BULETINUL INSTITUTULUI POLITEHNIC DIN IAŞI
Publicat de
Universitatea Tehnică „Gheorghe Asachi” din Iaşi
Tomul LX (LXIV), Fasc. 1, 2014
Secția
ELECTROTEHNICĂ. ENERGETICĂ. ELECTRONICĂ

IMPROVING ADVANCED ENCRYPTION STANDARD USING ENLARGED DATA STRUCTURES

BY

LUMINIȚA SCRIPCARIU[*]

“Gheorghe Asachi” Technical University of Iași
Faculty of Electronics, Telecommunications and Information Technology

Received: April 19, 2013
Accepted for publication: May 28, 2013

Abstract. Information security is an important issue of any communication system. Different cryptographic algorithms are used to ensure the confidentiality of the transmitted information: public-key algorithms, such as RSA (Rivest, Shamir, Adleman) or El-Gamal, and secret-key algorithms, such as DES (Data Encryption Standard), TDES (Triple DES) and AES (Advanced Encryption Standard) (Schneier, 1996). The most powerful algorithm is considered to be AES, which uses 128, 192 and 256-bit encryption keys but technology evolves quickly so it is a matter of time to develop efficient attacks. Therefore it is necessary to make AES more robust and one way to use larger data structures, larger algebraic fields and longer encryption keys with length of 384, 512, 768 and 1,024 bits. The modified AES algorithm based on larger data structures is presented and its performances are analysed.

Key words: data security; data structures; Galois field; cryptography; AES.

1. Introduction

Confidentiality is an important security service in any communication system ensured using different cryptographic algorithms.

A robust cryptographic system should resist to all types of attacks namely:

[*] *e-mail*: lscripcariu@etti.tuiasi.ro

a) *Differential attack*, based on the correlation properties of the coded sequence (discovered in 1990).

b) *Linear attack*, searching for some correlation between the coded signal and the original.

c) *Brute-force attack*, looking for the encryption key from a large set of possible key-sequences.

Public-key techniques compete with the secret-key systems for the most robust algorithms, but it seems that the algorithms based on secret encryption keys are the most powerful.

The goal of a brute-force attack is to find the encryption key analysing a set of possible values. Its success depends on the key space dimension so it is recommended to use longer key sequences. Nowadays, 256-bit keys are long enough to avoid this type of attack.

The number of iterations made by an encryption algorithm is also important to ensure its robustness.

DES (Data Encryption Standard) is a binary symmetric block-code, with a 64-bit secret encryption key and it has 16 identical rounds (ANSI X3.92). It is vulnerable to brute-force attack. TDES (Triple Data Encryption Standard) was developed to avoid this weakness. It applies three times the DES encoding and decoding algorithms, with a longer key, of 128 bits.

AES (Advanced Encryption Standard) operates on octets, on the Galois Field (256), with secret encryption keys of 128, 192 up to 256 bits, and it makes 10, 12 or 14 iterations, depending on the length of the encryption keys. But successful attacks were developed only for a reduced number of iterations. Therefore, nowadays, it is widely used on computer networks, to ensure the privacy of stored information and transmitted data. It is adopted for different governmental and military applications but it is successfully used in other domains.

RSA (Rivest, Shamir, Adleman) operates on the decimal value of the message and its robustness is based on the impossibility to factorize very large numbers (of at least 300 digits). It can use public encryption keys up to 2,048 bits to prevent the brute force attack.

2. About Galois Fields

Many encryption algorithms can be defined on Galois Fields (GF). The binary alphabet is, in fact, a GF with only two elements.

A GF is an algebraic finite field with a number of elements equal to a power of 2. It is denoted by GF (2^m) (Proakis & Manolakis, 1988).

The GF is very convenient for encoding algorithms, because computation is simple and there are dedicated circuits and fast algorithms for GF algebra. There is no difference in complexity comparing to binary algebra.

