Indonesian Journal of Electrical Engineering and Computer Science
Vol. 11, No. 3, September 2018, pp. 942~948
ISSN: 2502-4752, DOI: 10.11591/ijeecs.v11.i3.pp942-948

Modified AES for Text and Image Encryption

Heidilyn V. Gamido[1], Ariel M. Sison[2], Ruji P. Medina[3]
[1,3]Technological Institute of the Philippines, Quezon City, Philippines
[2]Emilio Aguinaldo College, Manila Philippines

Article Info

Article history:

Received Feb 20, 2018
Revised Apr 21, 2018
Accepted May 27, 2018

Keywords:

Aes
Bit Permutation,
Encryption,
Mixcolumns

ABSTRACT

Advanced Encryption Standard (AES) is one of the most frequently used encryption algorithms. In the study, the Advanced Encryption Standard is modified to address its high computational requirement due to the complex mathematical operations in MixColumns Transformation making the encryption process slow. The modified AES used Bit Permutation to replace the MixColumns Transformation in AES since bit permutation is easy to implement and it does not have any complex mathematical computation. Results of the study show that the modified AES algorithm exhibited increased efficiency due to the faster encryption time and reduced CPU usage. The modified AES algorithm also yielded higher avalanche effect which improved the performance of the algorithm.

Corresponding Author:

Heidilyn V. Gamido,
Technological Institute of the Philippines,
Quezon City, Philippines.
E-mail: htvgamido@tsu.edu.ph

1. INTRODUCTION

The security of digital data is still one of the significant challenges in the increasing demand for the use of computers and for the need to communicate from one location to another [1]. Securing digital data is needed to protect the confidentiality, integrity, authenticity, and availability of data only to the intended recipient. Encryption provides a solution to ensuring the security of data before transmitting it over the network by encoding the data in a manner that is unreadable to the unauthorized parties and is decoded only by the authorized party [2].

Advanced Encryption Standard (AES) is one of most frequently used symmetric algorithms because of its combination of security, performance both in hardware and software, efficiency, and flexibility [3]. AES was established by National Institute of Standards and Technology (NIST) to replace DES [4] and 3DES and with a key length of 128, 192, and 256 bits. It takes a 128-bit data length and considers it as an array of bytes, and takes it as a 4x4 matrix called states. The number of rounds of AES depends on the key length [5] where the states are subjected to various transformations to encrypt the plaintext to ciphertext [6].

The AES is one of the commonly used encryption algorithms because of its simplicity and high efficiency [7]. However, it uses more processing power when compared to other algorithms [8], [9]. There are four transformations involved in AES namely: AddRoundKey, SubBytes, ShiftRows, and MixColumns and among these four transformations, the MixColumns have a higher computational load. MixColumns consists of having two arithmetic operations: multiplication and addition [10]. It is an expensive transformation because of the complex mathematical operations requiring computational resources in a software implementation of AES [11] making the encryption process slow [12], [13].

Replacing MixColumns using DES permutation table [14], multiple S-boxes [15], and shifting the column similar to ShiftRows [16] resulted in faster encryption time. However, security side has weakened due to low avalanche effect [15].

The research modified the standard AES by using bit permutation technique instead of the MixColumns Transformation. Bit permutation is a method that shuffles the bits of the states or rearranges the bits across the state [17], and bit permutation technique also alters the value of the state [18]. Like the MixColumns, bit permutation also provided diffusion in cryptographic algorithms [13]. The use of a permutation technique in an encryption algorithm achieved a minimum encryption time, decreased memory requirement [19] and is easy to implement [20].

2. MODIFIED AES ALGORITHM

Figure 1 shows the modification of AES using Bit Permutation. Bit Permutation does not have any complex mathematical computation but only involves shifting of the position of bits of every state. The modified AES algorithm also takes 128-bit data length as input and 128 key length which has ten rounds of transformation. For the last round, it follows the standard AES with only the SubBytes, ShiftRows and AddRoundKey Transformation.

