International Journal of Computational Intelligence Research
ISSN 0973-1873 Volume 13, Number 7 (2017), pp. 1679-1698

http://www.ripublication.com

Expansion of Round Key Generations in Advanced Encryption Standard for Secure Communication

Awadhesh Kumar[1] and R.R. Tewari[2]

[1] & [2]: Department of Electronics & Communication University of Allahabad, Allahabad, Uttar Pradesh, India.

Abstract

21st century is the digital era of information technology and nowadays, cryptography is widely used to solve real-world information security problems that arise in many digital applications such as ATM cards, computer passwords and online shopping. Since all the transaction and communication are paper less and totally depends on the communication technology hence security is one of the challenging issues in making a secure communications. There are several cryptography algorithms such as DES, Double DES, 3DES, IDEA, and AES are used for securing the messages in the form of encryption and decryption. In this paper we implement a round sub keys generation algorithms based on the pre-computation key schedule and on the fly key generation schedule should be tested on differed operational modes of AES. The AES algorithm is faster as compared to other algorithms such as DES, 2DES and 3DES because encryption and decryption are done in nonlinear manner and less number of round. We propose a byte oriented 512 bit AES key generation algorithms which provides better security and faster implementation since less iteration are required for generating all the round sub keys.

Keywords: Cryptology, AES, Encryption, Decryption, Key schedule, Block cipher, Round Transformation.

1. INTRODUCTION:

Cryptography is an art and science of designing security algorithm for providing security services whereas cryptanalysis is the science of breaking the security algorithm and obtained the original message [1]. Cryptography is the process of maintaining the

secure transmission of information by encrypting the information when it is sent or shared by an entity i.e. sender and decrypts the shared information/ data when received by the entity that is receiver. This means cryptography algorithms are used for protecting the data in the form of encryption and decryption. Any cryptographic algorithms is a computational procedure consisting of three element input, process and output. Input of cryptographic algorithm is a combination of plaintext i.e. original message and cryptographic key, process them according to the algorithm and produces a result as a output in the form of encrypted text also known as cipher text. Based on the nature of key, cryptographic algorithm are categories in three broad classes such as symmetric key, asymmetric key and hybrid key cryptography. In a symmetric key cryptography same key are used for encryption and decryption while Asymmetric key also known as public key cryptography uses pair of different but mathematically related keys for encryption and decryption process. Hybrid key cryptography is a combination of Symmetric and Asymmetric key Cryptography. Cryptographic key is a parameter used in conjunction with a cryptographic algorithm that determines its operation in such a way that an entity with knowledge of the key can reproduce or reverse the operation, while an entity without knowledge of the key cannot [2]. Cryptographic operations requiring the use of cryptographic keys include:

- The transformation of plaintext data into cipher text data and vice versa.
- The computation of a digital signature from data and verification of digital signature
- The computation of an authentication code from data and verification of authentic code.
- The computation of a shared secret that is used to derive keying material

There are two types of cipher use in symmetric algorithms block cipher and stream cipher. In block cipher a block of plaintext transformed into cipher text while in stream cipher each bit of plaintext is transformed into cipher text. Data Encryption standard (DES) and Advanced Encryption standard (AES) are two most popular block cipher symmetric key algorithms used for secret communication. DES operates on 64 bit block with a 56 bit key using a Feistel structure with a total of 16 iteration or round from computing block of cipher text from a 64 bit block of plaintext. Hence in an each separate round separate keys are required. Here 16 sub keys are computed from the original key or 56 bit one for each round. In 1999 the US National Institute of Standards and Technology (NIST) indicated that DES should only be used for legacy systems and even though 3DES resists brute-force attacks with today's technology, there are several problems with it [3] and AES is used instead of DES and 3DES because of:

- DES was not very efficient with regard to software implementations and 3DES is three times slower than DES.
- DES having relatively short block size of 64 bits as AES uses 128 bit block, hence AES is faster as DES.

- DES uses Feistel structure and Feistel structure do not encrypt the entire block in one round, it takes two round for encrypting the entire block. This is the reason AES uses fewer number of round as compared to DES.
- AES is based on S-P network in which the entire 128 bits input block is organized as 4x4 bytes array called State and is processed in several rounds.
- DES uses a 56 bit key instead AES uses 128,192,256 bit key sizes which improves the security of message.
- AES is a Byte oriented cipher while DES uses bit oriented cipher
- The number of internal rounds of the cipher in AES is a function of the key length according to the table given below:

Key Length	**Round Required**
128	10
192	12
256	14

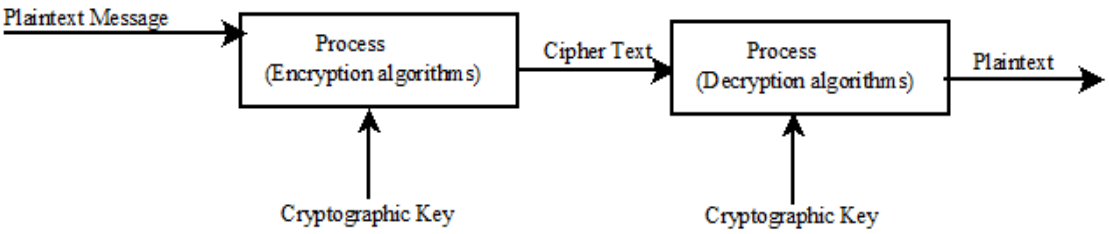


Figure 1: Cryptography component

2. METHODOLOGY

***2.1 Generation of keys for Symmetric key Algorithms*:** Symmetric key algorithm uses same key for protecting the shared information, removal or verify the protection [5]. Only authorized users knows the keys and apply it for verify or remove the protection. Shared secret keys are known by multiple entities that are said to share the confirmation and it is not owned and used by single entity. The shared secret key can be generated by [5]:

- One or more of the entities that shared a secret key
- A Trusted Party that provides the key to the intended sharing entities in a secure manner. The Trusted Party must be trusted by all entities that will share the key not to disclose the key to unauthorized parties or otherwise misuse the key.

There are different ways of key generation in symmetric key cryptography are as follows [5]:

***Direct Generation of Keys*:** Random bit generator are used for direct creation of symmetric keys. We adopt any of the random bit generator algorithms for generating the key and choice of secret key generation depends of the type of applications. The length of generated key is determined by algorithm and for which desired security strength to be provided.

Key derivation from pre-Shared Key: Sometimes cryptographic keys can be derived from other cryptographic keys. For generating a derived keys, key distribution function and previously derived keys are used for generating a new keys. In a DES and AES the process of Transforming plaintext into cipher text is completed in a number of rounds and each round needs a separate keys that are derived from its previous round. The security strength of generated key depends on the security strength of derived key and security strength of key distribution function.

