Switching between the AES-128 and AES-256 Using Ks * & Two Keys

Moceheb Lazam Shuwandy, Ali Khalil Salih, Firas layth Khaleel Alameen, and Adib M.Monzer Habbal

University Utara Malaysia: Malaysia \ Kedah \ Changlon \ Sintok \UUM\CAS college
Tikrit University: Iraq \ Salah Din \ Tikrit \ Alqadisiya \ TU \ college of Computer Science and Math

Summary
This paper proposes a method to switch between the AES−128 and the AES−256 encrypters by using special key; that is to determine the process sequence. To encrypt the information, the user must provide the two keys (K1,K2) of the recipient as well as the message to be encrypted. This method can make it reduces a period of the time, especially in saving the hardware resource in implementing the AES 256 bit module and AES 128 bit module. Most designed modules can be used for both AES encryption and decryption. Besides, the architecture can still deliver a high data rate in both encryption and decryption operations. Therefore, we take advantage of the power from AES−265, and speed from AES−128. Therefore, at least one complexity by three to increase the speed, and speed will increase at least one by three of complexity. The proposed architecture is suited for hardware−critical applications and Network applications, such as using the chatting to exchange the secret messages, etc.

Key words:
Switching, Ks, SDA, K1,K2.

1. Introduction

In January 1997, the National Institute of Standards and Technology (NIST) announced the initiation of an effort to develop the AES and made a formal call for algorithms on September 12, 1997. After reviewed the results of this preliminary research, the algorithms MARS, RC6TM, Rijndael , Serpent and Twofish were selected as finalist . And further reviewed public analysis of the finalist, NIST has decided to propose Rijndael as the Advanced Encryption Standard (AES). It is expected to replace the DES and Triple DES so as to fulfill the stricter data security requirement because of its enhanced security levels.
Besides, an ASIC solution of AES is also required, because it can be more secure and consumes less power than that implemented by software.
In this paper, we used two ways to encrypt and decrypt the AES 256 bit and AES 128 bit . By comparing it with other designs, we can find that it yields safe the time and keep the power of the key 256 bit in good security when hardware used for both AES encryption and AES decryption so, this method is called **MOLAZ**** method.
The random key for switching between the two methods (K_s) when we enter the two keys of AES−128 and AES−256 each one using by its algorithm (see Fig.1). K_s is generated randomly by the Keygen as shown below:

K_s=kygengenerat(10000);

Except if K_s =0000,… or 9999 not all even or all odd.

The system do the algorithm AES−128 if the key is even numbers {0,2,4,6,8} and algorithm AES−256 if the key is odd numbers {1,3,5,7,9} ,and by using the SDA (System to determine the algorithm) system to determine the number of times (n0 and n1) each algorithm are used . A total number of times used this way is n=4 that is mean 4 times for all methods, and send to each algorithm K_{si} to determine the sequence, see Fig. 1.

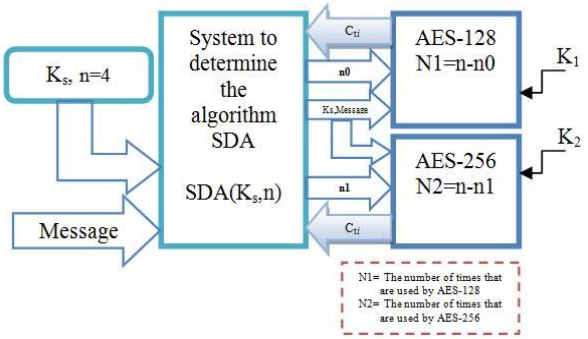

Fig. 1: the SDA using K_s and the message

2. K_s generation

To meaningful time period we must use 128 bit keys since whenever, increased the number of bits increased period of time and less speed. We use the AES−128 encryption to obtain to cipher text when the switch (K_s) has the even

Manuscript received August 5, 2010
Manuscript revised August 20, 2010

numbers 0,2,4,6 and 8, and AES−256 has the odd numbers 1,3,5,7 and 9.