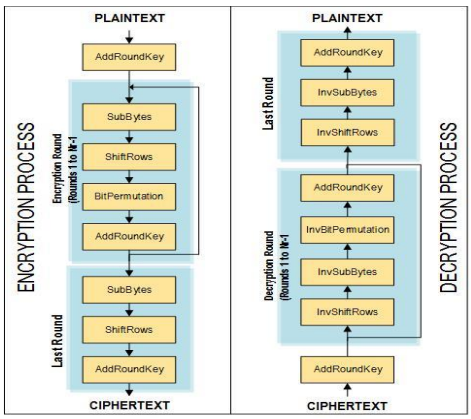


Figure 1. Modified AES Algorithm

2.1. Bit Permutation – Encryption Process

Bit Permutation or shuffling of bits for encryption process is as follows

1. We take the state value from ShiftRows Transformation starting from column 0. Column 0 has four rows, these are the states a(0,0), a(1,0), a(2,0), and a(3,0), as shown in Figure 2.
2. Each state from step 1 is composed of 8 bits, represented as ((x,0),b) where x value represents the row, 0 is column 0 and b represents the bit place in each state. Taking the number of bits per state results to a 4x8 matrix as shown in Figure 3.

$a_{0,0}$	$a_{0,1}$	$a_{0,2}$	$a_{0,3}$
$a_{1,0}$	$a_{1,1}$	$a_{1,2}$	$a_{1,3}$
$a_{2,0}$	$a_{2,1}$	$a_{2,2}$	$a_{2,3}$
$a_{3,0}$	$a_{3,1}$	$a_{3,2}$	$a_{3,3}$

Figure 2. Bit Permutation Step 1

$a_{0,0}$	(0,0),1	(0,0),2	(0,0),3	(0,0),4	(0,0),5	(0,0),6	(0,0),7	(0,0),8
$a_{1,0}$	(1,0),1	(1,0),2	(1,0),3	(1,0),4	(1,0),5	(1,0),6	(1,0),7	(1,0),8
$a_{2,0}$	(2,0),1	(2,0),2	(2,0),3	(2,0),4	(2,0),5	(2,0),6	(2,0),7	(2,0),8
$a_{3,0}$	(3,0),1	(3,0),2	(3,0),3	(3,0),4	(3,0),5	(3,0),6	(3,0),7	(3,0),8

Figure 3. Bit Permutation Step 2

3. The states from step 2 are then partitioned creating four 4x2 block matrix, labelled as block 0, block 1, block 2 and block 3, as shown in Figure 4.
4. The four blocks from step 3 are then transposed resulting in 2x4 transposed matrix, as shown in Figure 5.

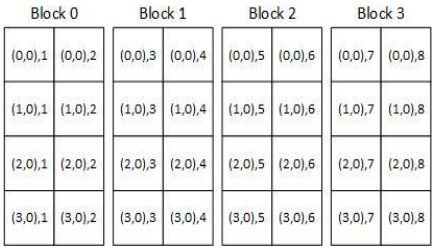


Figure 4. Bit Permutation Step 3

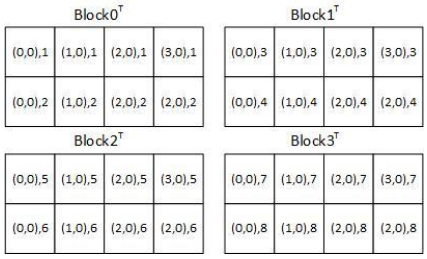


Figure 5. Bit Permutation Step 4

5. To get the new value at state a'(x,y), we need to do a row-wise concatenation of the bit values of the transposed block where x is the column_value considered in step 1, y is the block_value in step 4, as shown in Figure 6.

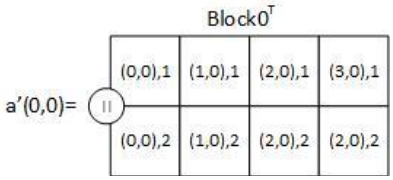


Figure 6. Bit Permutation Step 5

2.2. Inverse Bit Permutation – Decryption Process

The decryption process used Inverse Bit Permutation. The steps for the inverse bit permutation are as follows:

1. We take the state value from InvSubBytes Transformation starting from row 0. Row 0 has four columns, this are the states from a(0,0), a(0,1), a(0,2), and a(0,3), as shown in Figure 7.

