FAST ENCRYPTION FOR MULTIMEDIA

Xun Yi, Chik How Tan, Chee Kheong Siew and Mahbubur Rahman Syed

Abstract— **The security of multimedia data is important for multimedia commerce. The encryption algorithms with high security, such as DES and IDEA, may not be suitable to multimedia applications because large data sizes and real time constraint.**

This paper proposes a fast encryption algorithm for multimedia data, called FEA-M. FEA-M is based on Boolean matrix theory. The plaintext and the ciphertext are 64×64 Boolean matrices while the secret key is also a 64×64 matrix. The structure of FEA-M is chosen to provide confusion and diffusion and to facilitate both hardware and software implementation.

Keywords—**Encryption, multimedia, Boolean matrix.**

I. Introduction

THE security of multimedia data is important for multimedia commerce. For example, in video on demand and video conferencing applications, it is desirable that only those who have paid for the services can view their video or movies.

Authentication control mechanisms can be used to secure distributed multimedia applications. However, it is not enough to secure multimedia data broadcast on wireless, satellite or Mbone networks. Multimedia data is still needed to be encrypted during transmission.

Encryption algorithms can be divided into two basic classes - secret-key and public-key encryption algorithms. They have distinct characteristics and are used in different ways to provide security services.

Secret-key encryption algorithms have been in use in commercial networks since the early 1970s. The U. S. Data Encryption Standard (DES)[1] is the first secret-key encryption algorithm which has had its full specification published as a public standard. It was developed at IBM in 1976. It encrypts 64-bit blocks of data with a 56-bit key. Considering that there is disagreement over whether a 56-bit key is sufficiently strong, a number of secret-key encryption algorithms have been proposed to replace DES in recent years. The International Data Encryption Algorithm (IDEA) [2], developed by Xuejia Lai and Jame Massey in 1990, is one of them. IDEA encrypts 64-bit blocks of data with 128-bit key.

Recently, the U. S. National Institute of Standards and Technology (NIST) is organizing the international competition in a drive to develop an Advanced Encryption Standard (AES) to protect sensitive information in federal computer systems. The candidates include Serpent[3], Mars[4], Rijndael[5], Crypton[6], RC6[7], Twofish[8] and etc. On October 2, 2000, NIST announces that Rijndael has been selected as the proposed AES.

Another class of secret-key encryption algorithm is the stream cipher which uses a short key to generate the key-stream to encrypt a digital data stream one bit at a time.

Public-key encryption algorithm provides a radical departure from all that has gone before. For one thing, public-key algorithms are based on mathematical functions rather than on substitution and permutation. More important, it is asymmetric involving the use of two separate keys, in contrast to symmetric secret-key algorithm which uses only one key. The use of two keys has profound consequences in the areas of confidentiality, key distribution and authentication. RSA[9], developed by Ron Rivest, Adi Shamir and Len Adleman at MIT in 1978, is the first public-key encryption algorithm. It encrypts 1024-bit blocks of data at a time.

State-of-the-art public-key encryption algorithm with high security transmission performance require high processing resources when applied to high bit-rates and result not suitable for modern multimedia communications. Although existing secret-key encryption algorithms, such as DES, operate much faster than public-key algorithms, they are very complicated and involves large computations. A software DES implementation is not fast enough to process the vast amount of data generated by multimedia applications and a hardware DES implementation (a set-top box) adds extra costs both to broadcasters and to receivers.

The challenges of multimedia data encryption come from two facts. Firstly, multimedia data size usually are very large. Secondly, multimedia data needs to be processed in real time. Encryption algorithms with high security will put great burden on storage space requirement and increase latency. For some commercial applications, such as pay-per-view, very expensive attack of the encrypted multimedia data are not interesting to adversary [10], because most multimedia data are different from military secrets or financial information. For multimedia applications, information rate is very high, but the information value is very low. To break such encryption code is much more expensive than to buy the programs.