***Key Derivation from Password*:** Now a days there are number of popular application in which keys are derived from passwords. Since password usually contained little number of bits and little entropy therefore it can be easily guessed. Hence these application users are strongly advised to select a password with large amount of entropy.

***Component Based Key Generation*:** The direct and indirect both process of key generation is performed if one entity can be trusted to have a full control of particular key generation process. In some situation for secret key generation, it is not desirable to trust one entity with key generation. In such situation distribute the process of key generation among a group of entities in such a way that no member of the group individually have control over the process. So we generate a key in component form as follows:

Assume we generate a key of 64 bit. There are three entity named as A, B, C generate a randomly key component of 64 bit and denote it as K_A, K_B, K_C respectively. The entities A, B and C securely transfer their component to secure combiner. The secure combiner derive the key K from the separate component $K = K_A \oplus K_B \oplus K_C$. The key K is only reconstructed within the secure combiner and not output to the entities involved in the key derivation process. Suppose $R = K_A \oplus K_B$ and hence $K = R \oplus K_C$ and it is different from R.

***Replacement of Symmetric Key*:** Some situation occurs such as end of key crypto period then replacement may be accompanied by a rekeying process. Rekeying is the replacement of old key with new key and is generated in a manner that is entirely independent of the value of old key.

2.2 Mode of operation in Block cipher Cryptography: The objective of encrypting the data by using different mode of operation to provide confidentiality and authentication of message send from sender to receiver [6]. There are several types of encryption modes for block cipher but here we discuss four mode of operation named as ECB, CBC, OFB and CFB. In case of ECB and CFB mode the length of plain text message be an exact multiple of the size of the cipher block. If the plaintext does not have multiple of size of block, it must be padded by using some padding technique [7]. The following are the modes of operation in [8].

***Electronic code book mode*:** In ECB mode block cipher operates in parallelized manner. This allowed high speed of data implementation. Each block is separately encrypted/decrypted by using

$$Encryption: Y_i = E_k(X_i)$$

$$Decryption: X_i = E_k^{-1}(Y_i) = E_k^{-1}(E_k(X_i)) = X_i$$

Encryption in ECB mode is highly deterministic i.e. identical plain text block results identical cipher text block as long as the key does not change. Hence attacker recognize if the same message has been sent twice by looking at cipher text.

Cipher Block Chaining Mode: In this mode of encryption consists of two steps: (i)all block are enhanced together such that the cipher text Y_i depends not only on block X_i but on all previous plaintext block as well. (ii) The encryption is randomized by using Initialization Vector (IV) and if we choose new initialization vector every time the encryption by using CBC mode become a probabilistic encryption scheme. The encryption and decryption of plain text are as follows:

$$Encryption\ First\ Block: Y_1 = E_k(X_1 \oplus \text{IV})$$

$$Encryption\ Genrel\ Block: Y_i = E_k(X_i \oplus Y_{i-1}) \quad \forall i \geq 2$$

$$Decryption\ First\ Block: X_1 = E_k^{-1}(Y_1) \oplus \text{IV}$$

$$Decryption\ Genral\ Block = E_k^{-1}(Y_i) \oplus Y_{i-1} \quad \forall i \geq 2$$

Output Feed Back Mode: In OFB mode a block cipher is used to build a stream cipher encryption scheme. In this scheme key is not generated bit wise but instead in a block wise fashion. The OFB Encryption is non deterministic i.e. encrypting the same plaintext twice resulting different cipher text. Let $E()$ be a block cipher of block size b and $X_i, Y_i\ and\ S_i$ be a bit string of length b and Initialization vector IV be a nonce of length b then The encryption and decryption are perform as follows:

$$Encrytion\ first\ block: S_1 = E_k(\text{IV}), \qquad Y_1 = S_1 \oplus X_1$$

$$Encryption\ for\ Genral\ Block: S_i = E_k(S_{i-1}),\ Y_i = S_{i-1} \oplus X_i \quad \forall i \geq 2$$

$$Decryption\ for\ first\ Block: S_1 = E_k^{-1}(IV), X_1 = S_1 \oplus Y_1$$

$$Decryption\ for\ Genral\ Block: S_i = E_k^{-1}(S_{i-1}), X_i = S_{i-1} \oplus Y_i \quad \forall i \geq 2$$

Cipher Feed Back Mode*:* this mode is used as a building block of stream cipher. This mode of operation generally similar to OFB mode but instead of feeding back the output of block cipher, the cipher text is fed back. CFB mode uses Initial Vector and is nondeterministic and also forms asynchronous stream cipher, since stream cipher output is also a function of the cipher text. The idea behind this mode is as follows:

$$Encryption\ for\ first\ Block: Y_1 = E_k(\text{IV}) \oplus X_1$$

$$Encryption\ for\ Genral\ Block: Y_i = E_k(Y_{i-1}) \oplus X_i \quad \forall i \geq 2$$

$$Decryption\ for\ first\ Block: X_1 = E_k^{-1}(IV) \oplus X_1$$

$$Decryption\ for\ Genral\ Block: X_i = E_k^{-1}(Y_{i-1}) \oplus Y_i \quad \forall i \geq 2$$

2.3 Internal Structure of Round in AES: In the encryption and decryption process of AES, the State array is modified at each round by a round function that defines four different byte-oriented transformations [8]. The Cipher Key is similarly pictured as a rectangular array with four rows and number of columns of the Cipher Key is denoted by Nk and is equal to the key length divided by 32. 4-byte vectors will sometimes be referred to as words. The structure of round transformation are as follows:

1. All the 16 byte input message are arranged in a four-of-four byte matrix called state matrix are as follows:

$$[A]_{4X4} = \begin{bmatrix} A_0 & A_4 & A_8 & A_{12} \\ A_1 & A_5 & A_9 & A_{13} \\ A_2 & A_6 & A_{10} & A_{14} \\ A_3 & A_7 & A_{11} & A_{15} \end{bmatrix}$$

2. Key bytes are arranged in to a matrix with four rows and 4, 6 or 8 columns as the length of key of 128,192 or 256 bit respectively. The key array of 256 bit keys are as follows:

$$[K]_{4X8} \quad \begin{bmatrix} K_0 & K_4 & K_8 & K_{12} & K_{16} & K_{20} & K_{24} & K_{28} \\ K_1 & K_5 & K_9 & K_{13} & K_{17} & K_{21} & K_{25} & K_{29} \\ K_2 & K_6 & K_{10} & K_{14} & K_{18} & K_{22} & K_{26} & K_{30} \\ K_3 & K_7 & K_{11} & K_{15} & K_{19} & K_{23} & K_{27} & K_{31} \end{bmatrix}$$

3. Byte Substitution Layer: this is the first layer of each round for encryption of text. In this layer 16 S-Boxes are used and each byte of state matrix A_i is replace by another byte of state matrix $B_i$by using the following equation:

$$S(A_i) = B_i$$

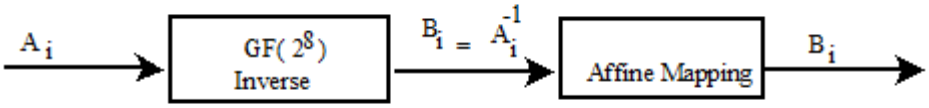


Figure 2: Process of Sunsitution layer

The substitution of S-Box is a bijective mapping. This property allow us to uniquely reverse the S-box which is needed for decryption. The AES S-box can be viewed as two step mathematical transformation as given in the figure2.