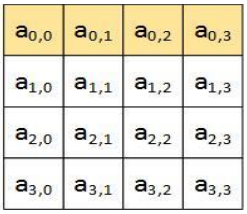


Figure 7. Inverse Bit Permutation Step 1

2. Each state from step 1 is composed of 8 bits, represented as ((0,y),b) where 0 is row 0, y represents the column and b represent the bit place in each state. Taking the number of bits per state results in a 1x32 matrix, as shown in Figure 8.

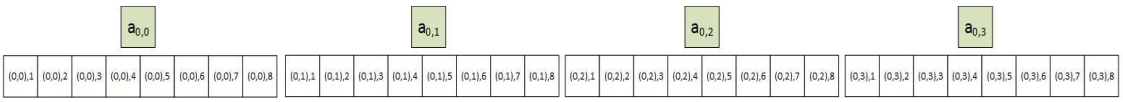


Figure 8. Inverse Bit Permutation Step 2

3. The 1x32 matrix from step 2 is then partitioned creating eight 1x4 block matrix labelled as block 0, block 1, block 2, block 3, block 4, block 5, block 6 and block 7, as shown in Figure 9.

Figure 9. Inverse Bit Permutation Step 3

4. Blocks 0 to 7 are then transposed creating eight 4x1 transposed matrix, as shown in Figure 10.

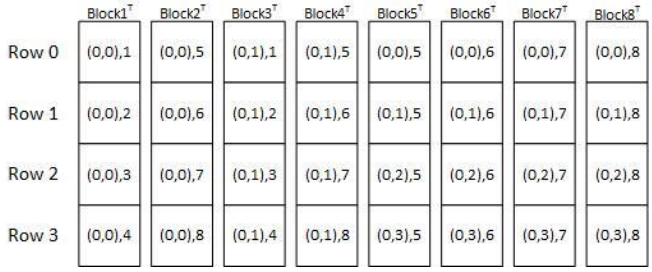


Figure 10. Inverse Bit Permutation Step 4

5. To get the new value at state a'(x,y), we need to do row-wise concatenation of the bit values of the transposed block, where x is the block_value in step 4 and y is the row_value considered in step 1, as shown in Figure 11.

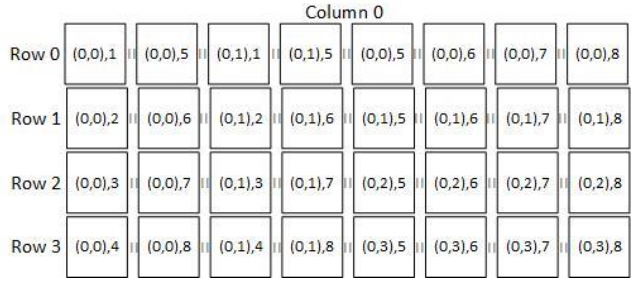


Figure 11. Inverse Bit Permutation Step 5

3. RESULTS AND ANALYSIS

Text files and images were encrypted to analyze the performance of the modified AES algorithm. During the experiments, different sizes of text files and images were tested for ten trials to get the average encryption time and CPU usage of the standard AES and modified AES. The standard and modified AES algorithm were both written in Microsoft. Net Framework and simulated on Core i5-2400 CPU @ 3.10 GHz with 4GB RAM and 64-bit Windows 7 OS.

Encryption Time: This refers to the time it takes for the algorithm to convert the plaintext to ciphertext.

CPU usage: This refers to the maximum utilized memory of the CPU during the execution of the algorithm.

3.1. Text file

Table 1 showed the comparison of the averaged encryption time and CPU usage of the standard and modified AES for encrypting text files. The result shows that the modified algorithm using bit permutation technique is faster by 100 ms and reduced CPU usage by at least 1%. Therefore, the modified algorithm increased the throughput and efficiency of the standard AES.

