,j} \end{bmatrix} = \begin{bmatrix} b_{0,j} \\ b_{1,j-1} \\ b_{2,j-2} \\ b_{3,j-3} \end{bmatrix}$$

Mixcolumn:

$$\begin{bmatrix} d_{0,j} \\ d_{1,j} \\ d_{2,j} \\ d_{3,j} \end{bmatrix} = \begin{bmatrix} 02 & 03 & 01 & 01 \\ 01 & 02 & 03 & 01 \\ 01 & 01 & 02 & 03 \\ 03 & 01 & 01 & 02 \end{bmatrix} \begin{bmatrix} c_{0,j} \\ c_{1,j} \\ c_{2,j} \\ c_{3,j} \end{bmatrix}$$

Add Round Key:

$$\begin{bmatrix} e_{0,j} \\ e_{1,j} \\ e_{2,j} \\ e_{3,j} \end{bmatrix} = \begin{bmatrix} d_{0,j} \\ d_{1,j} \\ d_{2,j} \\ d_{3,j} \end{bmatrix} \oplus \begin{bmatrix} k_{0,j} \\ k_{1,j} \\ k_{2,j} \\ k_{3,j} \end{bmatrix}$$

In the Shift Rows equation, the column indices are taken mod 4.We can combine all of these expressions into a single equation:

$$\begin{bmatrix} e_{0,j} \\ e_{1,j} \\ e_{2,j} \\ e_{3,j} \end{bmatrix} = \begin{bmatrix} 02 & 03 & 01 & 01 \\ 01 & 02 & 03 & 01 \\ 01 & 01 & 02 & 03 \\ 03 & 01 & 01 & 02 \end{bmatrix} \begin{bmatrix} S[a_{0,j}] \\ S[a_{1,j-1}] \\ S[a_{2,j-2}] \\ S[a_{3,j-3}] \end{bmatrix} \oplus \begin{bmatrix} k_{0,j} \\ k_{1,j} \\ k_{2,j} \\ k_{3,j} \end{bmatrix}$$

$$= \left(\begin{bmatrix} 02 \\ 01 \\ 01 \\ 03 \end{bmatrix} \cdot S[a_{0,j}] \right) \oplus \left(\begin{bmatrix} 03 \\ 02 \\ 01 \\ 01 \end{bmatrix} \cdot S[a_{1,j-1}] \right) \oplus \left(\begin{bmatrix} 01 \\ 03 \\ 02 \\ 01 \end{bmatrix} \cdot S[a_{2,j-2}] \right)$$

$$\oplus \left(\begin{bmatrix} 01 \\ 01 \\ 03 \\ 02 \end{bmatrix} \cdot S[a_{3,j-3}] \right) \oplus \begin{bmatrix} k_{0,j} \\ k_{1,j} \\ k_{2,j} \\ k_{3,j} \end{bmatrix}$$

In the above equation, we are expressing the matrix multiplication as a linear combination of vectors. We define four 256-word (1024-byte) tables as follows.

$$T_0[x] = \begin{pmatrix} 02 \\ 01 \\ 01 \\ 03 \end{pmatrix} \cdot S[x] \quad T_1[x] = \begin{pmatrix} 03 \\ 02 \\ 01 \\ 01 \end{pmatrix} \cdot S[x] \quad T_2[x] = \begin{pmatrix} 01 \\ 03 \\ 02 \\ 01 \end{pmatrix} \cdot S[x] \quad T_3[x] = \begin{pmatrix} 01 \\ 01 \\ 03 \\ 02 \end{pmatrix} \cdot S[x]$$

Thus, each table takes as input a byte value and produces a column vector (a 32-bitword) that is a function of the S-box entry for that byte value. These tables can be calculated in advance.
We can define a round function operating on a column in the following fashion.