4. Shift Row Transformation Layer: In this layer cyclically shift the second row by one position left, third row by two position left and forth row by three position left and first row has no change.

$$\begin{bmatrix} B_0 & B_4 & B_8 & B_{12} \\ B_1 & B_5 & B_9 & B_{13} \\ B_2 & B_6 & B_{10} & B_{14} \\ B_3 & B_7 & B_{11} & B_{15} \end{bmatrix} \xrightarrow{After\ Shift\ Row\ Transformation} \begin{bmatrix} B_0 & B_4 & B_8 & B_{12} \\ B_5 & B_9 & B_{13} & B_1 \\ B_{10} & B_{14} & B_2 & B_6 \\ B_{15} & B_3 & B_7 & B_{11} \end{bmatrix}$$

5. Mix Column Transformation: This is a linear transformation which mix each column of the state matrix obtained after shift row transformation and all the arithmetic involving the coefficients is done in Galois Field (GF(2^8)).

$$MixColum(B) = C$$

$$\begin{bmatrix} C_0 \\ C_1 \\ C_2 \\ C_3 \end{bmatrix} = \begin{bmatrix} 02 & 03 & 01 & 01 \\ 01 & 02 & 03 & 01 \\ 01 & 01 & 02 & 03 \\ 03 & 01 & 01 & 02 \end{bmatrix} \times \begin{bmatrix} B_0 \\ B_5 \\ B_{10} \\ B_{15} \end{bmatrix}$$

The combination of shift row and mix column transformation is also known as Diffusion layer and it perform linear operation on state matrix i.e.

$$DIFF(A) + DIFF(B) = DIFF(A + B)$$

6. Key Addition Layer*:* In this layer the state byte matrix obtained after mix column transformation layer are XORed with the sub keys of previous round. Which also consist of 16 bytes.

2.3.1. Enhanced Key AES Algorithms:

Input: 128 bit block of plaintext message and 512 bit cipher key

Output: 128 bit block of Encrypted message of cipher text.

Procedure:

1. 128 bit input block of plaintext is divided in to sixteen byte input $A0, A1, \ldots\ldots A15$ and these bytes are XORed with round zero sub key $K0$ that is first four word of 32 bit each and this key $K0$ is just like as original cipher key.
2. All Sixteen byte obtained after step first i.e. $A'0, A'1, \ldots\ldots, A'15$ that are fed byte wise into the S-box and Generate Sixteen byte output $B0, B1, \ldots\ldots, B15$
3. The output of S-box byte wise are passed through Diffusion layer in which shift by shift Sub layer and Mix Column Transformation C(x) are performed.
4. Finally the 128 bit sub keys Ki are XORed with the intermediate results obtained in step three.
5. The above three steps namely 2,3,4 are repeated in $N_r - 1$ rounds where N_r is sthe number of round required depend on the size of key(16 in case of 512 bit key) and after completion of 9 rounds final round are computed as follows.
6. In the last round input are taken as an output after round $N_r - 1$ and apply Byte substitution layer, shift row sub layer and Add key layer and take an output in the form of cipher text.

***2.3.2 AES Round Key Generations Algorithm*:** there are two different approaches exist for implementing any of the key schedules.

Pre-Computation key schedule: In this approach of key schedule firstly all sub keys are expanded in to an array of word[W] and then encryption and decryption are carried out. This approach takes large amount of memory for computation hence not suitable for limited memory devices such as smart card , PDA, etc[8]. The required memory in the computation of key depends on the length of key and the number of round as follows:

$$\begin{aligned} Required\ Memory\ &= (nr+1).Size\ of\ key\ in\ \ bytes \\ &= 11 * 16 = \ 176\ byte\ in\ case\ Length\ of\ key\ is\ 128\ bit \\ &= 13 * 24 = \ 312\ bytes\ in\ case\ of\ key\ length\ is\ 192\ bit \\ &= 15 * 32 \\ &= \ 480\ bytes\ in\ case\ length\ of\ key\ is\ 256\ bit. \\ &= 17 * 64 \\ &= \ 1088\ bytes\ in\ case\ of\ length\ of\ key\ is\ 512\ bit \end{aligned}$$

***On the Fly Key Schedule*:** In this approach of key generation a new sub keys are derived for every new round during encryption/decryption of plaintext/ cipher text [8]. When decrypting cipher text it is required to recursively derive all sub keys and then start the decryption of cipher text on the fly generation of sub keys.

2.3.3. AES Different Length Key Schedule Algorithms: In Advanced Encryption

Standard (AES) algorithm key schedule algorithms are used for deriving the round key Ki from the original key K=128, 192,256,512 bit in the form of word where size of each word is 4 byte i.e. 32 bit and words are stored in the word Matrix W and rows and columns in the word matrix W depends on the size of key. The XOR addition of sub keys are used both at input and output of AES and sometimes this process is referred to as Key Whitening [9]. The number of sub keys needed for computing the cipher text from plaintext is equal to the number of round in algorithm plus one. If r is the number of round used, then the number of sub keys is r+1. One extra key needed for key whitening in the first key addition layer. The sub keys in AES are computed recursively such as for computing the key K_i , K_{i-1} must be known and The key schedule of AES is word oriented where the size of each word is 32 bit and sub keys are stored in key Expansion Array W that consist of words. The number of iteration required for generation of all the round keys depends on the size of key and size of fixed block size. When the length of key increases the required number of iteration for generating a round sub key reduces according to the given table:

Key Length	**Round Required**	**Number of Sub keys**	**Required Iteration**
128	10	11	10
192	12	13	8
256	14	15	7
512	16	17	4

Following are the algorithm used for maintaining round sub keys from given original key of Length 128,192,256 and 512 bits. here we give 256 and 512 bit key generation algorithm modal.

