Indonesian Journal of Electrical Engineering and Computer Science
Vol. 16, No. 2, November 2019, pp. 897~905
ISSN: 2502-4752, DOI: 10.11591/ijeecs.v16.i2.pp897-905

File encryption based on reduced-round AES with revised round keys and key schedule

Edjie M. De Los Reyes[1], Ariel M. Sison[2], Ruji P. Medina[3]
[1,3]Graduate Programs, Technological Institute of the Philippines (TIP), Philippines
[2]School of Computer Studies, Emilio Aguinaldo College (EAC), Philippines

Article Info

Article history:

Received Jan 25, 2019
Revised Apr 17, 2019
Accepted May 10, 2019

Keywords:

Algorithm
Avalanche effect
Randomness
Security
Throughput

ABSTRACT

The continuing advancement of technology had provided security issues in protecting the confidentiality of information. The need to protect unauthorized access of a third party is warranted. In this paper, the reduced-round modified AES with revised round keys and key schedule is proposed to ensure file confidentiality. The modifications to the AES cipher round was the reduction of the round iterations from 10 to 6, and additional key permutations were added in between states; while in the key schedule, additional byte substitution process was appended. Time and throughput were utilized to measure the performance of the application's encryption/decryption process; while the avalanche effect and randomness tests were used to measure the security of the modified AES algorithm. The results of evaluations have shown that the encryption and decryption time have improved by 1.27% and 1.21% respectively while the throughput has similarly improved by 1.29% and 3.19% for both encryption and decryption respectively. Whereas the avalanche effect of the modified AES was 50.06% which was more than the ideal value of 50% and it was also better than the standard AES which was 49.94% using the sample dataset. Finally, all the ciphertext outputs of the modified AES passed the randomness tests.

Corresponding Author:

Edjie M. De Los Reyes,
Graduate Programs, Technological Institute of the Philippines,
938 Aurora Blvd, Cubao, Quezon City, Philippines.
Email: emdelosreyes@tsu.edu.ph

1. INTRODUCTION

The exponential growth of information technology has immensely influenced the daily routines of people in all sectors especially in the use of the services provided by the internet [1]. The interconnectivity of devices continues to grow; it is projected that by 2020 50 billion devices will be connected to the internet [2-3]. This interconnection of devices will result in the enormous amount of data being transferred: raising security and privacy challenges [4-6]. The average cost of a data breach in a 2018 report is $3.86 million while the average cost per lost or stolen record is $148 [7]. Hence the need for a sufficient provision of security or protection from unauthorized access of confidential data or classified information is essential [8-10].

In cryptographic transformation, the plaintext is converted into an unreadable format called ciphertext utilizing an algorithmic process to safeguard privacy and integrity [11]. Other studies refer to cryptographic transformation simply as encryption. One form of cryptographic transformation is symmetric

encryption, wherein the transformation is reversible using a single secret key also called as a cipher key for both encryption and decryption. Encryption is the process of changing the plaintext into ciphertext, while decryption is the process of recovering the plaintext from the ciphertext. The current standard of symmetric encryption is the Advanced Encryption Standard (AES) based on the Rijndael algorithm. AES has different attractive advantages like high security, high throughput, and can be easily implemented both in hardware and software [3, 12]. However, with the development of new computing concepts that may weaken the strength of current and standardized cryptography, the need to strengthen and provide modifications and diversities in cryptography are being widely accepted [13]. Some studies have proposed modifications to AES: the replacement of mixcolumn transformation to bit permutation to speed up the encryption of texts and images was proposed in [14]. Whereas in [15], three modifications were proposed for encrypting high definition images, first a reduction in the use of mixcolumn from 10 to 5 rounds, second was the addition of mixcolumn in the key schedule algorithm, and third was the use of single S-box for both encryption and decryption.