The study related to fast video encryption can be found in [10][11][12]. These algorithms are only applied to video data.

In this paper, we propose a fast encryption algorithm for multimedia data, called FEA-M. The mathematical principle of FEA-M is founded on Boolean matrix theory. FEA-M encrypts 64×64 Boolean plaintext matrices by a 64×64 Boolean key matrix. The structure of FEA-M is chosen to provide confusion and diffusion and to facilitate both hardware and software implementation. The security of FEA-

Xun Yi, Chik How Tan, Chee Kheong Siew are with the Information and Communication Institute of Singapore, School of EEE, Nanyang Technological University, Singapore 639798, e-mail: exyi@ntu.edu.sg.

Mahbubur Rahman Syed is with the Department of Computer and Information Sciences, 273 Wissink Hall, Minnesota State University, Mankato, MN 56001, USA.

Contributed Paper

Original manuscript received November 21, 2000

M is based on the difficulty of solving non-linear equation groups and variable linear equation groups. The following sections are arranged as follows: Section II introduces some basic concepts and principle of Boolean matrix. Section III describes the fast encryption algorithm for multimedia data, FEA-M. Section IV discusses features of FEA-M. Section V deals with implementation of FEA-M. Section VI compares computation complexities of some existing encryption algorithms with FEA-M. Conclusion is drawn in the last section.

II. Boolean matrix

In the following discussion, we consider **Boolean matrix set**

$$\mathbf{B} = \{M | M = (a_{ij})_{m\times n}, a_{ij} \in GF(2), m, n \in Z^*\}$$

where $Z^* = \{1, 2, \cdots\}$ and $GF(2) = \{0, 1\}$ in which addition and multiplication are defined as follows:

$$\begin{array}{ll} 0 \oplus 0 = 0 & 0 \wedge 0 = 0 \\ 0 \oplus 1 = 1 & 0 \wedge 1 = 0 \\ 1 \oplus 0 = 1 & 1 \wedge 0 = 0 \\ 1 \oplus 1 = 0 & 1 \wedge 1 = 1 \end{array}$$

It is not difficult to verify that the following distributive property holds.

$$(a \oplus b) \wedge c = (a \wedge c) \oplus (b \wedge c)$$
$$a \wedge (b \oplus c) = (a \wedge b) \oplus (a \wedge c)$$

for any $a, b, c \in GF(2)$.

On basis of the above definitions, **Boolean matrix addition** and **Boolean matrix multiplication** are defined as follows:

For any $A = (a_{ij})_{m\times n}, B = (b_{ij})_{m\times n}, C = (c_{ij})_{n\times l} \in \mathbf{B}$,

$$A \oplus B = (a_{ij}) \oplus (b_{ij}) = (a_{ij} \oplus b_{ij}) \quad (1)$$
$$A \cdot C = (a_{ij}) \cdot (c_{ij}) = (\bigoplus_{1\le k\le n} a_{ik} \wedge c_{kj}) \quad (2)$$

where

$$\bigoplus_{1\le k\le n} a_{ik} \wedge c_{kj} = (a_{i1} \wedge c_{1j}) \oplus (a_{i2} \wedge c_{2j}) \oplus \cdots \oplus (a_{in} \wedge c_{nj})$$

Note that $A \cdot C$ is an $m \times l$ matrix. Usually, $A \cdot C \neq C \cdot A$ even if $m = n = l$.

Different from addition and multiplication of general matrices, addition and multiplication of Boolean matrices has the following property:

Property 1: For any $A = (a_{ij})_{m\times n} \in \mathbf{B}$, $A + A = 0$, i.e., $-A = A$.

The reason is $A \oplus A = (a_{ij} \oplus a_{ij}) = (0)$.

