Applying transpose matrix on advanced encryption standard (AES) for database content

E B P Manurung¹, O S Sitompul² and Suherman*³
¹,² Department of Computer Science, Universitas Sumatera Utara, Indonesia
³ Department of Electrical Engineering, Universitas Sumatera Utara, Indonesia

*Email: suherman@usu.ac.id

Abstract. Advanced Encryption Standard (AES) is a specification for the encryption of electronic data established by the U.S. National Institute of Standards and Technology (NIST) and has been adopted by the U.S. government and is now used worldwide. This paper reports the impact of transpose matrix integration to AES. Transpose matrix implementation on AES is aimed at first stage of cyphertext modification for text based database security so that the confidentiality improves. The matrix is also able to increase the avalanche effect of the cryptography algorithm 4% in average.

1. Introduction

The most common threat on databases is the security lack on integrity, availability, and confidentiality so that the 4 (four) database security mechanisms should be implemented, namely: access control, inference control, flow control, and data encryption [1].

The access control on database is performed by requesting user password, but its content is not protected [2], some unsecured informations may be exposed [3]. Data encryption on databases may increase the security; however, the databases performance may decrease as it introduces additional processing delay [4]. As a matter of fact the speed of computer system increases, processing delay on encryption may not be an issue.

The Advanced Encryption Standard (AES) algorithm uses symetrical keys for encryption and description [6]. Research on adding encryption has been conducted by many researchers. The AES is one of the methods used in protecting database. For instance, Vishal and Patel [7] applied AES for cloud database, while Dawle et al. uses intrusion detection [8].

This paper modifies AES by using transpose matrix to enhance security when the plain text is altered before encrypted. Further, the transpose matrix insertion is also aimed at improving the avalanche effect so that the small change on plaintext results significant differences in cyphertext.

2. Research Method

The proposed approach for securing database content using AES algorithm is initialized by transposing the cypher text within the AES algorithm. The database contents are protected by the original AES as well as the modified one so that transpose matrix impact can be identified. Afterwards, the described information is analyzed.

The main parameter to be extracted from both method is the number of altered bits on cypher text that changed when the plain text is slightly changed. The next parameter is the percentage of avalanche effect. The higher avalanche effect the higher the security. The research flow is depicted in Figure 1.
In order to evaluate the method, My-SQL was employed and accessed by using php scripted web page. The network is simplified, and error correction due to network condition is guaranteed by the lower layer. The same evaluation runs for both AES and modified AES.

3. AES modification
Modification of AES is performed by implementing transpose matrix on MixColumns process on encryption and decryption process as shown in Figure 2.

![Figure 1. Research flow](image)

**Figure 1. Research flow**

![Figure 2. Transpose matrix insertion](image)

**Figure 2. Transpose matrix insertion**
In description process, transpose matrix is also applied to the opposite process which is InvMixColumn.

### 3.1 Transforming MixColumns dan InvMixColumns

MixColumns transformation on AES is performed by randomizing data in each array state column. Afterward, AddRoundKey performs XOR process between current state and round key. MixColumns process each state column. This operation converts column as polynomial on $GF(2^8)$ [6].

**Table 1. Multiplication matrix of MixColumns [5]**

| 2 | 3 | 1 | 1 |
|---|---|---|---|
| 1 | 2 | 3 | 1 |
| 1 | 1 | 2 | 3 |
| 3 | 1 | 1 | 2 |

The values on Table 1 are calculated by using the following equations:

$$S_{0,c} = ((02) \cdot S_{0,c}) \oplus ((03) \cdot S_{1,c}) \oplus S_{2,c} \oplus S_{3,c}$$

$$S_{1,c} = S_{0,c} \oplus ((02) \cdot S_{1,c}) \oplus ((03) \cdot S_{2,c}) \oplus S_{3,c}$$

$$S_{2,c} = S_{0,c} \oplus S_{1,c} \oplus (02) \cdot S_{2,c} \oplus ((03) \cdot S_{3,c})$$

$$S_{3,c} = ((03) \cdot S_{0,c}) \oplus S_{1,c} \oplus S_{2,c} \oplus (02) \cdot S_{3,c}$$

InvMixColumns treats each column as 4 variables of polynomials in the Galois field which are multiplied to $c(x)$ as $(x^4+1)$ $c(x)=11x^3+13x^2+9x+14$ [6]. The InvMixColumns transformation employs the same process as MixColumns as shown in Table 2.

