Enhanced Security of Rijndael Algorithm using Two Secret Keys

Ibtihal Mohamed Abdullateef Fadul[1] and Tariq Mohamed Hassan Ahmed[2]

Department of Computer Science University of Khartoum Khartoum Sudan

[1]Ebtihal_ma_lateef@hotmail.com, [2]T_m71@yahoo.com

Abstract

Many ways to encrypt data have been innovated as the time moving, starting with simple ideas like changing positions of letters throughout to very high and complicated mathematical calculations to provide the element of security to our important data. Advanced Encryption Standard (AES) is one of the famous algorithms – specially Rijndael Algorithm- although there are other algorithms have been preferred upon rijndael algorithm in the field of security, rijndael algorithm has been selected as a symmetric cipher standard algorithm because of its high performance. So In this thesis there is an attempt to Enhance the security of rijndael algorithm by adding another key in both encryption and decryption operations to increase the strength of security and keep the performance close to traditional algorithm as much as possible using java programming language to implement the algorithm and monitoring the performance by calculating the execution time of encryption, decryption and generating key(s) in different numbers of sessions which had been opened between sender and receiver.

Keywords: *AES, Enhance, Performance, Rijndael, Security*

1. Introduction

Advanced Encryption Standard (AES) is "the United States Government's Federal Information Processing Standard for symmetric encryption", it is one of the famous algorithms which encrypts and decrypts different length of data blocks (AES 128, AES 192, AES 256) [1] using different length of keys (128bit, 192bit or 256bit) after choosing the length the key will be divided into sub-keys which will enter many rounds , determining the number of these rounds depend on both block size and key size (10 round, 12 round or 14 round), each sub-key response for encrypt or decrypt a block of data. The sender and receiver in AES use the same key to encrypt and decrypt data that called symmetric encryption so the key must be kept secured [12].

AES contain four separate blocks repeated for 10 rounds that mean AES based on re-encryption of blocks and mix them to give the cipher message.

Each block in AES passes through four primary processes which are:

1. SubBytes: as above the message division to block 4 × 4 byte, in this process each byte enter to S-box to replaced by other value.
2. ShiftRows: In this process the order of the block change by right to left byte shift except the first row and increase rate of one in each row.
3. MixColumns: After ShiftRow each column of block multiplied by fixed matrix to give new values in data block.
4. AddRoundKey: The last process is XORing between the message block and sub-key block, and the result is cipher block.

2. Related Work

Many published papers write about improving the performance of AES algorithm. Tanzilur Rahman and Shengyi Pan and Qi Zhang in Design of a High Throughput 128-bit AES paper provided the ability to doing multiple round in the same time, and increased the performance by provided multi chooses to process some round in algorithm [2], M. A. Matin in Performance Evaluation of Symmetric Encryption Algorithm in MANET and WLAN paper modified the block size (to be 200 bit); also as we know if we want to increase the security we make the key long to be stronger [3], Dr (Mrs) Pushpa R. Suri and Sukhvinder Singh Deora in Design of a modified rijndael algorithm using 2D rotation paper added a 5th steps to make the algorithm strong in security by change the data position in the data block [4], Priyanka Pimpale, Rohan Rayarikar and Sanket Upadhyay in Modification to AES Algorithm for complex encryption paper merged all round and execute them together in the same block at same time [5]. Sumira Hameed, Faisal Riaz ,Riaz Moghal, Gulraiz Akhtar, Anil Ahmed and Abdul Ghafoor Dar in Modified Advanced Encryption Standard For Text And Images paper took the advantage of DES algorithm which is the basic whom help to design the AES algorithm by using the Permutation step instead of MixColumns step [6].

Also Vishal Pachori, Gunjan Ansari and Neha Chaudhary in Improved Performance of Advance Encryption Standard using Parallel Computing paper used many computers and networks to apply steps of algorithm parallel [7]. Abdulkarim Amer Shtewi, Bahaa Eldin M. Hasan and Abd El Fatah .A. Hegazy in An Efficient Modified Advanced Encryption Standard (MAES) Adapted for Image Cryptosystems paper changed roles of ShiftRows step depend on the value of the first position on data block [8].