Property 2: Suppose $A = (a_{ij})_{m\times n}, B = (b_{ij})_{n\times l} \in \mathbf{B}$, $C = A \cdot B = (c_{ij})_{m\times l}$, $B_p = (b_{p1}, b_{p2}, \cdots, b_{pl})$, $C_q = (c_{q1}, c_{q2}, \cdots, c_{ql})$ where $p = 1, 2, \cdots, n$ and $q = 1, 2, \cdots, m$, then

$$C_q = \bigoplus_{p\in\{p|a_{qp}=1, 1\le p\le n\}} B_p \quad (3)$$

Proof: On basis of the multiplication definition of Boolean matrices, we have

$$\begin{aligned} c_{qj} &= \bigoplus_{1\le k\le l} a_{qk} \wedge b_{kj} \\ &= (a_{q1} \wedge b_{1j}) \oplus (a_{q2} \wedge b_{2j}) \oplus \cdots \oplus (a_{qn} \wedge b_{nj}) \\ &= \bigoplus_{p\in\{p|a_{qp}=1,1\le p\le n\}} b_{pj} \end{aligned}$$

Let $P_q = \{p | a_{qp} = 1, 1 \le p \le n\}$, Therefore,

$$\begin{aligned} C_q &= (c_{q1}, c_{q2}, \cdots, c_{ql}) \\ &= (\bigoplus_{p\in P_q} b_{p1}, \bigoplus_{p\in P_q} b_{p2}, \cdots, \bigoplus_{p\in P_q} b_{pl}) \\ &= \bigoplus_{p\in P_q} (b_{p1}, b_{p2}, \cdots, b_{pl}) \\ &= \bigoplus_{p\in P_q} B_p \end{aligned}$$

The **inverse** of a Boolean matrix is defined as follows:

An $n \times n$ Boolean matrix A is **invertible** (or **nonsingular**) if there exists an $n \times n$ Boolean matrix B such that

$$A \cdot B = B \cdot A = I_n$$

where I_n is the identity matrix of order n. If A is an invertible matrix, then its inverse is unique. We denote the inverse of A by A^{-1}.

In $\mathbf{B}$, there are two **elementary row operations**

- Interchange two rows.
- Add one row to another.

and two **elementary column operations**

- Interchange two columns.
- Add one column to another.

An $n \times n$ matrix is called a **elementary matrix** if it can be obtained from I_n by a single elementary row or column operation. Same as general matrices, Boolean elementary matrices have the following properties:

Property 3: Let E be an elementary matrix obtained by performing an elementary row operation on I_m and F be an elementary matrix obtained by performing an elementary column operation on I_n. If the same row operation is performed on an $m \times n$ Boolean matrix A, then the resulting matrix is given by the product $E \cdot A$. If the same column operation is performed on A, then the resulting matrix is given by the product $A \cdot F$.

Property 4: A square Boolean matrix A is invertible if and only if it can be written as the product of elementary matrices.

Different from general matrices, Boolean elementary matrices have the following special properties:

Property 5: If E is a Boolean elementary matrix of order n, then E^{-1} exists and is E.

It is not difficult to prove $E \cdot E = I_n$ for any Boolean elementary matrix E of order n.

Property 6: If a $n \times n$ Boolean matrix $A = E_k \cdots E_2 \cdot E_1 \cdot I_n$, then $A^{-1} = I_n \cdot E_1 \cdot E_2 \cdots E_k$.

This is because $(E_k \cdots E_2 \cdot E_1 \cdot I_n) \cdot (I_n \cdot E_1 \cdot E_2 \cdots E_k) = I_n$ according to Property 5.

In Property 6, although the results of $E_1 \cdot I_n$ and $I_n \cdot E_1$ are same, the two operations are different. If $E_1 \cdot I_n$ represents adding the ith row to the jth row in I_n, $I_n \cdot E_1$ means adding the jth column to the ith column in I_n. If $E_1 \cdot I_n$ represents interchanging the ith row and the jth row in I_n, $I_n \cdot E_1$ means interchanging the jth column and the ith column in I_n.