An element, a, from the GF (2^m), can be expressed:

a) in the decimal system;

b) as a binary sequence of m bits

$$a = [a_0\ a_1\ a_2\ \ ....\ a_{m-1}]; \tag{1}$$

c) as the polynomial associated to the binary sequence

$$a(x) = \sum_{i=0}^{m-1} a_i x^i\ ; \tag{2}$$

d) exponentially, using a primitive element, *b*, of the field (if the element itself is not null)

$$a = b^e,\ a \neq 0. \tag{3}$$

For example, GF (256) works with bytes, having decimal values from 0 to 255:

$$\text{GF (256)} = \{0, 1, 2 \ldots 255\}. \tag{4}$$

Each element of GF(256) can be written as an 8-bit sequence, as a polynomial, and also, with an exponent of 3 (if it is not null), which is a primitive element of this field.

For example, 23 is expressed as the octet 00010111, it has the associated polynomial $x^4 + x^2 + x + 1$ and it is equal to 3^{239} on GF (256).

Addition and multiplication are inner operations of the algebraic finite field.

On GF, addition of two numbers, *a* and *b*, is made modulo-2, bit-by-bit, using the binary expression of the elements

$$s = a + b \Leftrightarrow s_i = a_i \oplus b_i, \text{ for } i = 0,\ldots,m. \tag{5}$$

Subtraction is made as the addition with the opposite element of the subtractive which is the element itself

$$c = a - b = a + (-b) = a + b. \tag{6}$$

On GF, multiplication of two elements is made multiplying their polynomials and finally, a primitive polynomial, $p(x)$, is used to reduce modulo-$p(x)$ the result

$$c(x) = \{a(x) \cdot b(x)\} \bmod p(x), \tag{7}$$

where $a(x) = \sum_{i=0}^{m-1} a_i x^i$, $b(x) = \sum_{i=0}^{m-1} b_i x^i$, $c(x) = \sum_{i=0}^{m-1} c_i x^i$ are the polynomials of the elements *a*, *b* and *c* of the GF.

On GF (256), we will use the primitive polynomial also called the polynomial number 285 having the set of binary coefficients: [1 0 0 0 1 1 1 0 1]

$$p(x) = x^8 + x^4 + x^3 + x^2 + 1. \tag{8}$$

Other primitive polynomials may also be used.

Division of an element a by b is computed as the multiplication of a with the inverse element b^{-1} which exists for all not-null elements of the field

$$c = a \cdot b^{-1}, \text{ with } b \neq 0 \text{ and } b \cdot b^{-1} = 1. \tag{9}$$

Other algebraic operations are defined on GFs, including exponentiation, logarithm, matrix operations and so on.

3. AES Algorithm Description

AES is a block code, defined on 128, 192 or 256-bit words, according to the length of the encryption key and denoted as AES-128, AES-192, and AES-256.

AES operates on the GF with 256 elements, denoted GF (256).

Each data word is written as a sequence of 16, 24 or 32 bytes, which become the elements of the data-matrix, defined on the 8-bit GF (256). So, AES is applied on a 4 × 4, 4 × 6 or 4 × 8 data matrix.

The serial input data stream is partitioned into blocks of 8 bits, which represent elements of the finite algebraic Galois Field GF (256). These elements are grouped into state-matrices, with 4 rows and 4 columns

$$S = \begin{bmatrix} s_{00} & s_{01} & s_{02} & s_{03} \\ s_{10} & s_{11} & s_{12} & s_{13} \\ s_{20} & s_{21} & s_{22} & s_{23} \\ s_{30} & s_{31} & s_{32} & s_{33} \end{bmatrix}. \tag{10}$$

First step is a substitution process called the *SubBytes stage*. Each i^{th} block of 8 bits [b_i b_{i+1} b_{i+2} b_{i+3} b_{i+4} b_{i+5} b_{i+6} b_{i+7}] is non-linearly processed bit-by-bit, as modulo-2 sums of transmitted bits and some constant values, c_i, read from the constant byte 0x.63:

$$b_i' = b_i \oplus b_{(i+4)\bmod 8} \oplus b_{(i+5)\bmod 8} \oplus b_{(i+6)\bmod 8} \oplus b_{(i+7)\bmod 8} \oplus c_i. \tag{11}$$

This relation can be compactly written using a substitution 8-by-8 matrix (T) and the vector c = [0 1 1 0 0 0 1 1]. The substitution byte results as:

$$b' = b \times T + c. \tag{12}$$

All algebraic operations are made modulo-2.