This paper focuses on the development of a modified AES for file cryptographic transformation based on reduced-round with revised round keys and key schedule to secure confidential files that are either stored locally or to be sent through a network to prevent an unwanted third party from reading the confidential information [16-18]. The revised cipher round and key schedule of the modified AES aim to provide a better avalanche effect in the first three rounds and randomness of the ciphertext results [19] while the reduced round compensates the execution time due to the addition of operations in both the cipher round and key schedule algorithm. Moreover, the strength of the reduced-round modified AES (RRMA) will be tested against the standard AES using avalanche effect and randomness tests, while the performance of the proposed application will be evaluated in terms of speed and throughput.

2. RESEARCH METHOD

2.1. System Architecture

The file encryption was designed to secure a user's file deemed as confidential; the confidential files can be of any type and format. The system was developed using Microsoft Visual Studio 2015 and C# programming language.

The user will select a confidential file and supply the secret key that serves as the password for the file, and the application then prepares the file and the key for encryption by converting them to their hexadecimal equivalent. The hexadecimal values are then fed to the RRMA algorithm to transform the file into an unreadable and secured format. Subsequently, the encrypted file is saved with the same file extension as with that of the original file for storage or transmission.

To recover the original file, the user selects the encrypted file and inputs the appropriate secret key. The encrypted file and the key are then converted into their equivalent hexadecimal values and passed to the RRMA for decryption. The illustration of the file encryption process is shown in Figure 1.

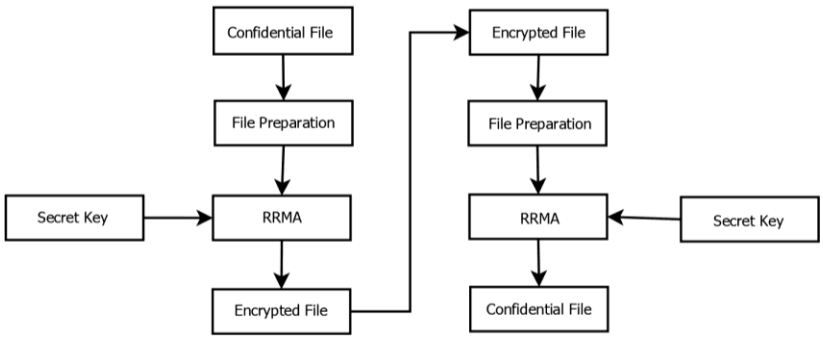

Figure 1. Block diagram of file encryption process

2.2. Reduced-Round Modified AES Algorithm

The algorithm used in the proposed file encryption was a modified AES. There were three modifications to AES: (a) additional add round key operations between the states of the standard AES cipher round ;(b) additional byte substitution operation and round constant addition in the key schedule algorithm before the key expansion process and; (c) reduction of the number of round iterations from 10 to 6.

Two operations were inserted in the AES cipher round: in the first five rounds, AddRoundKey using XOR function is placed after the SubBytes process, and ModAddRoundKey based on modulo addition is appended after the ShiftRows process. Another ModAddRoundKey is added after the SubBytes process in the last round. The RRMA cipher round is shown in Figure 2. The decryption will be the inverse of encryption processes: SubBytes to InvSubBytes; ShiftRows to InvShiftRows; MixColumns to InvMixColumns and; ModAddRoundKey to ModSubRoundKey. In the RRMA key schedule algorithm shown in Figure 3, the cipher key bytes are substituted using the AES s-box.