3. Proposed Work

In rijndael algorithm there are basic information to start both of encryption and decryption:

- Data Length determine between one of these lengths (128,192 or 256) bits.
- Key Length also based on these lengths (128,192 or 256) bits.
- Data length and key length must be same to encrypt or decrypt.
- Depend on data and key lengths we determine number of rounds in encryption and decryption operation (10, 12 or 14) round.
- The Key enter in expand process to distribute on rounds.
- Any all rounds except the last round there are four processes: (SubBytes, ShiftRows, MixColumns and AddRoundKey).
- In the last round we remove the MixColumns process and keep the others.

So in enhanced algorithm the above steps will be followed and another key will be added using the same previous lengths (128,192 or 265bits) just like first key, then XORing the new key (key no.2) with the data this called initial AddRoundKey, after that the traditional key (key no.1) will be expanding and dividing into sub-keys. Finally insert these sub-keys to the rounds. The same thing in decryption but by inversing the processes (Inv SubBytes, Inv ShiftRows and Inv MixCoulmns) and XOR the new key (key no.2) with cipher text, then XOR the output with the traditional sub-keys (key no.1) after entering the rounds to retrieve the plain text again. As the next figures illustrate:

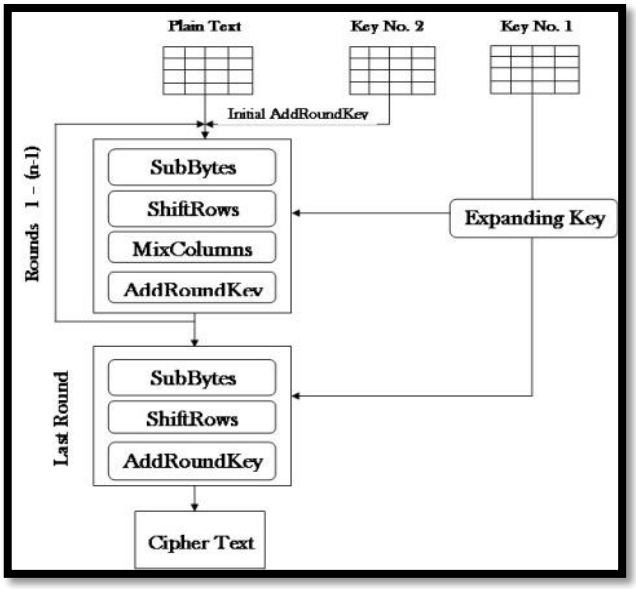

Figure 1. Encryption Flow Chart in enhanced Rijndael Algorithm

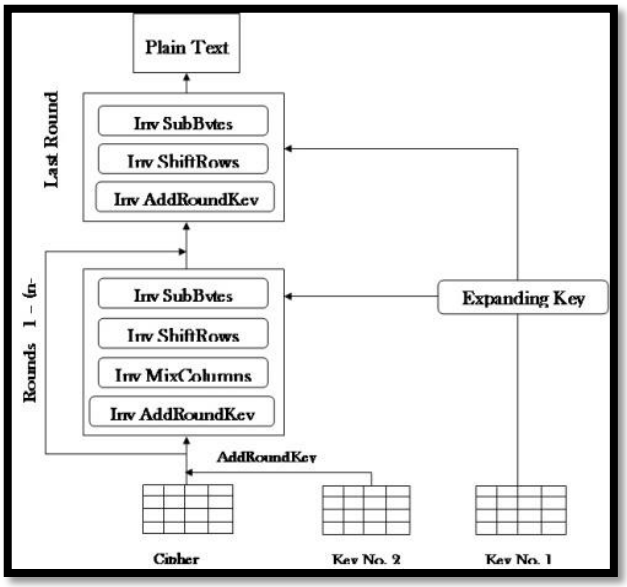

Figure 2. Decryption Flow Chart in Enhanced Rijndael Algorithm

4. Results