Table 1. Performance of Standard and Modified AES for Text Encryption

File Size (KB)	Encryption Time of AES (ms)	CPU %	Encryption Time of Modified – AES (ms)	CPU %
10	1151	13	979	11
20	1561	21	1481	20
30	2149	25	2056	24
40	2836	24	2732	24
50	3426	25	3380	25
60	4092	25	4017	25
70	4959	25	4838	25
80	5780	26	5619	25
90	6643	26	6432	26
100	7485	27	7327	26

Avalanche effect is a property of an encryption algorithm where a change in one bit of the plaintext produces a change in many bits of the ciphertext. The computation for the avalanche effect is:

$$Avalanche\ Effect = \frac{Number\_of\_bits\_flipped\_in\_ciphertext}{Number\_of\_bits\_in\_ciphertext} \quad (1)$$

Table 2 shows the result of avalanche effect of the standard and the modified AES. The tests were performed by changing one bit from the plaintext, either on the last, first or middle bit. Although the avalanche effect of an encryption algorithm not only depends on the complexity of the algorithm but also the key and plaintext have a certain impact, based on the result, it shows that the modified AES using bit permutation produced a higher avalanche effect than the standard AES. A high avalanche effect improves the level of the security of the algorithm [21].

Table 2. Avalanche Effect Result

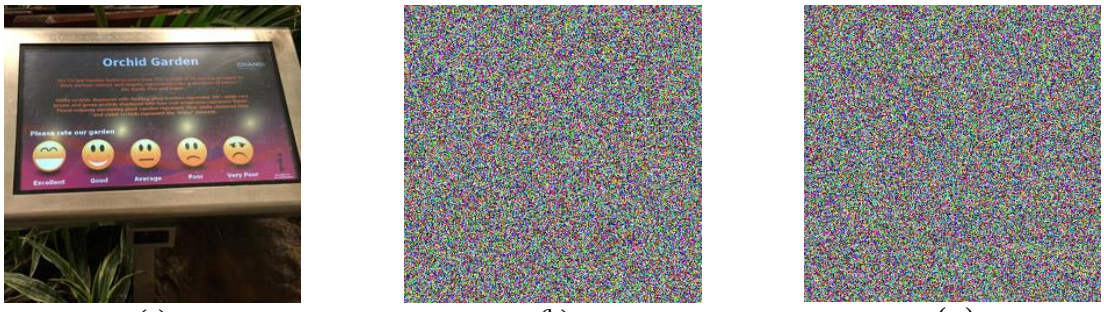
Plaintext	Ciphertext (AES)	%	Ciphertext (Modified - AES)	%
1111111111111111	d57f511479a3a2b6ada8837f15f8d73b	48.44	158c023b607a860c17717555ab25d380	57.81
1111111111111110	3c568184d212c7033be2c8e9ba6ea373	(62)	e5d70c50545f3d6182db80b309fc0457	(74)
1111111111111110	3c568184d212c7033be2c8e9ba6ea373	48.44	e5d70c50545f3d6182db80b309fc0457	59.38
0111111111111110	39a209f8a7ae0b112070ee6f3c924d2a	(62)	09bcce4828acf4964a247aed8285ceb5	(76)
1234567891011124	0bbb7c4ff6aa7fa9bd8952dd67c48cba	46.09	c4383d94ebd6efb68cdb801b6a937897	52.34
1234567891011120	fb0fa42b2a6e6528270a6a5852eb6579	(59)	91d1b650876854fffb27b9f06f51f213	(67)
A1B2C3D4E5F6G7H8	f83ac894fbe749877777c56bbb598e02	44.53	9bb1cad619c975fdb62b6688ac4cebb6	50.78
A1B2C3D4E5F6G7H9	a07a4ef38ff48422e51330784b914d4e	(58)	6aa93193890ef347eb884a4f793ddb42	(65)
9876543210987650	bfe737ac8c175c8e1db725640c1bde27	43.91	9557141fb253e980f7f37dc0b20b4e70	57.03
9876543210987652	2f3d860aee24a93178ea7f7a762a7294	(69)	7ac8a851bcb0cd5003211272e8618586	(73)
abcdefghijklmnop	7c52fb5fcf79667e4428b16060f9aeed	47.66	4b4a6e06e75d0f2542692a02802699fe	58.60
abcdefghijklmnoq	65f996097fbb6098de7e8ca0ba7b75b5	(61)	8274db79ebb83bda7fbead6f2691b5cc	(75)
abcdefghijklmnop	7c52fb5fcf79667e4428b16060f9aeed	54.69	4b4a6e06e75d0f2542692a02802699fe	64.01
abcdefgiijklmnop	4e0b3421913caf8dedbb0a4b38e8535d	(70)	9e74b0781b90d2caa548fde5ae95d397	(82)
zyxwvutsrqponmlk	83e7e14e35f48bc3a3c220eaf4a81bfb	43.75	2e2b9d7cbf4870c92aa8b0cbfd7b466d	52.34
zyxwwutsrqponmlk	3ab79816858e2f6b955883efe5af5d65	(56)	4f5750dc4e81be0d9daaed91339cb57c	(67)