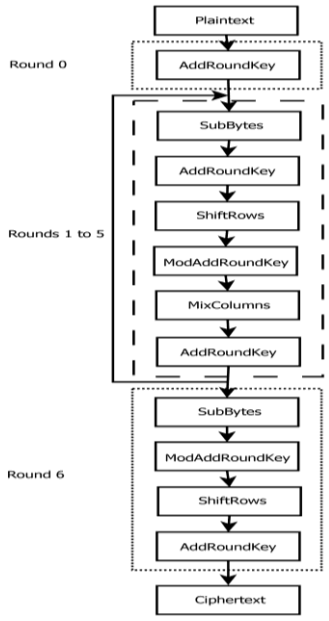

Figure 2. RRMA cipher round algorithm

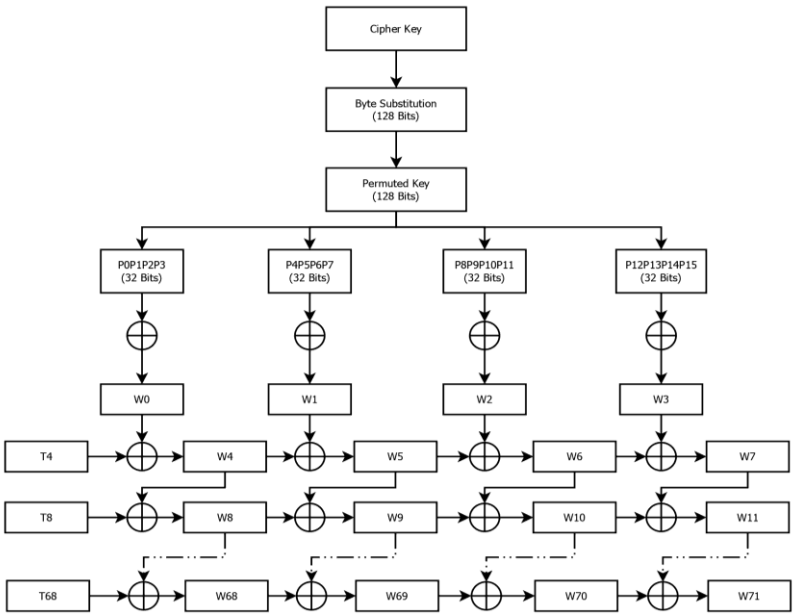

Figure 3. RRMA key schedule algorithm

The result is then divided into four words, where each word is XORed with a constant whose value, taken from the AES round constant, is dependent on a specific byte of the word. After the round constant addition, the following processes in the key expansion is the same as the standard AES [19].

2.3. Cipher Block Chaining Mode

AES is a 128-bit block algorithm, meaning the plaintext file is broken into blocks of 128-bit in length, and then each block is encrypted using the same secret key. The Cipher Block Chaining (CBC) mode is utilized to ensure that all blocks are uniquely encrypted. In this mode, a 128-bit initialization vector is XORed with the first block to be encrypted then the resulting ciphertext is XORed with the second block before being encrypted, the process is repeated until the last block. However, this key-chaining nature of CBC requires more processing time [20-22].

2.4. Metrics

The gauge of a suitable cryptographic algorithm is through diffusion and confusion [23-27]. Diffusion means that a small change in either the input plaintext or in the secret key will have a substantial effect in the ciphertext output and it is measured through avalanche effect [28]. Confusion, on the other hand, ensures that the ciphertext result gives no clue about the plaintext and it is measured by testing the randomness of the ciphertext [26-29]. The assumption of randomness is that the computed p-value should be > 0.01 [30].

The following defines the different tests that were used in the evaluation of the RRMA.

a) Avalanche Effect - Is the measure of the effect of a small change in either the input text or in the cipher key on the ciphertext. It is computed as the number of changed bit in the ciphertext over the total number of bits in the ciphertext.
b) Frequency Test within a Block (FTB) – This test determines the frequency of the number of ones in a block and checks whether it follows the assumption of randomness of one-half the size of the block.

c) Runs Test (RT) – The purpose of the test is to know whether the various lengths of an uninterrupted sequence of bits (whether ones or zeros) is as expected for a random sequence.
d) Binary Matrix Rank Test (BMRT) – The focus of this test is to examine for linear dependence among fixed length substrings of the original sequence.
e) Discrete Fourier Transform (Spectral) Test (DFT) – The purpose of the test is to determine whether the tested sequence has periodic features that deviate from the assumption of randomness.
f) Linear Complexity Test (LCT) – This test determines the complexity of a sequence based on the length of a linear feedback shift register.
g) Approximate Entropy Test (AET) – The test compares the frequency of overlapping blocks of two adjacent lengths against the expected result for a random sequence.