To prove the enhancement files with different sizes have been chosen (1 Kilo, 2Kilo, 4Kilo, 7Kilo, 10Kilo, 15Kilo, 20Kilo, 40Kilo, 70Kilo, 100Kilo and 200Kilo) to test both implementation (for encryption and decryption) and performance - by calculating the execution time in encryption and decryption in millisecond- using java programming language to impalement two algorithms (traditional and enhanced) and put the execution time in two tables to make the comparison between traditional and enhanced algorithms easier also to help in graphical drawing. The Table 1 and Figure 3 show the encryption operation:

Table 1. Executing Time to Different Files (Encryption)

Tests / Executing Time	Traditional algorithm (in millisecond)	Enhanced algorithm (in millisecond)
Test No2 (on 1 Kilo File)	2.99	2.85
Test No.3 (on 2 Kilo File)	1.99	2.13
Test No.4 (on 4 Kilo File)	1.42	1.85
Test No.4 (on 7 Kilo File)	1.99	1.85
Test No.4 (on 10 Kilo File)	1.42	1.56
Test No.4 (on 15 Kilo File)	18.10	18.10
Test No.4 (on 20 Kilo File)	1.85	1.56
Test No.4 (on 40 Kilo File)	1.71	1.42
Test No.4 (on 70 Kilo File)	1.71	1.71
Test No.4 (on 100 Kilo File)	17.10	17.25

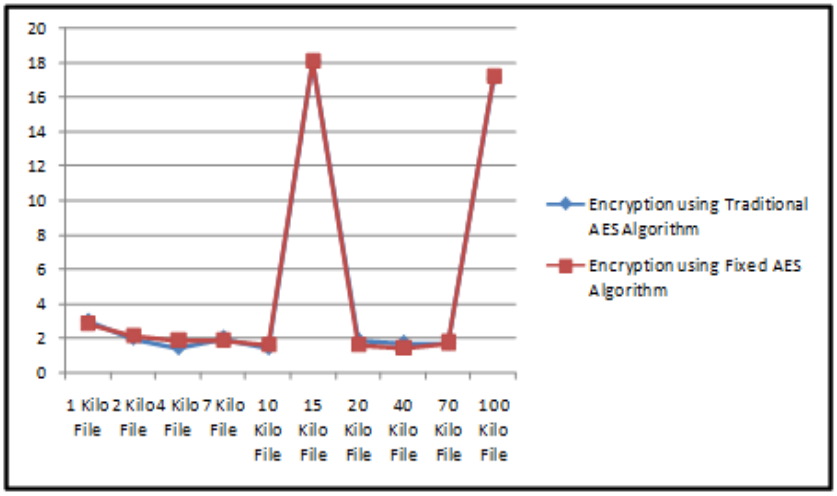

Figure 3. Encryption Tests from Traditional and Enhanced Algorithms

Table 2 and Figure 4 show the decryption operation:

Table 2. Executing Time to Different Files (Decryption)

Tests / Executing Time	Traditional algorithm (in millisecond)	Enhanced algorithm (in millisecond)
Test No2 (on 1 Kilo File)	6.41	6.84
Test No.3 (on 2 Kilo File)	5.98	6.13
Test No.4 (on 4 Kilo File)	5.84	5.84
Test No.4 (on 7 Kilo File)	5.70	6.13
Test No.4 (on 10 Kilo File)	5.70	4.99
Test No.4 (on 15 Kilo File)	28.22	28.37
Test No.4 (on 20 Kilo File)	5.56	5.98
Test No.4 (on 40 Kilo File)	5.70	5.56
Test No.4 (on 70 Kilo File)	3.13	2.99
Test No.4 (on 100 Kilo File)	24.95	25.23