For Example: if K_s=4615, i=0, see Table 1.

The sequence of operations	K_{si}	*i*
AES−128	4	0
AES−128	6	1
AES−256	1	2
AES−256	5	3

Table 1: show the sequence of algorithms

If K_s =0000,…or 9999 , nothing to do if all numbers odd or even; Since the benefit of this operation to using the two algorithms in the message.

Ks=kygengenerat(10000);
Except if K_s =0000 ,….or 9999
The flowchart in Fig. 3 shows the code below:
if(checkallodd (ks,4)==true || checkalleven (ks,4)==true)
ks=kygengenerat(10000);

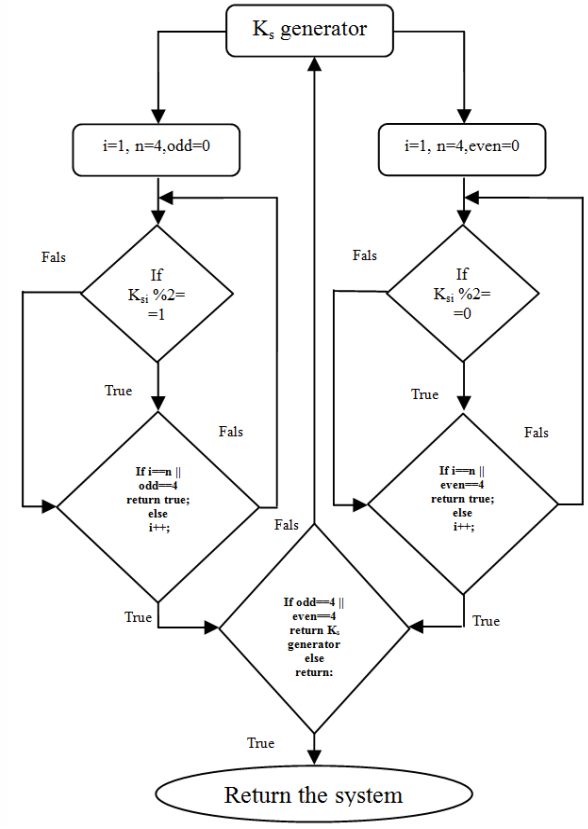

Fig. 3: Flow Char show check K_s elements

Convert the K_s to string array then send it to SDA method.

3. SDA method

The array of the key switching (K_s) enter this method with the message, therefore, the system check how many times each algorithm call to it and the sequence for ciphering the message. The message is divided into four sections m_i={m1,m2,m3,m4} as four arrays and will be determined by the key which algorithm will work on it and then add each number to the beginning of array message see Fig. 2. Send the array to its algorithm AES−128 or AES−256.

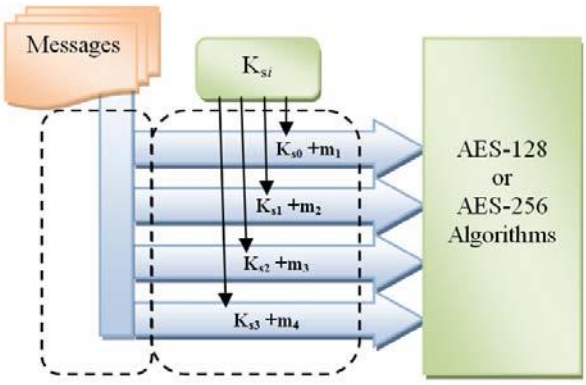

Fig. 2: SDA system sends K_s and m_i to algorithms

4. AES SYSTEM

4.1. AES General operations en/decryption

An AES system is a symmetric−key system in which the sender and receiver of a message share a single, common key, which is used to encrypt and decrypt the message. The data length of a key or message may be chosen to be any of 128 or 256 bits. The AES encryption/decryption algorithms are shown in Table 2.