The performance of the file cryptographic transformation is analyzed using the metrics time and throughput [31]. The evaluation parameters are as follows:

a) Encryption Time – The required time to process the transformation from plaintext to its equivalent ciphertext and measured in milliseconds.
b) Decryption Time – The time to recover back the original plaintext from the ciphertext and measured in milliseconds.
c) Encryption Throughput – Defined as the speed of encryption and calculated by multiplying the size of the file by 1000 and dividing it with the encryption time in milliseconds.
d) Decryption Throughput – The calculated speed of decryption based on the size of the file and the decryption time.

2.5. Research Procedure

In the experiments, the secret keys used were "Th1sS3cr3tP@ssW!", and "TH1sS3cr3tP@ssW!". Different file types and sizes were also used in the experimentation as shown in Table 1.

Table 1. Sample Test Files

File	Type	Size (KB)
1	Text	21
2	Text	27
3	Text	28
4	Text	32
5	Text	52
6	Text	172
7	Document	15
8	Image	110
9	PDF	117
10	PDF	125

The hardware used for the experimentation is an Intel® Core™ i5-8250U CPU at 1.6 GHz with 8GB of RAM. Each sample test file is encrypted with the two cipher keys mentioned above, and the results were used to compute the avalanche effect. For the randomness tests, the ciphertexts resulting from using the key: Th1sS3cr3tP@ssW! were used.

In determining the time and throughput performance of the file cryptographic transformation, each sample file is encrypted and decrypted ten times. The average of these ten trials was taken as the encryption and decryption time respectively for each sample file. The average time is then used to compute the throughput of each file; the overall throughput of encryption/decryption was computed by taking the average of all the individual computed throughputs.

3. RESULTS AND ANALYSIS

The interface of the file cryptographic transformation application is shown in Figure 4. The interface showing the different command buttons are defined:

Load file button – this button when clicked prompts a new window for the user to browse and select the confidential file to either be encrypted or decrypted.

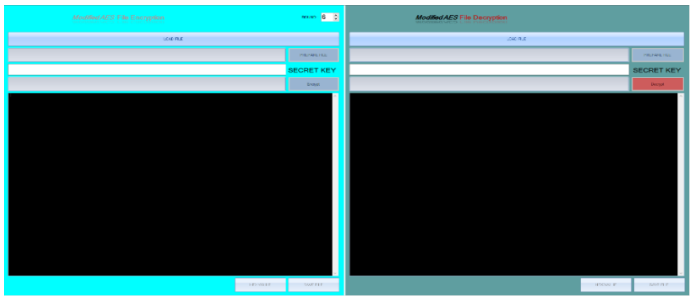

Figure 4. RRMA file encryption applications interface

a) Prepare file button – this will convert the selected file into its hexadecimal equivalent to be compatible with the cryptographic algorithm.
b) Secret key – the user supplies the secret key by the user that will serve as the master cipher key to be used in both the encryption and decryption.
c) Encrypt / Decrypt button –starts the file cryptographic transformation.
d) Save button – opens a new dialog box prompting the location where the file will be saved. The user enters the appropriate file name.
e) Hex Value button – shows the equivalent of the encrypted/decrypted file in the hexadecimal format used in the evaluation.

The output of the file cryptographic transformation can only be viewed on a text editor, but unreadable since the whole file is encrypted, and the intended reader based on its file extension cannot understand the formatting of the file. These are shown in Figure 5.

Figure 5. Viewing of cryptographic results

Ten (10) files were encrypted using the two cipher keys mentioned above using the standard AES algorithm, and the RRMA algorithm, portions of the ciphertexts for the first three files are shown in Table 2.

The ciphertext results of these samples were used to compute the avalanche effect and is shown as a graph in Figure 6.