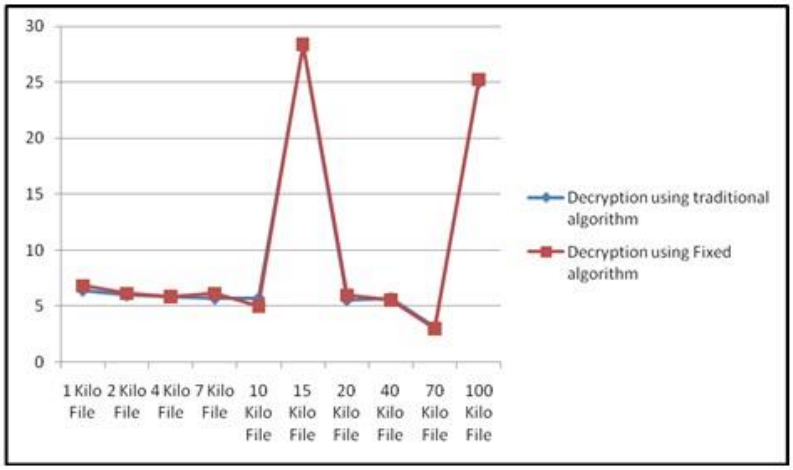

Figure 4. Decryption Tests from Traditional and Enhanced Algorithms

The last result is obtained by monitoring the performance of generating key in traditional Rijndael algorithm by calculates the execution time of generating key, this operation happened to the enhanced Rijndael algorithm as well just by summing two keys instead of one key, all of this done when a lot of sessions opened between sender and receiver, as it shows in next table:

Table 3. Results of Execution Time of Generating Key(s) in Different Sessions

Sessions / Executing Time	Numbers Of Keys	Traditional algorithm (in millisecond)	Number Of Keys	Enhanced algorithm (in millisecond)
1 session	1	26.09	2	38.78
3 sessions	3	48.33	6	95.09
7 sessions	7	103.50	14	410.04
13 session2	13	908.19	26	2661.86
19 sessions	19	1108.65	38	2091.42
25 sessions	25	981.33	50	2406.51
50 sessions	50	2334.08	100	2331.80
70 sessions	70	1602.53	140	3477.10
100 sessions	100	2156.58	200	2629.78
300 sessions	300	1915.34	600	8073.27
500 sessions	500	3200.56	1000	6758.026
700 sessions	700	3794.90	1400	7924.58
1000 sessions	1000	5148.39	2000	9720.31
3000 sessions	3000	10422.02	6000	21142.6
7000 sessions	7000	24569.69	14000	40900.08

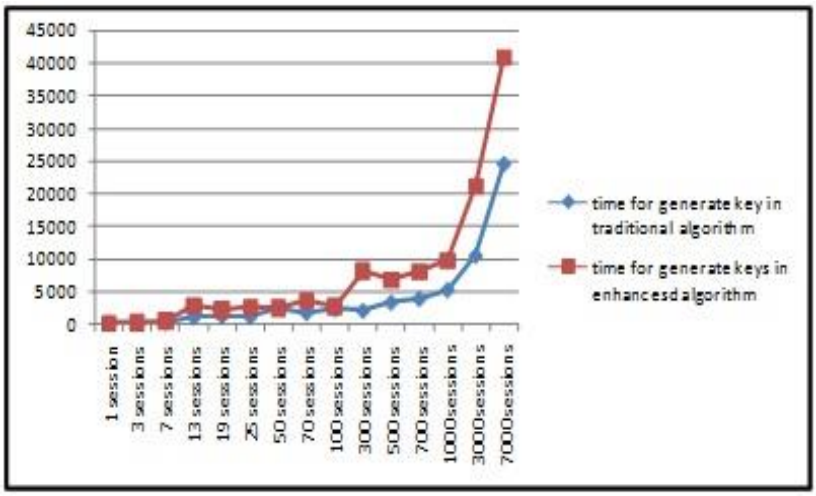

Figure 5. Differences between Generating Key Test

5. Discussion

After the success of Rijndael algorithm as symmetric cipher standard algorithm [9], it became in formally assessment state every 5 years [10] which help designers to discover any attack early and try to make the Algorithm stronger to stand against these attacks and make it active as long as possible. Both of brute force attack and square attack are most successful attacks face the Rijndael algorithm, the algorithm designers are the same who design square attack to test the algorithm and they come out with the