2.3.3.1. AES Round Key Generations Schedule for key size 256 bit: The process of computing 256 bit round keys are depicted in the figure 3 and 4 and algorithms are as follows:

$Input$: $the\ original\ key\ of\ length\ 256\ bit$

$Output$: $15\ round\ subkeys\ named\ as\ K0, K1, \ldots\ldots, K15.$

Procedure:

1. The element of original input key arranged in byte wise such as $K0\ K1\ \ldots\ldots\ldots K15\ K16, \ldots\ldots\ldots\ldots K30, K31$ and starting from most

significant byte to least significant byte key length is divided in four words each are equal size of 32 bit hence form 8 word in each row.

2. All the sub keys are stored in key expansion array with the element $W[0], W[1], \ldots . W[59]$ because there are 15 sub keys used for maintaining 14 round and 7iteration are required. First sub keys $K0$ is obtained from taking firsts four words from starting original key of AES and key is copied in to first four element of key array $[W0, W1, W2, W3]$ and second key is obtained from least significant 4 word of original key $[W4, W5, W6, W7]$ each of word size 32 bit and remaining sub keys are obtained by the step defined below.
3. The two function $g( )$ and $h( )$ is computed where $g( )$ is computed over least significant word of key length that rotates its 4 input bytes, then perform byte wise S-box substitution and adds Round Coefficients(RC) is an element of Galois Field(2^8) i.e. an 8 bit value. It is added only the left most byte in the function g( ) and round coefficient vary from iteration to iteration according to the following rule:

$$RC(1) = x0 = (00000001)_2$$

$$RC(2) = x1 = (00000010)_2$$

$$\ldots\ldots\ldots\ldots$$

$$\ldots\ldots\ldots\ldots\ldots$$

$$RC(7) = x6 = (01000000)_2$$

The function h is computed as follows: h ( ) takes 4 byte input and apply the substitution on each byte and produces 4 byte output

4. All other element of the array are computed as follows: the left most word of key of iteration 1 to 7 are $W[6 * i]$ where $i = 1\ to\ 7$
   a. $W[8 * i] = W[8 * i] + g(W[8 * i - 1])$
   b. $W[8 * i + j] = W[8 * i + j - 1] + W[8(i - 1) + j]\ where\ i = 1\ to\ 7\ and\ j = 1,2,3$
   c. $W[8 * i + j] = h(W[8 * i + j - 1]) + W[8 * (i - 1) + j], for\ i = 1to\ 7\ and\ j = 4$
   d. $W[8 * i + j] = W[8 * i + j - 1) + W[8 * (i - 1) + j], for\ i = 1to\ 7\ and\ j = 5,6$
5. After computing All the element of word Matrix [W0,W1,……..W59] , we compute 15 sub keys starting from K0 to K14 by taking first four word for making $K0 = [W0, W1, W2, W3], K1 = [W4, W5, W6, W7], \ldots \ldots K14 = [W56, W57, W58, W59]$.

***2.3.3.2. AES Round Key Generations Schedule for key size 256 bit*:** The process

of computing 17 round keys are depicted in figure 5 and describes as follows

$Input$: $the\ original\ key\ of\ length\ 512\ bit$

$Output$: $17\ round\ sub\ keys\ named\ as\ K0, K1, \ldots\ldots, K16.$

Procedure:

1. The element of original input key arranged in byte wise such as $K0, K1 \ldots\ K15, K16, \ldots\ldots\ldots\ldots K62, K63$ and starting from most significant byte to least significant byte key length is divided in four words each are equal size of 32 bit hence form 16 word in each row.
2. All the sub keys are stored in key expansion array with the element $W[0], W[1], \ldots\ldots W[67]$ because there are 17 sub keys used for maintaining 16 round and four iterations are required. First four sub keys $K0$ to $K3$ is obtained from taking firsts four words of matrix from original key of AES and key is copied in to first four element of key array $[W0, W1, W2, W3]$ and second key is obtained from next four word of original AES key $[W4, W5, W6, W7]$ and so on, each of word size 32 bit and remaining sub keys are obtained by the step defined below.
3. The four function $g()$ and $h()$ $t()$ and $y()$ is computed where $g()$ is computed over least significant word of key length that rotates its 4 input bytes, then perform byte wise S-box substitution and adds Round Coefficients(RC) in an element of Galois Field(2^8) i.e. an 8 bit value. It is added only the left most byte in the function $g()$ and round coefficient vary from round to round according to the following rule:

$$RC(1) = x0 = (00000001)2$$

$$RC(2) = x1 = (00000010)2$$

…… ……..

…… …… …… ……

$$RC(4) = x3 = (00001000)2$$

   a. The function $h()$ is computed as follows: $h(), t()\ and\ ()$ takes 4 byte input and apply the substitution on each byte and produces 4 byte output.
4. All other element of the array are computed as Follows: the left most word of key of iteration 1 to 3 are $W[16 * i]$ where $i = 1\ to\ 3$
   a. $W[16 * i] = W[16 * i] + g(W[16 * i - 1])$
   b. $W[16 * i + j] = W[16 * i + j - 1] + W[16(i - 1) + j]\ where\ i = 1\ to\ 4 and\ j = 1,2,3$
   c. $W[16 * i + j] = h(W[16 * i + j - 1]) + W[16(i - 1) + j]\ for\ i = 1 to\ 4\ and\ j = 4$
   d. $W[16 * i + j] = h(W[16 * i + j - 1]) + W[16(i - 1) + j]\ for\ i =$

$1 to\ 4\ and\ j = 5,6,7$

e. $W[16 * i + j] = t(W[16 * i + j - 1]) + W[16(i-1) + j]\ \ for\ i = 1 to\ 4\ and\ j = 8$

f. $W[16 * i + j] = W[16 * i + j - 1] + W[16(i-1) + j]\ \ for\ i = 1 to\ 4\ and\ j = 9,10,11$

g. $W[16 * i + j] = y(W[16 * i + j - 1]) + W[16(i-1) + j]\ \ for\ i = 1 to\ 4\ and\ j = 12)$

h. $W[16 * i + j] = W[16 * i + j - 1]) + W[16(i-1) + j]\ \ for\ i = 1 to\ 4\ and\ j = 13,14,15)$

5. After computing All the element of word Matrix $[W0, W1, \ldots\ldots\ldots..W67]$, we compute 15 sub keys starting from K0 to K14 by taking first four word for making $K0 = [W0, W1, W2, W3], K1 = [W4, W5, W6, W7], \ldots\ldots K15 = [W60, W61, W62, W63]$.