$$\begin{bmatrix} s'_{0,j} \\ s'_{1,j} \\ s'_{2,j} \\ s'_{3,j} \end{bmatrix} = T_0[s_{0,j}] \oplus T_1[s_{1,j-1}] \oplus T_2[s_{2,j-2}] \oplus T_3[s_{3,j-3}] \oplus \begin{bmatrix} k_{0,j} \\ k_{1,j} \\ k_{2,j} \\ k_{3,j} \end{bmatrix}$$

Similarly for decryption lookup table can be generated as,
TD0[x] = Si[x].[0e, 09, 0d, 0b];
TD1[x] = Si[x].[0b, 0e, 09, 0d];
TD2[x] = Si[x].[0d, 0b, 0e, 09];
TD3[x] = Si[x].[09, 0d, 0b, 0e];
TD4[x] = Si[x].[01,01,01,01];
Where,
Si [ ] = inverse substitution box

5. Pair-based Authentication Scheme

We proposed a new authentication scheme called as Pair-based Authentication Scheme [2]. In the course of registration, the user submits the secret pass. The minimum length of the secret pass is 8 and it should contain even number of characters. During the primary level authentication, when the user chooses the pair-based authentication scheme, an interface consisting of 6X6 grid is displayed. The grid contains both alphabets and numbers which are placed at random and the interface changes every time. The mechanism involved in the pair-based authentication scheme is as follows: Firstly, the user has to consider the secret pass in terms of pairs. The first letter in the pair is used to select the row and the second letter is used to select the column in the 6X6 grid. The intersection letter of the selected row and column generates the character which is a part of the session password. In this way, the logic is reiterated for all other pairs in the secret pass. Thereafter, the password inputted by the user i.e. the session password is now verified by the server to authenticate the user.

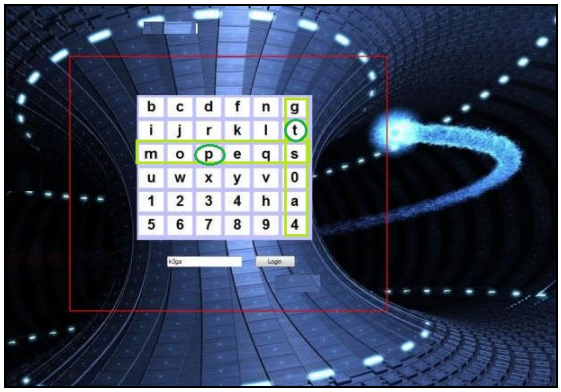

Figure 2

The first letter in the pair is used to select the row and the second letter is used to select the column. The intersection letter is part of the session password. This is repeated for all pairs of secret pass. Fig shows that S is the intersection symbol for the pair “PT”. The password entered by the user is verified by the server to authenticate the user. If the password is correct, the user is allowed to enter in to the system. The grid size can be increased to include special characters in the password.

6. Conclusion

The 4 lookup tables can not only combine the different steps of the round transformation so that reduce the code size, but also improve the performance efficiency. Due to increase in processing efficiency, this implementation is also useful for encryption and decryption in mobile device where battery is the most critical resource. It has a kind of simple style which can help readers study multiplication in GF (2^8).Along with this the new authentication technique generate session passwords and are resistant to dictionary attack, brute force attack and shoulder-surfing.

7. References

1. Landau, Susan. "Polynomials in the Nation's Service: Using Algebra to Design the Advanced Encryption Standard." American Mathematical Monthly 111.2 (2004): 89-117.JSTOR. Web. 7 Mar. 2013. http://www.jstor.org/stable/4145212
2. Z. Zheng, X. Liu, L. Yin, Z. Liu “A Hybrid password authentication scheme based on shape and text” Journal of Computers, vol.5, no.5 May 2010.
3. Courtois, Nicolas T., and Gregory V. Bard. "Algebraic Cryptanalysis of the Data Encryption Standard." Cryptography and Coding 4887 (2007): 152-69. Springer Link. Web. 7 Mar. 2013. <http://link.springer.com/chapter/10.1007%2F978-3-540-77272-9_10?LI=true#>