Matrix T is

$$T = \begin{bmatrix} 1 & 0 & 0 & 0 & 1 & 1 & 1 & 1 \\ 1 & 1 & 0 & 0 & 0 & 1 & 1 & 1 \\ 1 & 1 & 1 & 0 & 0 & 0 & 1 & 1 \\ 1 & 1 & 1 & 1 & 0 & 0 & 0 & 1 \\ 1 & 1 & 1 & 1 & 1 & 0 & 0 & 0 \\ 0 & 1 & 1 & 1 & 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 & 1 & 1 & 1 & 0 \\ 0 & 0 & 0 & 1 & 1 & 1 & 1 & 1 \end{bmatrix}. \quad (13)$$

On the next step, the elements on each row of the state-matrix are permuted, as it follows:

$$S'_{16} = \begin{bmatrix} s_{00} & s_{01} & s_{02} & s_{03} \\ s_{13} & s_{10} & s_{11} & s_{12} \\ s_{22} & s_{23} & s_{20} & s_{21} \\ s_{31} & s_{32} & s_{33} & s_{30} \end{bmatrix}. \quad (14)$$

This permutation corresponds to the cryptographic technique of transposition, in the simple manner of right-hand shifting. This stage is usually called *Shift-Rows*.

Then, AES processes each column, using an invertible polynomial, *a(x)*, for column multiplication

$$a(x) = a_0 + a_1 x + a_2 x^2 + a_3 x^3. \quad (15)$$

Rjindael proposed a particular reducible polynomial, $a(x)$, for column multiplication with the coefficient vector [3, 1, 1, 2], but other polynomials of third degree may also be used.

Each column of the state-matrix is associated to a polynomial, $s(x)$, which is multiplied by the encoding polynomial, $a(x)$.

The elements of the state matrix are octets, so all the arithmetic operations are made on GF (256).

The polynomial multiplication is made modulo-$p(x)$, where $p(x)$ is the polynomial number 17

$$p(x) = x^4 + 1. \quad (16)$$

The coefficients of the resulting polynomial are written back into the state-matrix.

For AES-128, instead of multiplying the polynomial of each column with $p(x)$, we can compute the multiplication between the invertible matrix, A, associated to the polynomial, $a(x)$, and each column vector of the state-matrix S:

$$A = \begin{bmatrix} a_0 & a_1 & a_2 & a_3 \\ a_3 & a_0 & a_1 & a_2 \\ a_2 & a_3 & a_0 & a_1 \\ a_1 & a_2 & a_3 & a_0 \end{bmatrix}; \tag{17}$$

$$\overline{c_i} = A \cdot \overline{s_i}, \quad (i = \overline{0, N_c - 1}). \tag{18}$$

$\overline{s_i} = \begin{bmatrix} s_{i,0} \\ s_{i,1} \\ s_{i,2} \\ s_{i,3} \end{bmatrix}$ represents the i^{th} column of the state matrix.

This step corresponds to the multiplication between A and S,

$$C = A \otimes S. \tag{19}$$

All arithmetic operations between coefficients are made on GF (256).

We prefer using this compact computing way instead of the polynomial mode, in order to easily implement the algorithm.

The final step of the algorithm encodes the data with the encryption key.

The encryption key is written as a key-matrix (K), having the same dimensions as the state-matrix, (S), and then the sum of the matrices is computed on GF (256), according to the following relation:

$$E = K \oplus C. \tag{20}$$

Decryption involves the same steps, in reverted order, and uses the inverse matrix, A^{-1}

$$C = E \oplus K; \; D = A^{-1} \otimes C \text{ or } \overline{d_i} = A^{-1} \cdot \overline{c_i}, \quad (i = \overline{0, N_c - 1}). \tag{21}$$

After this, the inverse permutation of the elements on each row is made and finally the original bits of each byte results using the inverted matrix, T^{-1}.

4. Improving AES

To increase the robustness of the AES algorithm, we have to use longer encryption keys and larger data matrix (Scripcariu & Ciornei, 2008). To keep

the processing time at low values, we have to maintain unchanged the complexity of the AES algorithm.