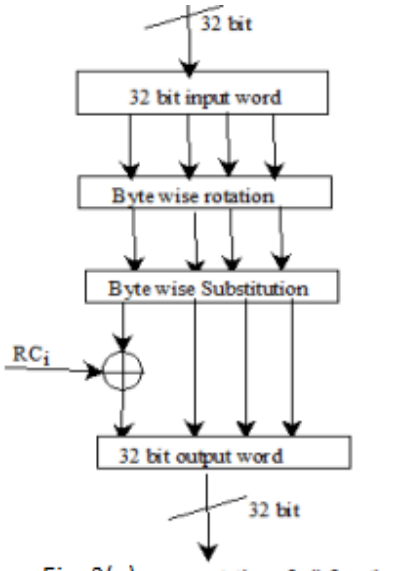


Fig. 3(a): computation of g() function

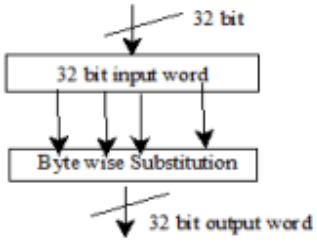


Fig. 3(b): h() function computation

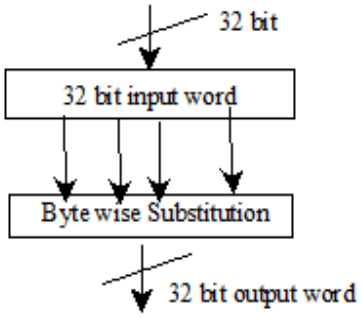


Fig. 3(c): t () function computation

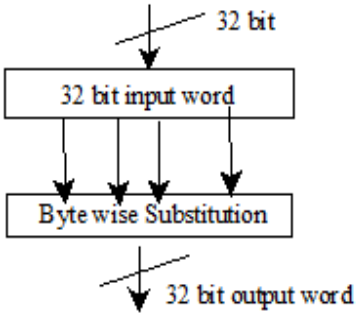


Fig.3(d): y () function computation

Figure 3: Computation of g(), h(), t() and y()