Table 2. Sample Ciphertext Results

FILE	KEY 1: Th1sS3cr3tP@ssW!		KEY 2: TH1sS3cr3tP@ssW!	
	AES	RRMA	AES	RRMA
1	78275a507075795606 f2e30dfb2d788eb9d03 7d4554884d414cb0f0 29f408638	3232abdc3df4a42de326ff409 54cacfa48dc12da6aaeb29ca7b 4ada07deac650	309abe358174a495929eed3c 6c9cde81b061c5b1a4839481 2f9ca4dedb8f45dc	033728cd1615d29bf8b13133 c62742f464f63b5793ace763a 598aafaa950712d
2	bace3429adc7d2269a6 a6a31f0d876fbbac545 987b8bca3ffcd7f8206 b1d7134	33a420488dd9c9f818f512a32 9d7111dcb329dbab49c72d7bf 8d319761e20f73	f778fc1142ead0640efb719c3 e6355d941457c4d8d37b093f df9dc3b0f86c915	e6a2fd57569f88914b7c64112 40c623f76e82b9ae6064e2da7 4224e854a34301
3	c23a48d7f687b24875c 52d80aa390a1fa5e522 5e2a6e7da0d720eee81 1c632a4	37492206444350414b22c08e 5c0283683c5d83958d1fbd445 72d32ea40ccd6ee	507a1e2d469a7d0ed994bf43 234fafb575e725ba993e7a39b 9cbdfe9e0dafb85	a5282024ff83843a7b71b584e 9918926f874549db346b6c21 220655ec6039b51

Although the iterations of the RRMA has been reduced to 6 rounds only, it can be seen, based on these results that it is still better than the standard AES in terms of avalanche effect based on the sample data set, thus showing that RRMA has a good diffusion characteristic as a cryptographic algorithm. While in terms of the average avalanche effect of the ten samples, the modified AES got 50.06%, which is more than the ideal value of 50% [23, 32], while the standard AES got 49.94%, hence the modified AES is 0.25% better than the standard AES.

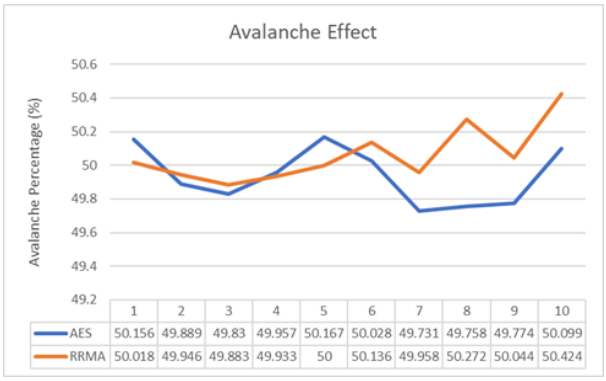

Figure 6. Avalanche effect results of RRMA vs AES

Another measure of a suitable cryptographic algorithm is its confusion characteristics that show the randomness of the output. Hence, the ciphertext results of both the AES and RRMA were also tested using 6 randomness tests namely frequency test within a block (FTB), runs test (RT), binary matrix rank test (BMRT), discrete Fourier transform (DFT), linear complexity test (LCT), and approximate entropy test (AET). In these tests, it is assumed that if the p-value > 0.01, the ciphertext string being observed is statistically random, otherwise it is non-random. The results of these randomness tests for RRMA are shown in Figure 7, while the results for AES are shown in Figure 8.

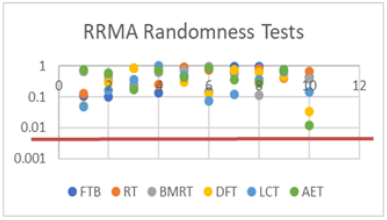

Figure 4. RRMA randomness tests results

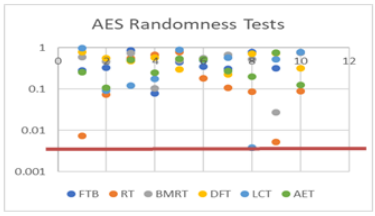

Figure 5. AES randomness tests results

The results of the six randomness tests show that RRMA is better than AES. It can be seen on the graph of Figure 8 that the computed p-values of all the tests on the sample data set are more significant than 0.01 indicating that the ciphertexts are random. Subsequently, the result of the randomness test on AES ciphertext results shows that there were three instances where the p-value is less than 0.01, as seen on the graph of Figure 9, indicating a non-random string. The AES has failed twice on runs test and once on linear complexity test.