III. Description of fast encryption algorithm for multimedia (FEA-M)

A. Encryption and decryption model

FEA-M is designed on basis of Shannon secure communication system model [13] which is shown in Fig.1.

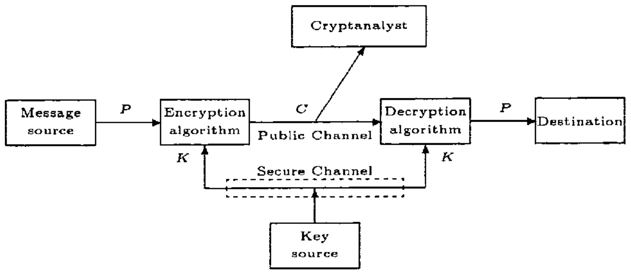

Fig.1. Shannon secure communication system model

With the message P and the encryption key K as input, the encryption algorithm forms the ciphertext C. We can write this as

$$C = E_K(P) \quad (4)$$

The intended receiver, in possession of the key, is able to invert the transformation:

$$P = D_K(C) \quad (5)$$

In Fig.1, the encryption and decryption keys may be different. However, the decryption key should be easily determined by the encryption key.

An opponent, observing C but not having access to K or P, may attempt to recover P or K or both P and K. It is assumed that the opponent knows the encryption algorithm E and decryption algorithm D.

In the following discussion, we only consider $n \times n$ Boolean matrices when $n = 64$. Suppose the sender and the receiver share a secret $n \times n$ invertible **master key matrix** K_0 in which the number of elements with value 1 is around $\frac{n^2}{2}$. Therefore, the key length of FEA-M is 4096 bits.

The description of master key matrix distribution is out of the arrange of this paper. Public-key encryption algorithm RSA may be used to distribute the master key matrix.

B. Key hierarchy

The key hierarchy of FEA-M is depicted in Fig.2.

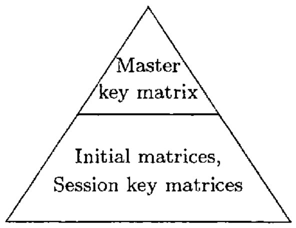

Fig.2. Key hierarchy of FEA-M

As shown in Fig.2, the key hierarchy of FEA-M has two levels. On the top is one master key matrix. On the bottom are initial matrices and session key matrices. Master key matrix is used to protect initial matrices and session key matrices. Initial matrices and session key matrices are used to protect transmission multimedia data.

C. Session key matrix generation

Initially, the sender is required to generate session key in order to encrypt plaintext message. Session key is an invertible matrix of order n, denoted as K.

The session key matrix K and its inverse K^{-1} can be randomly generated from the identity matrix I_n in the following program:

(1) Let $i = 1$ and $K_1 = K_2 = I_n$.
(2) Write the $n \times 2n$ matrix that consists of K_1 on the left and K_2 on the right to obtain $(K_1 \| K_2)$.
(3) Randomly choose integers $i_1, i_2, \cdots, i_{\frac{n}{2}-1}$ from the set $S_i = \{1, 2, \cdots, n\} - \{i\}$.
(4) On the left, add the ith row to the i_1th row, i_2th row, $\cdots$, $i_{\frac{n}{2}-1}$th row respectively in K_1 to produce a new matrix K_1'.
(5) On the right, add the i_1th column, i_2th column, $\cdots$, $i_{\frac{n}{2}-1}$th column to the ith column respectively in K_2 to produce a new matrix K_2'.
(6) Let $i = i + 1$, $K_1 = K_1'$ and $K_2 = K_2'$. If $i \le n$, then go to (3).
(7) Randomly choose integers i from the set $S_i = \{1, 2, \cdots, n-1\}$.
(8) On the left, interchange the nth row and the ith row in K_1.
(9) On the right, interchange the ith column and the nth column in K_2.
(10) Finally, output matrices K_1 and K_2.