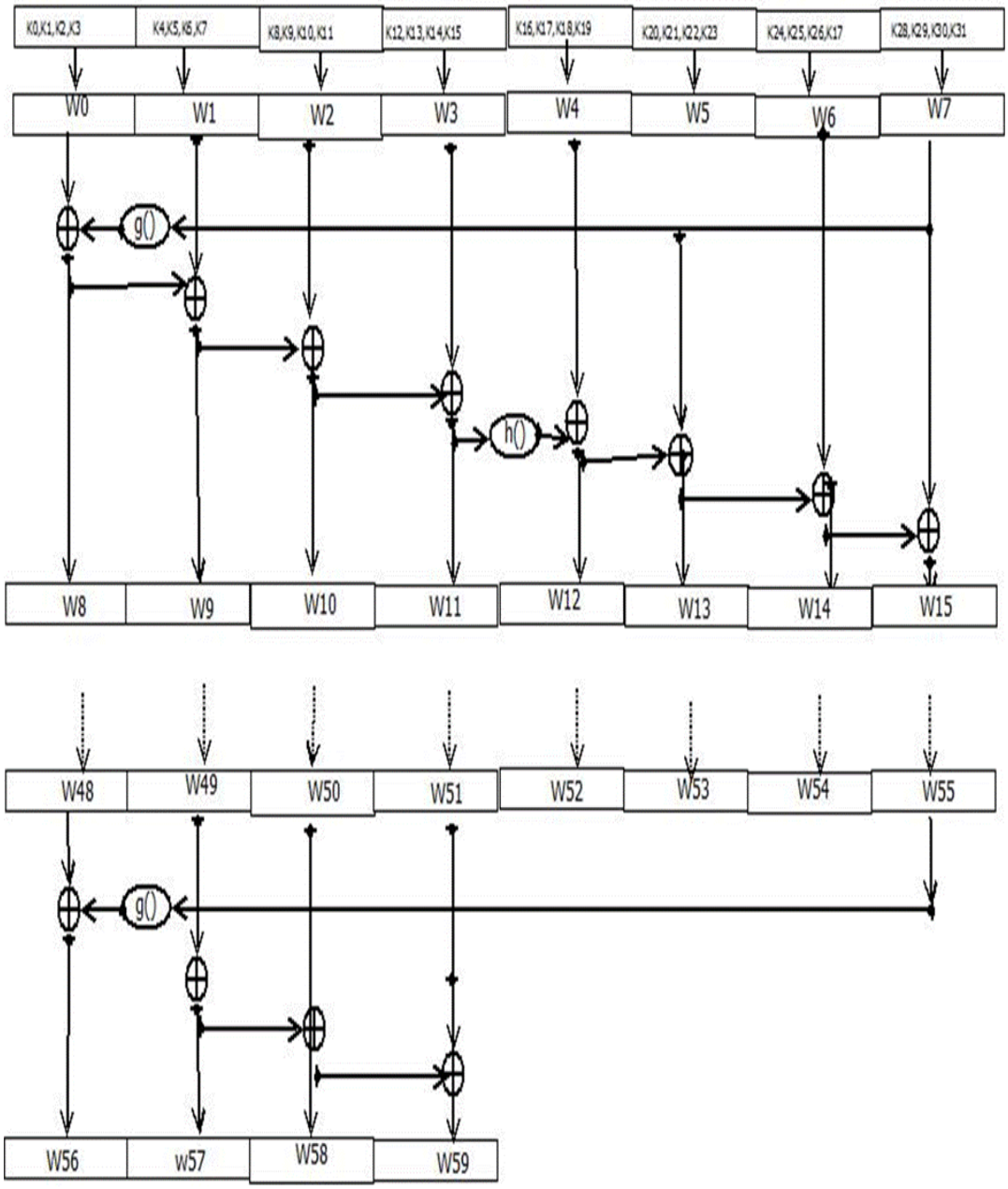


Figure 4: : 256 bit key Genration

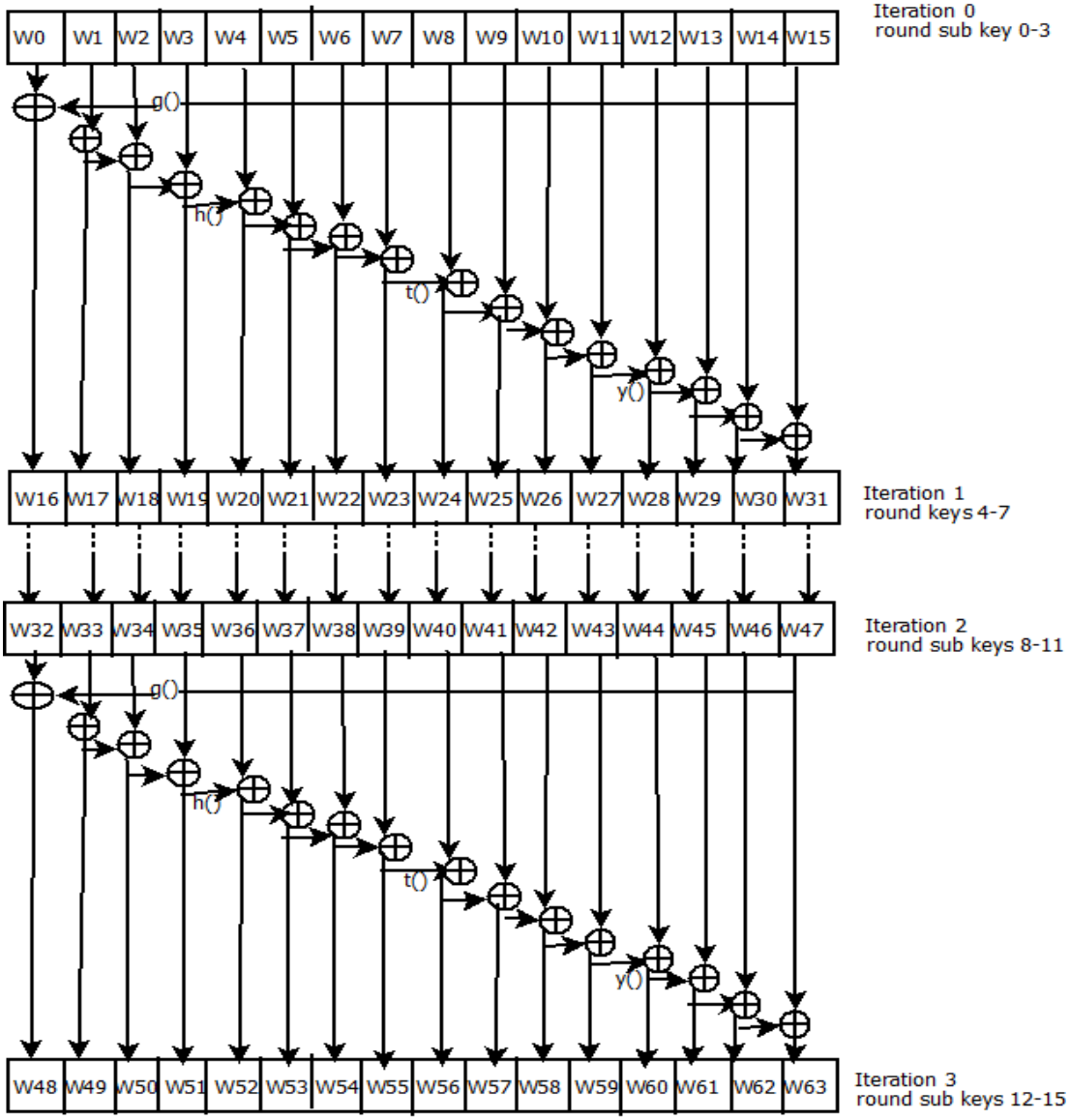


Figure 5 : 512 bit Round key genration

3. RESULT AND DISCUSSION:

The Advanced Data Standard (AES) algorithms is implemented in JAVA platform of jdk 1.8 and 64 bit windows operating system with hardware configuration of 4 GB RAM, 1TB hard disk. This algorithm is tested for both of Pre-schedule round key generation and on the fly key generation schedule and measure the performance of algorithms in terms of encryption, decryption time and total time required for both encryption decryption process with varies key size. There are different tables and figure shows the result and find the followings:

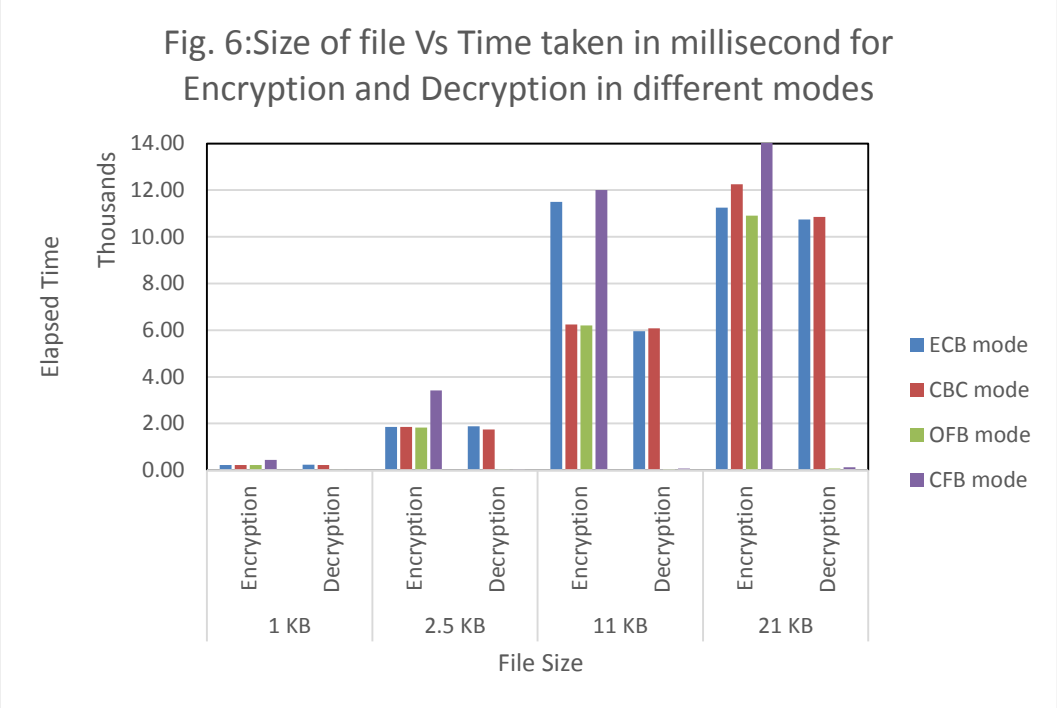


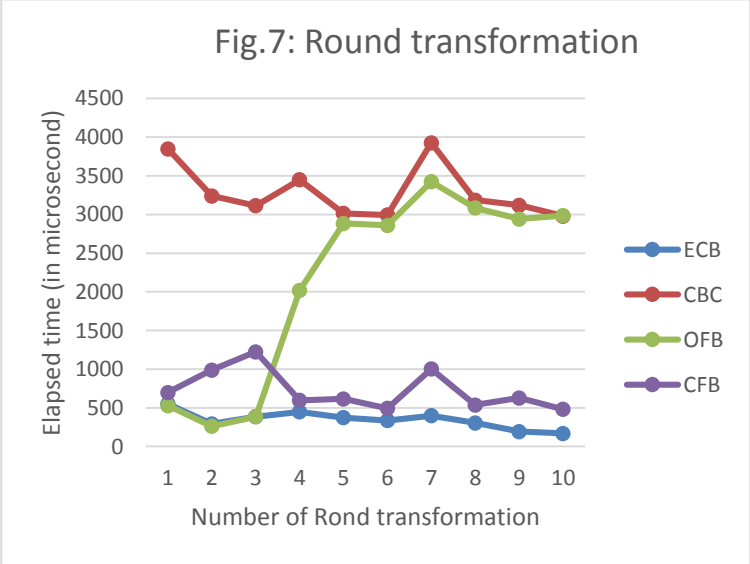
Table 1: Encryption-Decryption time (microsecond) of different size of text file with varies AES key sizes