The runs test result for AES shows that there were fewer oscillations or changes from ones to zeros or vice versa which is contrary to what is expected in a random sequence while the failure in the linear complexity test shows a variation in the observed frequency count from the expected values. The performance of the cryptographic application was also evaluated utilizing speed and throughput. The time taken to encrypt and decrypt a file was used as a metric for speed. The formula to compute for encryption or decryption time is illustrated in (1).

Elapsed time = End time – Start time (1)

The 10 sample files were encrypted for ten times. The average encryption time was taken for each sample and is shown in Table 3, and the unit is in milliseconds.

Table 3. AES vs RRMA Time Consumption

	Encryption time		Decryption time	
File	AES	RRMA	AES	RRMA
1	1183.6	1161.6	1037.2	1011.8
2	1396.5	1372.5	1213	1151.1
3	1448.9	1425.1	1255.6	1180.4
4	1608.9	1588.2	1388.1	1368.4
5	2383.2	2323.6	2010.5	1950.1
6	8818.9	8748.2	7539	7381.5
7	977	966.9	874.6	847.1
8	5283.5	5240.4	4536.1	4388.5
9	5651.1	5640	4872.1	4788.7
10	6077.7	6027.3	5211.2	5069.4

Based on these results, there is 1.27% improvement in the encryption time using the RRMA, even though additional operations were added in both the cipher round and key scheduling algorithms that have increased the processing time. This improvement in the encryption time is attributed to the reduction of the number of iterations in the cipher round.

To recover back the original file, the file to be decrypted is selected, and then the user clicks on the prepare key button to convert the file into its equivalent hexadecimal format. Subsequently, the user enters the corresponding secret key. Once the user clicks on the decrypt key, the decryption process commences, and the start time is recorded until the file has been fully decrypted and the end time is noted. The decryption time is then taken by getting the difference between the end time and the start time, as illustrated in (1). Ten (10) decryption trials were done for each sample file and the average decryption time was taken for each sample, the results are also shown in Table 3. It is observed from the results that decryption is faster than encryption and this is due to the cipher block chaining mode not being dependent on the previously decrypted plaintext.

The throughput is computed by multiplying the file size by 1000 and dividing it by the average encryption/decryption time in milliseconds. The unit of throughput is then kilobytes per second (KB/sec). The average throughput for both encryption and decryption of AES and RRMA are shown in Figure 9. There is 1.29% improvement in the encryption throughput of RRMA compared to AES and 3.19% in the decryption throughput.

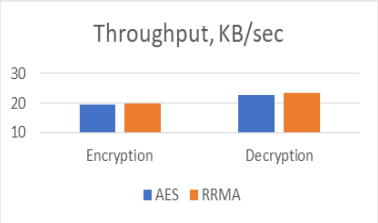

Figure 6. RRMA vs AES encryption/decryption throughput

4. CONCLUSION

This paper proposed a file encryption based on reduced-round modified AES with revised round keys and key schedule. The results have shown that it can successfully encrypt and decrypt different types of files. The enhancement has better cryptographic strength than AES as shown in the results of avalanche effect, and randomness tests and the speed and throughput have displayed better results as well. Hence, it can be noted that the additional operations that were used to modify the AES algorithm have improved the security of RRMA even with a reduced number of rounds.

The implementation of the application may be improved, by further investigation of the following: the block cipher mode implementation and the coding application of the decryption process to both promote parallelization to speed up the transformation and thereby improve the throughput efficiency.

REFERENCES

[1] A. Ray, A. Potnis, P. Dwivedy, S. Soofi, U. Bhade, and A. A. E. S. Algorithm, "XOR Operation , And Watermarking for Image Encryption," pp. 27-29, 2017.