On basis of Property 3,

$$\begin{aligned} K_1 &= E_k \cdots E_2 \cdot E_1 \cdot I_n \\ K_2 &= I_n \cdot E_1 \cdot E_2 \cdots E_k \end{aligned}$$

According to Property 4, K_1 is an invertible Boolean matrix. Based on Property 6, $K_1^{-1} = K_2$.

The above program can be also used to generate master key matrix K_0 and its inverse.

D. Initial matrix generation

Besides the session key matrix, the sender is required to randomly generate an initial Boolean matrix V_0. Each element of V_0 is randomly chosen from $GF(2)$ so that the distribution of 0 and 1 in V_0 obeys the uniform distribution. In view of it, the number of the elements with value 1 in V_0 is around $\frac{n^2}{2}$.

E. Session key and initial matrices distribution

By using the master key matrix K_0, the inverse of the session key matrix K and the initial matrix V_0 can be distributed from the sender to the receiver in the following way:

At the sender side, she computes

$$K^* = K_0 \cdot K^{-1} \cdot K_0 \quad (6)$$
$$V^* = K_0 \cdot V_0 \cdot K_0 \quad (7)$$

then sends (K^*, V^*) to the receiver.

At the receiver side, he recovers K^{-1} and V_0 by computing

$$K^{-1} = K_0^{-1} \cdot K^* \cdot K_0^{-1} \quad (8)$$
$$V_0 = K_0^{-1} \cdot V^* \cdot K_0^{-1} \quad (9)$$

F. Encryption and decryption

At first, the plaintext message should be divided into a series of blocks $P_1, P_2, \cdots, P_r$ with same length n^2. If the length of the last block P_r is less than n^2, we need append some 0s in it so that its length is right n^2. The n^2 bits of each block are arranged as a square matrix of order n.

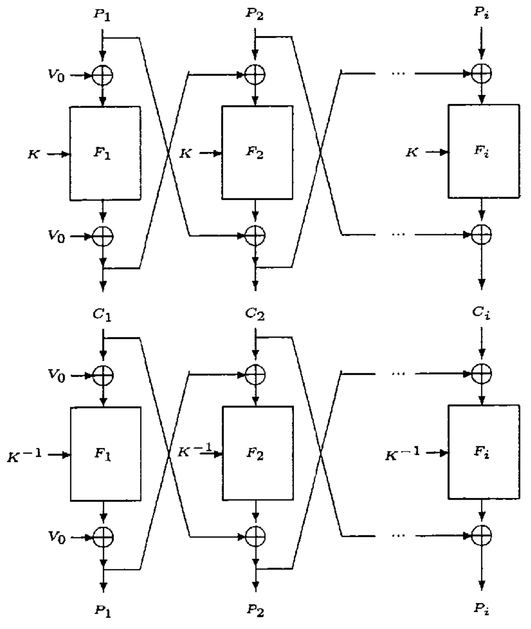

Fig.3. Illustration of encryption and decryption process

The encryption and decryption processes are depicted in Fig.3, in which the function F_i means

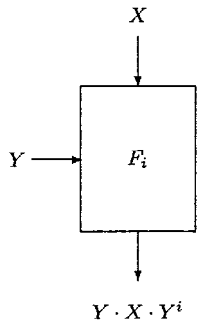

Fig.4. Illustration of the function F_i

Each plaintext matrix P_i is encrypted into ciphertext C_i in the following way:

$$\begin{aligned} C_1 &= K \cdot (P_1 \oplus V_0) \cdot K \oplus V_0 & (10) \\ C_2 &= K \cdot (P_2 \oplus C_1) \cdot K^2 \oplus P_1 \\ \cdots\ & \cdots\ \cdots\cdots\cdots \\ C_i &= K \cdot (P_i + C_{i-1}) \cdot K^i \oplus P_{i-1} & (11) \end{aligned}$$