result that the square attack become fast from 6 rounds and less while the algorithm use from 10 to 14 rounds, after 6 rounds the brute force attack become faster but to get the right results from thousands of odds it takes a billion years [9]. But every time the algorithm become very complex and depends on analysis [11], which keeps the algorithm remains in the risk cycle. So the goal is to continually enhancing the algorithm to make it stronger than these attacks now and for future.

Rijndael algorithm has been enhanced by adding second key as it has been explained above and the implementation shows that if any bit changed within key, the half of cipher file will be changed that is called "On bit test" which is the simplest way to test the algorithm strength. Unfortunately other tests are much expensive and need more hardware and special environment to do them and get clear results.

AS a result every attack face a difficulty in break Rijndael algorithm when it use one key and it became more difficult and also take almost multiplier time trying to break it when it use a second key.

Finally, the last result has been extracted from the data shows that if opened sessions between sender and receiver become smaller, the differences between two algorithms will be closer in execution time of generating key/keys.

6. Conclusion

Using design, implementation and result analysis prove that the enhanced Rijndael algorithm can increase the security by adding second key to algorithm and keep the performance close to traditional algorithm, and there are some results which have been extracted:

1. The execution time of encryption and decryption operations in both of enhanced and traditional algorithms is close from each other, if the execution time of generating key/keys has been excluded from this calculation.
2. If sender and receiver use large number of opened sessions, the differences between execution times of generating key/keys will be larger.

Acknowledgements

Our thanks to all the staff of University of Khartoum and my special thanks to my sister Weaam Mohamed for her help.

References

[1] Intel®, Advanced EncryptionStandard (AES) New Instructions Set, **(2012)** September.
[2] T. Rahman, S. Pan and Q. Zhang, "Design of a High Throughput 128-bit AES (Rijndael Block Cipher)", **(2010)**.
[3] M. A. Matin, "Performance Evaluation of Symmetric Encryption Algorithm in MANET and WLAN", **(2010)**.
[4] P. R. Suri and S. S. Deora, "Design of a modified rijndael algorithm using 2D rotation", **(2011)**.
[5] P. Pimpale, R. Rayarikar and S. Upadhyay, "Modification to AES Algorithm for complex encryption", **(2011)**.
[6] S. Hameed, F. Riaz, R. Moghal, G. Akhtar, A. Ahmed and A. G. Dar, "Modified Advanced Encryption Standard for Text and Images", **(2011)**.
[7] V. Pachori, G. Ansari and N. Chaudhary, "Improved Performance of Advance Encryption Standard using Parallel Computing", **(2012)**.
[8] A. A. Shtewi, B. E. M. Hasan, A. E. Fatah and A. Hegazy, "An Efficient Modified Advanced Encryption Standard (MAES) Adapted for Image Cryptosystems", **(2010)**.
[9] E. Conrad, Advanced Encryption Standard.
[10] S. Merrion, Global Information Assurance Certification Paper, Rijndael-The Future of Encryption, **(2000-2002)**.

[11] E. Oswald, J. Daemen and V. Rijmen, "AES-The State of the Art of Rijndael's Security, **(2002)** October 30.
[12] Townsend security, AES encryption, **(2012)**.

Authors

Tarig Ahmed (PhD in computer science) is an assistant Professor at Dept. Computer sciences, University of Khartoum. His main interested research areas: Mobile agent system architecture, Mobile Agent Security, Distributed Systems, database Systems and artificial intelligent systems.

Ibtihal Mohamed Abdullateef Fadul was born in Kingdom of Saudi Arabia in November 1990. She's a graduate of Sudan University of Science and Technology 2010 and a Master student in University of Khartoum 2012.