[2] B. Wei, G. Liao, W. Li, and Z. Gong, "*A Practical One-Time File Encryption Protocol for IoT Devices*," Proc. - 2017 IEEE Int. Conf. Comput. Sci. Eng. IEEE/IFIP Int. Conf. Embed. Ubiquitous Comput. CSE EUC 2017, vol. 2, pp. 114-119, 2017.

[3] Anuradha, S. Kumar, A. Misra, and K. R. Krishna, "*Improved rapid AES for secure digital images*," 2017 Int. Conf. Energy, Commun. Data Anal. Soft Comput. ICECDS 2017, pp. 1429-1431, 2018.

[4] R. S. R. Vijayan, "*Cryptographic-Steganography Network Communication*," IEEE Int. Conf. Power, Control Signals Instrum. Efile///D/OneDrive - Tarlac State Univ. Pap. Ref., vol. 1, no. 3, pp. 694-697, 2017.

[5] Nurhayati and S. S. Ahmad, "*Steganography for inserting message on digital image using least significant bit and AES cryptographic algorithm*," Proc. 2016 4th Int. Conf. Cyber IT Serv. Manag. CITSM 2016, 2016.

[6] A. Desai, K. Ankalgi, H. Yamanur, and S. S. Navalgund, "*Parallelization of AES algorithm for disk encryption using CBC and ICBC modes*," 2013 4th Int. Conf. Comput. Commun. Netw. Technol. ICCCNT 2013, no. November 2001, 2013.

[7] Ponemon Institute LLC, "2018 Cost of Data Breach Study, Global Overview," *IBM Secur.*, no. July, pp. 1-47, 2018.

[8] S. Sruthi, A. Vijay, S. Jose, and V. Athira, "*Encryption & decryption of text file and audio using LabVIEW*," 2017 Int. Conf. Networks Adv. Comput. Technol. NetACT 2017, no. July, pp. 462-466, 2017.

[9] A. Alabaichi and A. I. Salih, "*Enhance security of advance encryption standard algorithm based on key-dependent S-box*," in 2015 Fifth International Conference on Digital Information Processing and Communications (ICDIPC), 2015, pp. 44-53.

[10] A. Nath, M. Santra, S. Maji, and K. F. Aleya, "*Bit Level Encryption Algorithm - Implementation of Bit-Wise Operations and Randomized Bit-Wise Columnar Transposition Method*," Proc.-2015 Int. Conf. Comput. Intell. Commun. Networks, CICN 2015, pp. 1057-1063, 2016.

[11] D. S. Kundi, A. Aziz, and N. Ikram, "A high performance ST-Box based unified AES encryption/decryption architecture on FPGA," *Microprocess. Microsyst.*, vol. 41, pp. 37-46, 2016.

[12] N. Dalakoti, N. Gaur, and A. Mehra, "*Hardware efficient AES for image processing with high throughput*," Proc. 2015 1st Int. Conf. Next Gener. Comput. Technol. NGCT 2015, no. September, pp. 932-935, 2016.

[13] H. Nejatollahi, N. Dutt, and R. Cammarota, "*Trends, challenges and needs for lattice-based cryptography implementations*," Proc. Twelfth IEEE/ACM/IFIP Int. Conf. Hardware/Software Codesign Syst. Synth. Companion - CODES '17, pp. 1-3, 2017.

[14] H. V. Gamido, A. M. Sison, and R. P. Medina, "Modified AES for text and image encryption," *Indones. J. Electr. Eng. Comput. Sci.*, vol. 11, no. 3, pp. 942-948, 2018.

[15] S. M. Wadi and N. Zainal, "High Definition Image Encryption Algorithm Based on AES Modification," *Wirel. Pers. Commun.*, vol. 79, no. 2, pp. 811-829, 2014.

[16] R. Das, "*Cumulative Image Encryption Approach based on User Defined Operation , Character Repositioning , Text Key and Image Key Encryption Technique and Secret Sharing Scheme*," 2017 IEEE Int. Conf. Power, Control. Signals Instrum. Eng., pp. 748-753, 2017.