Each corresponding ciphertext matrix C_i is decrypted into plaintext P_i in the following way:

$$\begin{aligned} P_1 &= K^{-1} \cdot (C_1 \oplus V_0) \cdot K^{-1} \oplus V_0 & (12) \\ P_2 &= K^{-1} \cdot (C_2 \oplus P_1) \cdot K^{-2} \oplus C_1 \\ \cdots\ & \cdots\ \cdots\cdots\cdots \\ P_i &= K^{-1} \cdot (C_i \oplus P_{i-1}) \cdot K^{-i} \oplus C_{i-1} & (13) \end{aligned}$$

G. Update of session key matrix

For the sake of security, the session key matrix should be updated regularly during the encryption of multimedia data. The update of session key matrix could be based on how important the multimedia contents is. If the multimedia contents is very important, the session key matrix should be updated frequently. For the multimedia contents with very low value, only one session key may be enough. Therefore, in FEA-M, the update of session key matrix is flexible.

The session key matrix is updated in the following program:

(1) The sender randomly generates a Boolean matrix in the same way as generating the initial Boolean matrix V_0 and replaces the first row with 64-bit special code which represents session key update to obtain V. Then she encrypts V into ciphertext

$$C_{i+1}^* = K \cdot (V + C_i) \cdot K^i \oplus P_i$$

and sends C_{i+1}^* to the receiver.

(2) The receive recovers V from C_{i+1}^* by computing

$$V = K^{-1} \cdot (C_{i+1}^* + P_i) \cdot K^{-i} \oplus C_i$$

and detects the 64-bit special code for session key update in V.

(3) The sender randomly generates a new session key matrix K' and its inverse. Then she sends

$$K^* = K_0 \cdot K'^{-1} \cdot K_0$$

to the receiver.

(4) The receiver recovers K'^{-1} from K^* by computing

$$K'^{-1} = K_0^{-1} \cdot K^* \cdot K_0^{-1}$$

(5) Both the sender and the receiver replace the first row of V with

$$V_1 = \bigoplus_{j \in \{j | k_{1j}=1, 1 \le j \le n\}} V_j \quad (14)$$

where $(k_{11}, k_{12}, \cdots, k_{1n})$ is the first row of the old session key matrix K and V_j denotes the jth row of V.

After update of session key, the plaintext matrices after P_i are encrypted by the new session key matrix K' in the following way:

$$\begin{aligned} C_{i+1} &= K' \cdot (P_{i+1} + V) \cdot K' + V \\ C_{i+2} &= K' \cdot (P_{i+2} + C_{i+1}) \cdot K'^2 + P_{i+1} \\ \cdots &\cdots \cdots\cdots\cdots\cdots \end{aligned}$$

The ciphertext matrices are decrypted by K'^{-1} as follows:

$$\begin{aligned} P_{i+1} &= K'^{-1} \cdot (C_{i+1} + V) \cdot K'^{-1} + V \\ P_{i+2} &= K'^{-1} \cdot (C_{i+2} + P_{i+1}) \cdot K'^{-2} + C_{i+1} \\ \cdots &\cdots \cdots\cdots\cdots\cdots \end{aligned}$$

IV. Features of FEA-M

A. Similarity of encryption and decryption

In order to facilitate implementation, encryption and decryption algorithms should satisfy similarity of encryption and decryption.

The similarity of encryption and decryption means the decryption is essentially the same process as encryption, the only difference being that different key subblocks are used.

In FEA-M, suppose $P_1, P_2, \cdots, P_i$ have been transformed to $C_1, C_2, \cdots, C_i$ with K according to Formulae (10) and (11). Let us see what will happen when the same transformations with K^{-1} are performed on $C_1, C_2, \cdots, C_i$.