File size	Key Size	Encryption time	Decryption time	Total Time
1 KB	128	14594	616	15210
	192	14432	612	15044
	256	16222	644	16866
2 KB	128	14873	735	15608
	192	13546	833	14379
	256	15267	756	16023
20 MB	128	17569	4156	21725
	192	18339	4783	23122
	256	16543	4479	21022
26 MB	128	17934	3713	21647
	192	18146	4647	22793
	256	18491	5262	23753

Table1 & figure 6 shows the time taken (millisecond) for Encryption and Decryption in various modes such as ECB, CBC , OFB and CFB with different file sizes in pre-computation sub key generation schedule of 128 bit length. From this figure and table it is clear that when the file size is less than encryption and decryption time are approximately same in ECB and CBC but encryption time by CFB is much larger than the other discussed operational modes but when file size increases the encryption time taken by different modes are in descending order as follows: CFB>CBC>OFB>ECB. The decryption time of different modes of operation when file size is smaller such as 1 KB and 2.5 KB are in descending order as: ECB>CBC>CFB>OFB but for larger file size such as 11 KB and 21 KB the decryption time in order of CBC>ECB>CFB>OFB and for decrypting the data OFB and CFB operational modes takes very little time as compared to ECB and CBC modes of operation.

Table 2: Round sub keys generation time (Microsecond) in different modes of AES with key size 128 bit

ROUND / AES MODE	Round 1	Round 2	Round 3	Round 4	Round 5	Round 6	Round 7	Round 8	Round 9	Round 10
ECB	552	293	385	448	374	337	398	306	195	168
CBC	3849	3239	3115	3448	3014	2993	3927	3187	3119	2979
OFB	529	261	385	2019	2883	2860	3425	3083	2940	2985
CFB	698	989	1224	597	616	993	1003	558	628	480



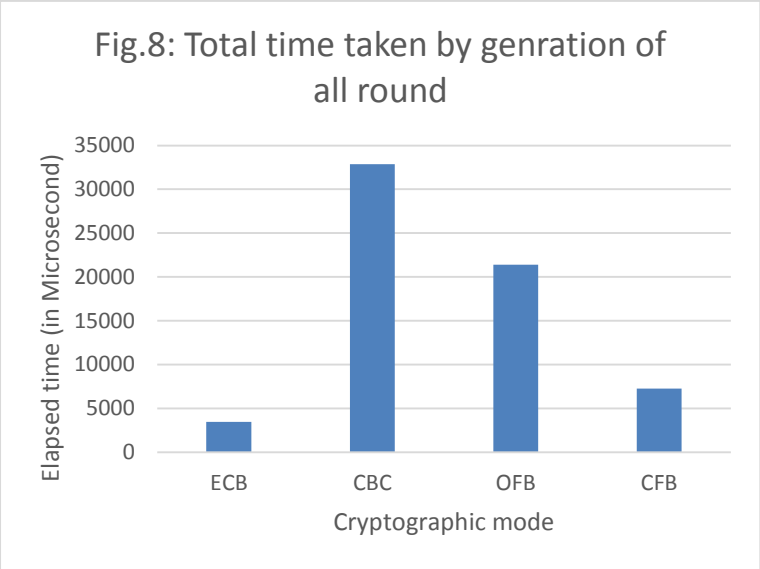


Table 2 and Figure 7 shows that the time taken by generation of round sub keys in different modes of operation. According to the table & figure when 10 round keys are generated in pre computation of key schedule CBC modes of operation takes much larger time among all and in ECB and CFB mode up to three round sub keys generation time are approximately same but after third round it is increases in the following order CBC> OFB >CFB >ECB. It shows that when apply ECB modes of operations in round sub keys generation then it will take very less time among remaining modes of operations.

Table 2 & Figure 8 shows that the total time taken by generation of all the sub keys by using pre computation key schedule in four modes of operation CBC mode takes much larger time for generation of all the sub keys and ECB takes much smaller time for generation of all round sub keys. The graph presented in fig 8 also shows that CBC, OFB and CFB modes takes approximately 6, 4, 2 multiples of time taken by ECB modes for generation of all the sub keys respectively. According to the time taken for generating all the round keys the modes in descending order as follows: CBC>OFB>CFB>ECB.

Table 3: Encryption-Decryption time (microsecond) of different size of text file with varies

File Size	Operations (Encryption/Decryption)	ECB mode	CBC mode	OFB mode	CFB mode
1 KB	Encryption	219.648	219.056	223.560	446.313
	Decryption	229.611	222.187	09.377	12.580
2.5 KB	Encryption	1848.040	1850.855	1826.728	3421.969
	Decryption	1877.184	1741.027	24.891	36.781
11 KB	Encryption	11497.18	6246.80	6208.342	12002.474
	Decryption	5951.17	6075.55	42.78	74.51
21 KB	Encryption	11254.230	12254.941	10909.23	21658.915
	Decryption	10746.423	10849.201	67.71	122.79

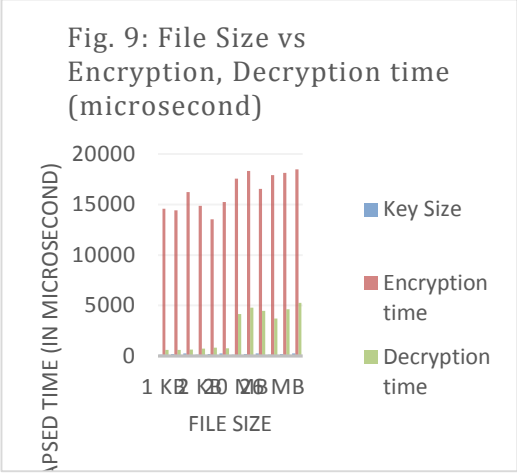


Fig. 9: File Size vs Encryption, Decryption time (microsecond)