3.2. Image

Figure 12 shows the encrypted images of the original image using the standard and the modified AES. Table 3 showed that the modified AES encrypted the original image faster by at least 1000 ms and has consumed lesser CPU resources as the image increases in size. A faster performance makes the modified algorithm suitable for encrypting images.

Table 3. Performance of Standard and Modified AES for Image Encryption

Image Size	Size on Disk (KB)	Encryption Time of AES (ms)	CPU %	Encryption Time of Modified AES (ms)	CPU %
256x256	47	20885.50	30.90	19139.90	27.80
512x512	100	160726.60	33.60	159554.30	32.00



(a) (b) (c)

Figure 12. (a) Original Image (b) Encrypted image using AES (c) Encrypted image using modified AES

4. CONCLUSION

The paper used bit permutation to replace the MixColumns Transformation of AES to improve the efficiency of AES. The paper compared the performance of the standard and modified AES algorithm by encrypting text files and images. The two algorithms were evaluated based on their encryption time, CPU usage and avalanche effect. Based on the results, the modified AES has increased the efficiency of the standard AES as it has faster encryption time in text files and images. The modified AES also consumed lesser CPU usage than the standard AES. During the testing, the modified algorithm also produced higher avalanche effect. Therefore, the use of the bit permutation in modified AES increased the efficiency, level of security and overall performance of the algorithm in encrypting text files and images.

For future work, we plan to implement the modified algorithm in the transmission of images across a network. Also, future work may extend the modified algorithm to audio and video standards like MPEG.