$$\begin{aligned} & K^{-1} \cdot (C_1 + V_0) \cdot K^{-1} + V_0 \\ = & K^{-1} \cdot (K \cdot (P_1 + V_0) \cdot K + V_0 + V_0) \cdot K^{-1} + V_0 \\ = & (P_1 + V_0) + V_0 \\ = & P_1 \end{aligned}$$

$$\begin{aligned} & K^{-1} \cdot (C_2 + P_1) \cdot K^{-2} + C_1 \\ = & K^{-1} \cdot (K \cdot (P_2 + C_1) \cdot K^2 + P_1 + P_1) \cdot K^{-2} + C_1 \\ = & (P_2 + C_1) + C_1 \\ = & P_2 \\ & \cdots\cdots\cdots\cdots \\ & K^{-1} \cdot (C_i + P_{i-1}) \cdot K^{-i} + C_{i-1} \\ = & K^{-1} \cdot (K \cdot (P_i + C_{i-1}) \cdot K^i + P_{i-1} + P_{i-1}) \cdot K^{-i} + C_{i-1} \\ = & (P_i + C_{i-1}) + C_{i-1} \\ = & P_i \end{aligned}$$

The above results show that the encryption and decryption process of FEA-M are same except that the different key matrix are used. Therefore, FEA-M satisfies the similarity of encryption and decryption.

B. Diffusion and confusion

Confusion [13][14] means that the ciphertext depends on the plaintext and key in a complicated and involved way. The diffusion requirement on a cipher is that each plaintext bit should influence every ciphertext bit and each key bit should influence every ciphertext bit [13][14]. Confusion and diffusion are two basic design criteria of secret-key encryption algorithm.

In FEA-M, suppose $K = (k_{lm})_{n \times n}$, $V = (v_{lm})_{n \times n}$, $P_i = (p_{lm}^{(i)})_{n \times n}$, $C_i = (c_{lm}^{(i)})_{n \times n}$ and $K^i = (k_{lm}^{(i)})_{n \times n}$. According to Formulae (10) and (11), we know

$$c_{lm}^{(1)} = (\bigoplus_{1 \le h \le n} k_{lh} \wedge \bigoplus_{1 \le g \le n} ((p_{hg}^{(1)} \oplus v_{hg}) \wedge k_{gm})) \oplus v_{lm} \quad (15)$$

For $i \ge 2$,

$$\begin{aligned} c_{lm}^{(i)} &= (\bigoplus_{1 \le h \le n} k_{lh} \wedge \bigoplus_{1 \le g \le n} ((p_{hg}^{(i)} \oplus c_{hg}^{(i-1)}) \wedge k_{gm}^{(i)})) \oplus p_{lm}^{(i-1)} \\ &= (\bigoplus_{1 \le h \le n} \bigoplus_{1 \le g \le n} (k_{lh} \wedge k_{gm}^{(i)} \wedge (p_{hg}^{(i)} \oplus c_{hg}^{(i-1)})) \oplus p_{lm}^{(i-1)} \quad (16) \end{aligned}$$

In Equation (15), although $c_{lm}^{(1)}$ only depends on the lth row and the mth column of key matrix K, it does not matter because a random bit v_{lm} is involved.

From Equation (16), we can see that $c_{lm}^{(i)}$ depends on all elements of plaintext matrix P_i and previous ciphertext C_{i-1}. In turn, all elements of C_{i-1} rely on all elements of key matrix K. Therefore, $c_{lm}^{(i)}$ also depends on all elements of K. It means FEA-M satisfies the diffusion design criterion if we do not consider the first encryption.

In addition, we can find that the relationship between $c_{lm}^{(i)}$ and each element of K is non-linear. Although the relationship between $c_{lm}^{(i)}$ and each element of P_i is linear, it is variable with the introduction of $k_{gm}^{(i)}$ where $g = 1, 2, \cdots, n$. Hence, FEA-M satisfies the confusion design criterion.

V. Implementation of FEA-M

Since FEA-M satisfies the similarity of encryption and decryption, we only need to implement the encryption algorithm of FEA-M. In FEA-M, the most time-consuming operation is the multiplication of Boolean matrices. In the following discussion, we will study software and hardware implementation of Boolean matrix multiplication.