The first modified AES algorithm (MAES) (Scripcariu & Frunză, 2012) works on GF (256), with data matrices having exactly 8 rows and a variable number of columns, (N_c): 6, 8, 12 and 16

$$S_{64} = \begin{bmatrix} s_{00} & s_{10} & s_{20} & s_{30} & s_{40} & s_{50} & s_{60} & s_{70} \\ s_{01} & s_{11} & s_{21} & s_{31} & s_{41} & s_{51} & s_{61} & s_{71} \\ s_{02} & s_{12} & s_{22} & s_{32} & s_{42} & s_{52} & s_{62} & s_{72} \\ s_{03} & s_{13} & s_{23} & s_{33} & s_{43} & s_{53} & s_{63} & s_{73} \\ s_{04} & s_{14} & s_{24} & s_{34} & s_{44} & s_{54} & s_{64} & s_{74} \\ s_{05} & s_{15} & s_{25} & s_{35} & s_{45} & s_{55} & s_{65} & s_{75} \\ s_{06} & s_{16} & s_{26} & s_{36} & s_{46} & s_{56} & s_{66} & s_{76} \\ s_{07} & s_{17} & s_{27} & s_{37} & s_{47} & s_{57} & s_{67} & s_{77} \end{bmatrix} \quad (22)$$

The same traditional steps are made using extended matrices. The encryption key can have an equivalent length of about 384, 512, 768 and 1,024 bits, and the modified algorithm is denoted according to it: MAES-384, MAES-512, MAES-768 and MAES-1,024.

Input data is processed as data blocks of 48, 64, 96 or 128 bytes, associated with the input polynomial:

$$s(x) = \sum_{i=0}^{7} \sum_{j=0}^{N_c-1} s_{ij} x^{ij} \ . \quad (23)$$

The first and the second steps of AES are similarly done by MAES, excepting the shift value which is increased, row-by-row, from 0 to 7.

The third step of MAES uses the following invertible polynomial:

$$a(x) = x^7 + 2x^6 + 3x^5 + 4x^4 + 5x^3 + 6x^2 + 7x + 8 \, . \quad (24)$$

The encryption polynomial has a higher order in comparison with the classic AES algorithm, considering the increased number of rows, from 4 to 8:

$$a(x) = \sum_{i=0}^{7} a_i x^i . \quad (25)$$

For column multiplication step, the following encryption invertible matrix, *A*, is applied:

$$A=\begin{bmatrix} 13 & 7 & 5 & 4 & 4 & 3 & 2 & 1 \\ 2 & 13 & 7 & 5 & 4 & 4 & 3 & 2 \\ 7 & 2 & 13 & 7 & 5 & 4 & 4 & 3 \\ 5 & 7 & 2 & 13 & 7 & 5 & 4 & 4 \\ 6 & 2 & 2 & 6 & 9 & 4 & 7 & 5 \\ 3 & 1 & 7 & 6 & 2 & 10 & 6 & 6 \\ 5 & 4 & 4 & 3 & 2 & 1 & 8 & 7 \\ 10 & 6 & 7 & 4 & 3 & 2 & 1 & 8 \end{bmatrix}. \tag{26}$$

Its determinant, computed with GF (256) using some Matlab functions (Hahn & Valentine, 2010), is equal to 242.

It is more efficient to use a secret encryption polynomial, with a set of coefficients deduced based on a password of the user or on the encryption key. If the resulting polynomial is not-invertible, a unit will be added to it and the problem is solved. Even if this procedure is more secure, it needs more time to compute the inverse matrix for the decoder.

For multiplication, the minimal polynomial of the primitive value 3 is used (see (8)).

The encoded polynomial is computed on GF (256), as

$$c(x)=\{a(x)\cdot s(x)\}\{\mathrm{mod}[-p(x)]\}=\sum_{i=0}^{7}c_i x^i. \tag{27}$$

Its coefficients are derived and eq. (18) is rewritten using the invertible encryption matrix A of 8 × 8 elements, a vector corresponding to the column of the state-matrix S and the output encoded column-vector

$$\bar{c}=A\cdot\bar{s}. \tag{28}$$

Relation (20) is finally applied for MAES, as AES does.

Same steps are followed by MAES in comparison with AES, but the dimensions of all the arrays are increased. The same GF (256) is used by MAES and AES.

The modified algorithm makes for each input data block an increased number of arithmetical operations in comparison with AES, but the number of operations-per-symbol (ops) remains the same since larger data structures are processed by MAES.