[17] U. Farooq and M. F. Aslam, "Comparative analysis of different AES implementation techniques for efficient resource usage and better performance of an FPGA," *J. King Saud Univ.-Comput. Inf. Sci.*, vol. 29, no. 3, pp. 295-302, 2017.

[18] K. M. Akhil, M. P. Kumar, and B. R. Pushpa, "*Enhanced cloud data security using AES algorithm*," Proc. 2017 Int. Conf. Intell. Comput. Control. I2C2 2017, vol. 2018-Janua, pp. 1-5, 2018.

[19] E. M. De Los Reyes, A. M. Sison, and R. P. Medina, "Modified AES Cipher Round and Key Schedule," vol. 7, no. 1, pp. 146-146, 2018.

[20] A. Abidi, Q. Wang, B. Bouallègue, M. Machhout, and C. Guyeux, "*Quantitative evaluation of chaotic CBC mode of operation*," 2nd Int. Conf. Adv. Technol. Signal Image Process. ATSIP 2016, pp. 88-92, 2016.

[21] R. M. Awangga, N. S. Fathonah, and T. I. Hasanudin, "*Colenak: GPS tracking model for post-stroke rehabilitation program using AES-CBC URL encryption and QR-Code*," Proc.-2017 2nd Int. Conf. Inf. Technol. Inf. Syst. Electr. Eng. ICITISEE 2017, vol. 2018–Janua, pp. 255-260, 2018.

[22] S. Fahd, M. Afzal, H. Abbas, W. Iqbal, and S. Waheed, "Correlation power analysis of modes of encryption in AES and its countermeasures," *Futur. Gener. Comput. Syst.*, vol. 83, pp. 496-509, 2018.

[23] H. M. Hussien, Z. Muda, and Sharifah Md Yasin, "*Enhance The Robustness Of Secure Rijndael Key Expansion Function Based On Increment Confusion*," 6th Int. Conf. Comput. Informatics, no. 169, pp. 722-728, 2017.

[24] E. Yavuz, R. Yazıcı, M. C. Kasapbaşı, and E. Yamaç, "A chaos-based image encryption algorithm with simple logical functions," *Comput. Electr. Eng.*, vol. 54, pp. 471-483, 2016.

[25] M. Shakir, A. Bit Abubakar, Y. Bin Yousoff, and M. Sheker, "Improvement keys of advanced encryption standard (AES) Rijndael_M," *J. Theor. Appl. Inf. Technol.*, vol. 86, no. 2, pp. 216-222, 2016.

[26] N. Islam, Z. Shahid, and W. Puech, "Denoising and error correction in noisy AES-encrypted images using statistical measures," *Signal Process. Image Commun.*, vol. 41, pp. 15-27, 2016.

[27] N. Thi and T. Nga, "On the improving Diffusion layer and Performance of AES algorithm," no. October 2000, pp. 288-292, 2017.

[28] M. Hamdi, R. Rhouma, and S. Belghith, "A selective compression-encryption of images based on SPIHT coding and Chirikov Standard Map," *Signal Processing*, vol. 131, pp. 514-526, 2017.

[29] J. Chu and M. Benaissa, "Error detecting AES using polynomial residue number systems," *Microprocess. Microsyst.*, vol. 37, no. 2, pp. 228-234, 2013.
[30] L. E. Bassham et al., "A statistical test suite for random and pseudorandom number generators for cryptographic applications," no. April, 2010.
[31] T. F. G. Quilala, A. M. Sison, and R. P. Medina, "Modified blowfish algorithm," *Indones. J. Electr. Eng. Comput. Sci.*, vol. 11, no. 3, pp. 1027-1034, 2018.
[32] M. H.S., a. Raji Reddy, and M. T.N, "Improving the Diffusion power of AES Rijndael with key multiplication," *Int. J. Comput. Appl.*, vol. 30, no. 5, pp. 39-41, 2011.