REFERENCES

[1] K. R. Saraf, V. P. Jagtap, and A. K. Mishra, "Text and Image Encryption Decryption Using Advanced Encryption Standard," *Int. J. Emerg. Trends Technol. Comput. Sci.*, vol. 3, no. 3, pp. 118–126, 2014.
[2] A. Makhmali and H. M. Jani, "Comparative Study On Encryption Algorithms And Proposing A Data Management Structure," *Int. J. Sci. Technol. Res.*, vol. 2, no. 6, pp. 42–48, 2013.
[3] H. O. Alanazi, B. B. Zaidan, A. A. Zaidan, H. A. Jalab, M. Shabbir, and Y. Al-Nabhani, "New Comparative Study Between DES, 3DES and AES within Nine Factors," *J. Comput.*, vol. 2, no. 3, pp. 2151–9617, 2010.
[4] S. El Adib and N. Raissouni, "AES Encryption Algorithm Hardware Implementation: Throughput and Area Comparison of 128, 192 and 256-bits Key," *Int. J. Reconfigurable Embed. Syst.*, vol. 1, no. 2, pp. 67–74, 2012.
[5] D. F. Garcia, "Performance Evaluation of Advanced Encryption Standard Algorithm," in *2015 Second International Conference on Mathematics and Computers in Sciences and in Industry (MCSI)*, 2015, pp. 247–252.
[6] P. Patil, P. Narayankar, Narayan D.G., and Meena S.M., "A Comprehensive Evaluation of Cryptographic Algorithms: DES, 3DES, AES, RSA and Blowfish," *Procedia Comput. Sci.*, vol. 78, no. December 2015, pp. 617–624, 2016.
[7] P. Kumar and S. B. Rana, "Development of modified AES algorithm for data security," *Optik (Stuttg).*, vol. 127, no. 4, pp. 2341–2345, 2016.
[8] R. Rejani and D. V Krishnan, "Study of Symmetric key Cryptography Algorithms," *Int. J. Comput. Tech.*, vol. 2, no. 2, pp. 45–50, 2015.
[9] P. V Kinge, S. J. Honale, and C. M. Bobade, "Design of AES Pipelined Architecture for Image Encryption / Decryption Module," *Int. J. Reconfigurable Embed. Syst.*, vol. 3, no. 3, pp. 114–118, 2014.
[10] M. R. Doomun, K. M. S. Soyjaudah, and D. Bundhoo, "Energy consumption and computational analysis of Rijndael-AES," *2007 3rd IEEE/IFIP Int. Conf. Cent. Asia Internet, ICI 2007*, 2007.
[11] K. Rahimunnisa, M. Priya Zach, S. Suresh Kumar, and J. Jayakumar, "Efficient techniques for the implementation of AES SubByte and MixColumn transformations," in *Advances in Intelligent Systems and Computing*, 2012, vol. 176 AISC, no. VOL. 1, pp. 497–506.
[12] S. Rehman, S. Q. Hussain, W. Gul, and Israr, "Characterization of Advanced Encryption Standard (AES) for Textual and Image data," *Int. J. Eng. Comput. Sci.*, vol. 5, no. October, pp. 18346–18349, 2016.
[13] N. Tyagi and Priyanka, "A Survey on Ensemble of Modifications on AES Algorithm," *J. Basic Appl. Eng. Res.*, vol. 1, no. 7, pp. 19–24, 2014.
[14] V. C. Koradia, "Modification in Advanced Encryption," *J. Information, Knowl. Res. Comput. Eng.*, vol. 2, no. 2, pp. 356–358, 2013.
[15] F. V Wenceslao, B. D. Gerardo, and B. I. T. Tanguilig, "*Modified AES Algorithm Using Multiple S-Boxes*," in Second International Conference on Electrical, Electronics, Computer Engineering and their Applications *(EECEA2015)*, 2015, vol. 5, no. 1, pp. 1–9.

[16] S. Anwarul and S. Agarwal, "Image enciphering using modified AES with secure key transmission," *Commun. Comput. Syst. - Prasad al.*, pp. 137–142, 2016.
[17] M. A. El-wahed, S. Mesbah, and A. Shoukry, "Efficiency and Security of Some Image Encryption Algorithms," *Proc. World Congr. Eng.*, vol. I, pp. 4–7, 2008.
[18] Y.-Q. Zhang and X.-Y. Wang, "Analysis and improvement of a chaos-based symmetric image encryption scheme using a bit-level permutation," *Nonlinear Dyn.*, vol. 77, no. 3, pp. 687–698, Aug. 2014.
[19] E. Thambiraja, G. Ramesh, and R. Umarani, "A Survey on Various Most Common Encryption Techniques," *Int. J. Adv. Res. Comput. Sci. Softw. Eng.*, vol. 2, no. 7, pp. 226–233, 2012.
[20] H. Ali-Pacha, N. Hadj-Said, A. Ali-Pacha, M. Mamat, and M. A. Mohamed, "An Efficient Schema of a Special Permutation Inside of Each Pixel of an Image for its Encryption," *Indones. J. Electr. Eng. Comput. Sci.*, vol. 11, no. 2, 2018.
[21] C. Dewangan and S. Agrawal, "A Novel Approach to Improve Avalanche Effect of AES Algorithm," *J. Adv. Res. Comput.*, vol. 1, no. 8, 2012.