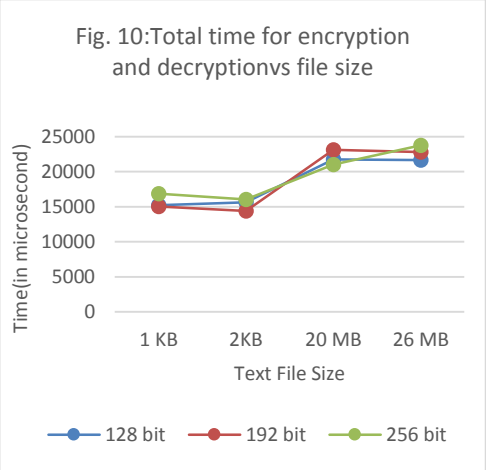


Fig. 10:Total time for encryption and decryptionvs file size

Table 3 and figure 9 shows that when size of key are 128 and 256 bit and file size are small the encryption time increases but when the key size 192 bit encryption time less as compared to 128,256 bit key. When the file size is larger the encryption time decreases with increasing the size of key. In case of decryption when file size and key size both increases the decryption time is also increases. The time required for encryption of the different size of data file is always larger than the time required for decrypting the same data file. this table and figure also shows that the total time for encryption and decryption both is less when the file size smaller such as 1 KB or 2 KB and for both encryption and decryption uses 192 bit key and is larger when 128 bit or 256 bit key are used. When the size of file increases and 256 bit key are used for encryption and decryption the time required for encryption/decryption is less in some cases but some cases 128 bit key is better for encryption and decryption and it will take less time but security is weaker as compared to other.

Table 4: Comparisons between pre-computation key schedule and on the fly key generation schedule

File size	Pre-Computation Key Generations		On the Fly Key Generation	
	Encryption time	Decryption Time	Encryption time	Decryption time
1 KB	1143	633	3.3	0.98
2 KB	1315	1290	4.11	1.5
1 MB	318583	196636	16.97	2.42
2 MB	1098386	382295	17.62	3.63

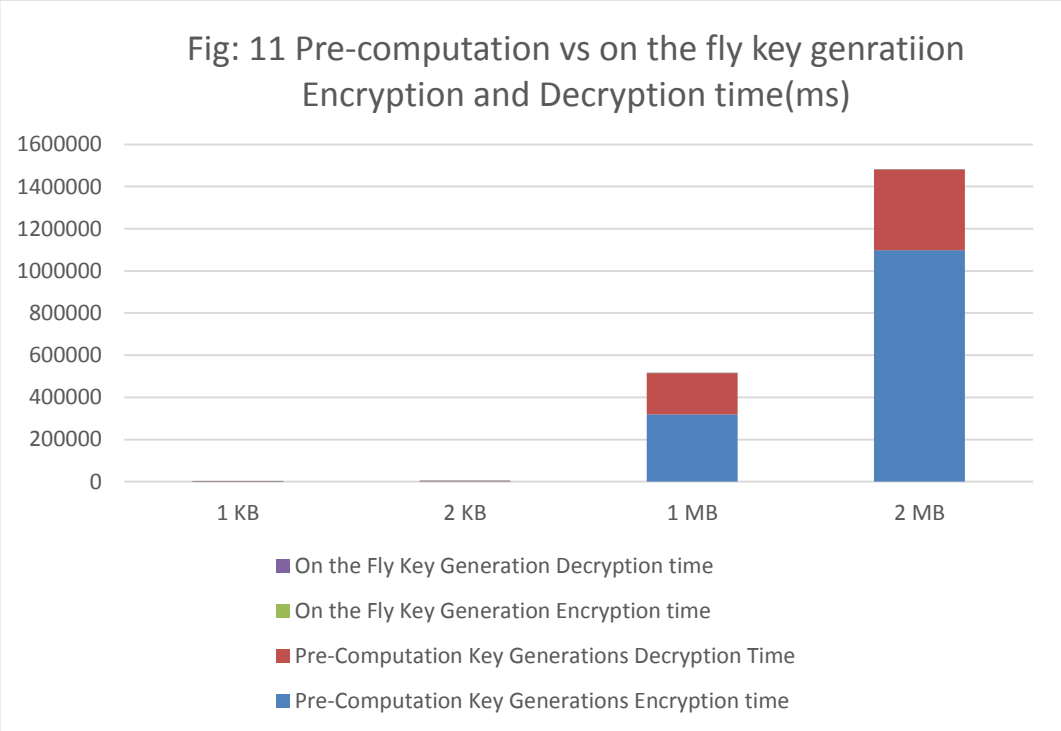


Table 4 and fig 11 it is clear that in pre-computation key schedule generation takes much larger time for encryption and decryption of data file as compared to on the fly key schedule generation. It is also clear that in pre-computation round sub key generation encryption will take larger time as compared to decryption the same data file. This result is also true in case of on the fly round key generation schedule.

4. CONCLUSIONS

Security is one of the big issue for making communication secure. AES algorithms uses the number of round depends on the length of key used for encrypting the plaintext in to cipher text. This paper analyse the time required for encryption and decryption during the pre-computation and on the fly key generation and find that on the fly round key generation takes very less time for encryption and decryption of data as compared to pre-computation key generation which takes large amount of time and memory. when data are encrypted /decrypted by using different operational modes CBC modes takes large time for performing both encryption and decryption process and OFB and CFB takes very little time for decrypting any text file. it is also find that when the key size is larger it will take a large time for encryption and decryption of data file of different size but provides better security and number of iterations for generation a round sub keys reduces when the size of key increases but computational task is complex. In the proposed model of 512 bit key generation there are 16 rounds required for transforming plaintext to cipher text but number of iteration reduces up to four and takes less time for generation of 17 sub key and provides better security then 256 bit keas. In future we implement this model and also makes it as word oriented rather than byte oriented.

REFERENCES

[1] Elbirt, Adam J. (2008). Understanding and Applying Cryptography and Data Security. London (New York): CRC Press, Taylor &Francis group

[2] Ferguson N. et al. (2001) Improved Cryptanalysis of Rijndael. In: Goos G., Hartmanis J., van Leeuwen J., Schneier B. (eds) Fast Software Encryption. FSE 2000. Lecture Notes in Computer Science, vol 1978. Springer, Berlin, Heidelberg.

[3] Barker, E. & Roginsky, A. (2012). Recommendation for Cryptographic Key Generation (p. 26).USA: NIST Special Publication 800-133,

[4] Dworkin, M. (2011). Recommendation for Block Cipher Modes of Operation: Methods for Key-Wrapping (p.29).USA: NIST Special Publication 800-38F.

[5] Dworkin, M. (2001). Recommendation for Block Cipher Modes of Operation Methods and Techniques, (p.59). USA: NIST Special Publication 800-38A.

[6] Paar, C., & Pelzl, J(2010). Understanding Cryptography: A Text book for Student and Practitioners, London: Springer-Verlag Berlin Heidelberg.

[7] Daemen,J., & Rijmen V. (1999).AES Proposal: Rijndael,Document (p.45) version 2, pp. 1-45, 1999

[8] Biryukov, A., & Khovratovich D. (2009). Related-Key Cryptanalysis of the Full AES-192 and AES-256(p.18).International Association of cryptographic research.