AES operates on a 4x4 array of bytes (referred to as "state"). The algorithm consists of performing four different simple operations. These operations are (shown in Table 2):

•SubBytes
•ShiftRows
•MixColumns
•AddRoundKey

SubBytes perform byte substitution which is derived from a multiplicative inverse of a finite field. ShiftRows shifts elements from a given row by an offset equal to the row number. The MixColumns step transforms each column using an invertible linear transformation. Finally, the

AddRoundKey step takes a 4x4 block from a expanded key (derived from the key), and XORs it with the "state".

AES is composed of four high - level steps. These are:

1.Key Expansion
2.Initial Round
3.Rounds
4.Final Round

The Key Expansion step is performed using Rijndael's key schedule. The Initial Round consists only of an AddRoundKey operation. The Rounds step consists of a SubBytes, ShiftRows, MixColumns, and an AddRoundKey operation. The number of rounds in the Rounds step varies from 10 to 14 depending on the key size. Finally, the Final Round performs a SubBytes, ShiftRows, and an AddRoundKey operations.

Decryption in AES is done by performing the inverse operations of the simple operations in reverse order. However, as shown later on in this paper, because of the block cipher mode of operation used, decryption was implemented but never used .

Table 2. AES encryption/decryption algorithm.

AES Decryption	***AES Encryption***
InvAddRoundKey	AddRoundKey
for round=1 to Nr−1	for round=1 to Nr−1
InvShiftRows	SubBytes
InvSubBytes	ShiftRows
InvAddRoundKey	MixColumes
InvMixColumes	AddRoundKey
end for	end for
InvShiftRows	SubBytes
InvSubBytes	ShiftRows
InvAddRoundKey	AddRoundKey

4.2. MOLAZ encryption

This operation begin when SDA system decided the sequence and the number of times do the algorithm ,after the system request to the algorithm to do the operation of encryption and send the data m_i the algorithm reply new array $C_t[i][\ ]$, i=1,2,3,4 ; since the message is split to four parts m_1,m_2,m_3 and m_4.
For example:
K_s [i]={9,6,4,1}, i=1,2,3,4
SDA send K_{s1}=9 that is mean AES−256 method in the sequence 1 and m_1 of total operations m_i.
m_1={'m','o','c','e','h','e','b'}
in Hexadecimal
m_1={ 6d,6f,63,65,68,65,62}
m_1 means the part one of the message, since i of K_{si} =1 so the sequence is one.

AES−256 encrypt the message one with using the key K_1 256 bit and reply to SDA the array of cipher text1,
K_1 = "my name is moceheb"

C_{temp}[16]=
{ D721,753A,B106,5324,7C35,57FD,28B8,50A7,E293,B85E,C889,9A93,E774,399D,73A0,387A }

$C_t[1][i]=C_{temp}[i]$, i=1 to 16

Then, the number of K_s=9 added to the encryption operation to begin of the C_{temp} array to the total array C_t and we added new random numbers before send C_t to receiver; like "A78" with K_s =9 the result is "9A78" , that means we must remove the first four numbers from each array of C_t before the decryption method.

$C_t[1][i+1]$={9A78}, j= i+1=17
m_1:C_t[**1**][**j**]={ **9**A78,D721,753A,B106,5324,7C35,57FD,28B8,50A7,E293,B85E,C889,9A93,E774,399D,73A0,387A }

This operation is apply to all the messages according to K_{si}={9,6,4,1}, i=1,2,3,4 :

i=2,K_{s2}=6 and m_2→ AES−128
m_2:C_t[**2**][**j**]={**6**DFF,….}

i=3,K_{s3}=4 and m_3→ AES−128
m_3:C_t[**3**][**j**]={**4**35A,….}

i=4,K_{s4}=1 and m_4→ AES−256
m_4:C_t[**4**][**j**]={**8**F65,….}

Finally, the array sends back to the SDA system in order to collect all the arrays in one array as above C_t and then the user of the system send the cipher text to the recipient.