Another way to improve AES is to work on a larger GF. We propose to use GF (2^{16}) which operates on 16-bit sequences (Scripcariu & Bogdan, 2006).

If the same 8-by-8 state matrix is used by the second modified AES block-code, denoted M2AES, then the dimension of the input block must be of about 1,024 bits. The encryption key will have 1,024 bits.

Each 16-bit sequence consists of two bytes, MSB (Most Significant Byte) and LSB (Least Significant Byte).

The substitution step is made using relation (10), applied on each byte (MSB and LSB) of each input value.

The second step of M2AES uses a permutation method, similar to MAES.

The third step will apply a 16-coefficient encoding polynomial which corresponds to an invertible matrix on GF (2^{16}).

Finally, the encryption key is added to the state-matrix.

The encryption key length is substantially increased using enlarged data structures and a higher GF (Table 1).

Table 1
Dimensions Used by AES-128, MAES and M2AES

State matrix dimensions	GF dimension	Encryption key length, [bits]
4-by-4	256	128
	65,536	256
8-by-8	256	512
	65,536	1,024
16-by-16	256	2,048
	65,536	4,096

Increasing the state-matrix dimension and the number of bits for the elements of the GF allow us to use longer encryption key and to improve the robustness of the algorithm.

It is recommended to use dedicated processors for a specific GF to increase the computing speed and to reduce the encryption/decryption time.

5. Conclusion

Larger data structures can be used by AES in order to increase its robustness. Operating on a larger Galois Fields represents an advantage for the algorithm robustness. Improvements of the traditional AES algorithm are presented. The modified AES algorithms, working on GF (256) or GF (65,536), with longer encryption keys (up to 4,096 bits) can be applied with very good time-performances using dedicated and faster processors.

REFERENCES

Hahn B.D., Valentine D.T., *Essential MATLAB for Engineers and Scientists, 4e*. Acad. Press, Burlington (MA), 2010.

Proakis J.G., Manolakis D.G., *Introduction to Digital Signal Processing*. MacMillan Publ. Co., NY, 1988.

Schneier B., *Applied Cryptography*. Sec. Ed., John Wiley & Sons, Inc., NY, 1996.

Scripcariu L., Alistar A., Frunză M.D., *JAVA Implemented Encryption Algorithm*. Proc. of the 8th Internat. Conf. "Develop. a. Appl. Syst.", Suceava, DAS, 2006, 424-429.

Scripcariu L., Bogdan, I., *Extended Advanced Encryption Standard Algorithm for Unicode Set*. ECIT, Iaşi, Sept. 20-23, 2006, 289-294.

Scripcariu L., Ciornei S., *Improving the Encryption Algorithms Using Multidimensional Data Structures*. Proc. of the Third Europ. Conf. on the Use of Modern Inform. a. Commun. Technol., ECUMICT, 2008, Gent (Belgium), 375-384.

Scripcariu L., Frunză M.D., *Modified Advanced Encryption Standard*. 11th Internat. Conf. "Develop. a. Appl. Syst.", Suceava, Romania, 2012, 87-90, http://www.dasconference.ro/cd2012/data/papers/B51.pdf .

ÎMBUNĂTĂȚIREA ALGORITMULUI AES FOLOSIND STRUCTURI DE DATE MAI MARI

(Rezumat)

Securizarea informației reprezintă un aspect important în sistemele de comunicații actuale. Pentru asigurarea confidențialității informațiilor transmise, se folosesc diferiți algoritmi de criptare: algoritmi cu cheie-publică precum RSA şi El-Gamal, dar şi algoritmi cu cheie-secretă, cum sunt DES, TDES şi AES. La ora actuală, AES este considerat cel mai puternic algoritm de criptare, fiind folosit în aplicații guvernamentale, militare dar şi civile. Lungimea cheii de criptare folosite de AES trebuie mărită pentru a creşte robustețea algoritmului. Pentru aceasta, se pot folosi structuri de date de dimensiuni mai mari sau se poate opera în câmpuri Galois cu elemente definite pe un număr mai mare de biți. În lucrare, sunt prezentați algoritmi AES modificați, cu chei de criptare de până la 4096 de biți.