**Table 2. Multiplication matrix of InvMixColumns [5]**

| 0E | 0B | 0D | 09 |
|----|----|----|----|
| 09 | 0E | 0B | 0D |
| 0D | 09 | 0E | 0B |
| 0B | 0D | 09 | 0E |

The following formulas are employed.

$$S_{0,c} = ((0E) \cdot S_{0,c}) \oplus ((0B) \cdot S_{1,c}) \oplus ((0D) \cdot S_{2,c}) \oplus ((09) \cdot S_{3,c})$$

$$S_{1,c} = ((09) \cdot S_{0,c}) \oplus ((0E) \cdot S_{1,c}) \oplus ((0B) \cdot S_{2,c}) \oplus ((0D) \cdot S_{3,c})$$

$$S_{2,c} = ((0D) \cdot S_{0,c}) \oplus ((09) \cdot S_{1,c}) \oplus ((0E) \cdot S_{2,c}) \oplus ((0B) \cdot S_{3,c})$$

$$S_{3,c} = ((0B) \cdot S_{0,c}) \oplus ((0D) \cdot S_{1,c}) \oplus ((09) \cdot S_{2,c}) \oplus ((0E) \cdot S_{3,c})$$

### 3.2 MixColumns and InvMixColumns modifications

MixColumns randomizes data in each array state column based on multiplication matrix [5]. MixColumns and InvMixColumns on AES are modified as shown in Table 3 and 4.
Table 3. Multiplication matrix modification of MixColumns

|    | 02 | 01 | 01 | 03 |
|----|----|----|----|----|
| 03 | 02 | 01 | 01 |
| 01 | 03 | 02 | 01 |
| 01 | 01 | 02 | 03 |

Table 4. Multiplication matrix modification of InvMixColumns

|    | 0E | 09 | 0D | 0B |
|----|----|----|----|----|
| 0B | 0E | 09 | 0D |
| 0D | 0B | 0E | 09 |
| 09 | 0D | 0B | 0E |

4. Results and Discussion

A sample of encryption process to the field of database by using modified AES is shown in Figure 3.

The analysis is performed by comparing encryption results of an original plain text and a single character change plaintext for both original AES and the modified one. By performing 10 rounds of encryption, the bit changes are shown in Figure 4.

Figure 3 MySQL database encryption

Figure 4. Bit changes as plaintext text changes
Conventional AES results bit changes up to 68 bits while the modified AES results up to 80 bits. In average the modified AES produce about 5 bit higher than the existing one.

![Avalanche Effect](image)

**Figure 5.** Avalanche effects

The avalanche effect of the modified AES are about 63% in average, while the standard one is 53% as shown in Figure 5. Avalanche effect increases as the transpose matrix is applied. It mean the security level is improved.

5. **Conclusion**

The application of transpose matrix to AES algorithm is successfully changing the plain text. When a single character in plain text is changed, significant bit changes occured. In average, the modification results in 5 bits higher and 4% higher avalanche rates compared to the original one.

5. **References**

[1] Rezk A, Ali H A and Barakat S I 2012 *International Journal of Computer Applications* **49**(19).
[2] Basharat I, Azam F and Muzaffar A W 2012 *Int. Journal of Computer Applications* **12**(47): 28-34.
[3] Al Khalil F, Gabillon A and Capolsini P 2014 *Foundations and Practice of Security* pp. 189-208.
[4] Singh P and Kaur P. 2015 *Int. Conference on Futuristic trend in Computational Analysis and Knowledge Management* (12): 353-358.
[5] Mahajan P and Sachdeva A 2013 *Global Journal of Computer Science and Technology* **13**(15).
[6] Bacon J, Eyers D, Pasquier T F M, Singh J, Papagiannis I, and Pietzuch P 2014, *IEEE Transactions on Network and Service Management* **11**(1): 76-89.
[7] Pancholi VR and Patel B P 2016 *International Journal for Innovative Research in Science and Technology* **2**(9):18-21.
[8] Dawle Y, Naik M, Vande S, and Zarkar N. *IJLERA*. 2017 2(03):01-6.