4.3. MOLAZ decryption

The recipient receives the encrypted text or cipher text and enters it into the SDA system where reverse the operation –deciphering, and according to the following steps:

(i) He must know the two keys K_1 and K_2.
(ii) Split the C_t array into 4 array C_t[4][n].
(iii) SDA system replaces the first four numbers of each array $C_t[i][j]$,i=1 to 4 and j=1 to 4, then add just the first number of each four numbers to K_{si} array see Fig. 4.
(iv) Send the value key switch K_{si}, number of sequence i of the operations and the cipher text C_i to the algorithm AES−128 or AES−256 according to the sequence i and K_{si} see Fig. 5.

(v) Use the K_1 for AES−128 and K_2 for AES−256 to produce the decryption text.

(vi) Finally, all the decryption texts added to the one array $m_t[i]$, i=0,1,…,n, n>=0 see Fig. 5.

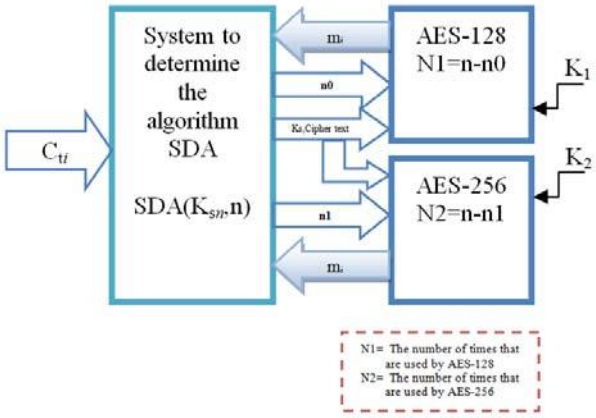

Fig. 4: the SDA using K_s and the cipher text Cti

5. Objectives

a. Depend on AES−256 Method to give more complexity to the system.

b. If complexity is low so the speed is increase so, we use 128 bit (in the probability of 1− 3 times / total cases) to give more speed to the system .

c. Efficiency to use and in exchange (switching) between the cases; This is due to use a center of processing (SDA).

d. High security, because the encrypt text cannot be dealt with anyone just who has the two keys for the algorithm under this system .

e. K_s generated randomly ,so impossible know anyone the sequence of the cases.

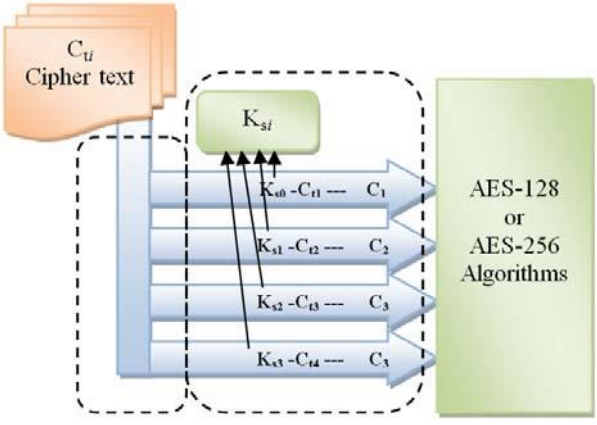

Fig. 5 : SDA system operations with C_{ti}

6. Conclusion

In this paper, a quick overview how we can use Encryption to send messages securely, using two AES methods. More importantly, we must use the new technology if we need upgrade the method that we used it .In essence, we use here three keys, first one to specify the sequence and two keys –different keys one 128 bit and another 256. Complexity at least one for three to increase the speed, and speed will increase by at least one for three of the complexity.. In finally, this system is efficiency and high security.

References

[1] "Advanced Encryption Standard (AES)" Federal Information Processing Standards Publication 197, Nov. 26, 2001.

[2] A. Lee, NIST Special Publication 800−21, Guideline for Implementing Cryptography in the Federal Government, National Institute of Standards and Technology, November 1999 http://csrc.nist.gov/publications/fips/fips197/fips_197.pdf .

[3] AES page available via http://www.nist.gov/CryptoToolkit .

[4] Chih−Chung Lu and Shau−Yin Tseng, "Integrated Design of AES (Advanced Encryption Standard) Encrypter and Decrypter", Internet Platform Application Department, Computer & Communications Research Laboratories, Industrial Technology Research Institute, Hsinchu, Taiwan, ROC.

[5] David Hook, "Beginning Cryptography with Java," in Wrox, 2005.

[6] E. Barker, L. Bassham, W. burr, M. Dworkin, J. Foti, J. Nechvatal, and E. Roback, "Report on the Development of the Advanced Encryption Standard (AES)." Available at http://home.ecn.ab.ca/~jsavard/crypto/co040801.htm .

[7] J. Daemen and V. Rijmen, AES Proposal: Rijndael, AES Algorithm Submission, September 3, 1999.

[8] J. Daemen, V. RijnMen, "The Rijndael Block Cipher," AES proposal, First AES Candidate Conference (AES1), Aug. 20−22, 1998.

[9] Jason Weiss, "Java Cryptography Extensions: Practical Guide for Programmers (The Practical Guides)," Morgan Kaufmann,2004.

[10] Jonathan B. Knudsen, "Java Cryptography," in Oreilly ,First Edition May 1998.

[11] Luis Miguel Cortés_Peña, "SeChat: An AES Encrypted Chat", http://users.ece.gatech.edu/~cortes/SeChat/SeChat.pdf .

[12] Oracle Sun Developer Network, Lesson 3: Cryptography, http://java.sun.com/developer/onlineTraining/Programming/BasicJava2/crypto.html.

[13] T. Ichikawa, T. Kasuya, M. Matsui, "Hardware Evaluation of the AES finalists," The Third Advanced EncryptionStandard (AES3) Candidata Conference, April, 13−14, 2000.

[14] Firas Layth Khaleel Alameen "Data Encryption Using Multi Codes For One Character (MCFOC)" IJCSNS International Journal of Computer Science and Network Security, VOL.10 No.9, September 2010, 1-4.

Moceheb Lazam Shuwandy received the B.S. degrees in Computer and Software Engineering from Al–Mustansiria University Baghdad – College of Engineering – Department of Computer Engineering and Software 1998–2003 and now study M.S. degrees in Information Technology of Utara University of Malaysia – College of Arts and Sciences in 2009 and he still. During 2005–2008, he worked as supervisor of the Internet networks in the University of Tikrit, Iraq (UTI). In 2005 he served as director of the website of the University of Tikrit, Iraq. In 2006–2009 he worked as a lecturer in the Faculty of Education / Department of Mathematics – University of Tikrit, Iraq in article Visual Studio® J #. He is a member of the Iraqi Engineers Syndicate /Baghdad 2004. He is now completing the study of a masters degree at the University of Utara.

Adib M.Monzer Habbal received his degree in computer engineering and post graduate Diploma in Informatics engineering from Aleppo University, Syria, in 2003 and 2005 respectively. In March 2007, he received the Master's degree in Information Technology from University Utara Malaysia, Malaysia. Currently, He is a lecturer at UUM, Malaysia. His main research interests include Internet performance engineering, network security, network application, TCP and congestion control in j Mobile Ad–hoc Networks. He serves on the Technical program committee of many international conferences, such as NETAPPS 2010, ICCAIE 2010, and ISCI 2011. Also he is a technical reviewer for IEEE TENCON 2009 and IEEE WiMob 2010.

Firas Layth Khaleel I'm graduated from AL–RAFIDAIN University College in Baghdad , Department of Software Engineering (2003–2004) and I have B.S.c In Software Engineering. I have been appointed at Tikrit University after having a B.S.c degree , as a programmer in internet and computer unit . After six month of my appointment date I have been appointed as a manager of internet and computer unit .Now I'm student M.S.c (IT) in University of Utara in Malaysia(UUM) This is 1st semester of Information Technology (2009–2010).

Ali Khalil Salih received the B.S. degrees in Computer of sciences from Tikrit University – College of sciences – Department of Computer sciences 2001–2005 and now study M.S. degrees in Information Technology of Utara University of Malaysia – College of Arts and Sciences in 2009 and I am still. During 2006–2009, I am worked as employed in the University of Tikrit, Iraq (UTI). . I am now completing a masters degree at the University of Utara.