A. Software implementation

Suppose $A = (A_i)_{n \times 1}$ where $A_i = (a_{i1}, a_{i2}, \cdots, a_{in})$, $B = (B_j)_{n \times 1}$ where $B_j = (b_{j1}, b_{j2}, \cdots, b_{jn})$ and $C = A \cdot B = (C_k)_{n \times 1}$ where $B_k = (b_{k1}, b_{k2}, \cdots, b_{kn})$. According to Property 2, the product C of A and B can be computed in the following program:

(1) Start
(2) $i = 1$.
(3) $j = 1, S = 0$.
(4) If $A_i \wedge 1 = 1$, $S = S \oplus B_{n-j+1}$.
(5) $j = j + 1$, $A_i >> 1$ (i.e., A_i is shifted 1 bit to the left).
(6) If $j \leq n$, go to (3).
(7) $C_i = S$, $i = i + 1$.
(8) If $i \leq n$, go to (2).
(9) Halt.

The above procedure can be depicted in Fig.5.

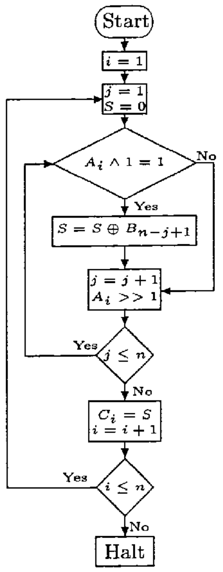

Fig.5. Software implementation of Boolean matrix multiplication

Because only 64-bit XOR operation is used in the above algorithm, it can be implemented efficiently in 64-bit processor.

B. Hardware implementation

The Boolean matrix multiplication can be also implemented by using linear feedback shift register (LFSR). In our hardware implementation, 129 64-bit LFSRs are needed to construct the Boolean matrix multiplication machine shown in Fig.6. The shift rate in the upper 64 64-bit LFSRs is 64 times of that in the lower 64 64-bit LFSRs. The lowest LFSR in the dashbox has the same rate as the upper 64 64-bit LFSRs.

The procedure of multiplication is as follows:

(1) Fill in elements of Boolean matrix $A = (a_{ij})_{n\times n}$ in the upper 64 64-bit LFSRs by row.
(2) Fill in elements of Boolean matrix $B = (b_{ij})_{n\times n}$ in the lower 64 64-bit LFSRs by column.
(3) The first stage content of each upper LFSR and corresponding first stage content of lower LFSR is ANDed.
(4) All the above results are XORed and sent to the last stage of the lowest LFSR.
(5) All upper LFSRs shift one position down each time. All lower LFSRs shift one position up only when the upper LFSRs shift the multiply of 64 times.
(6) Each 64 times when the lowest LFSR shifts to the right, the content of each stage is passed to the last stage of the lower 64 64-bit LFSRs.

After the 64 64-bit lower LFSRs shift up 64 times, the results in them are the product of A and B by column.

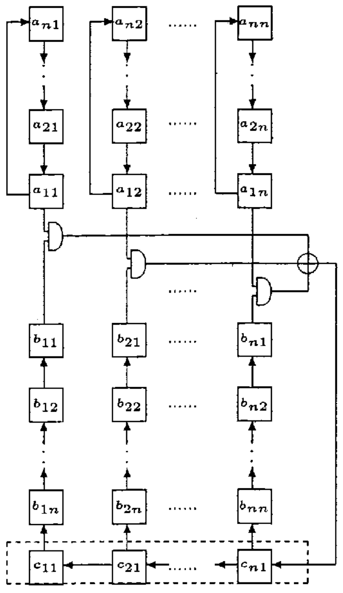

Fig.6. Hardware implementation of Boolean matrix multiplication

VI. Computation complexity comparison

In the following comparison, the target machine operations are micro-operations that is implemented in hardware in one clock cycle, with the exceptions of fetching data from cache/memory, and multiplication. These last two operations were assigned a latency of four clock cycles. Adequate atomic ALU data widths were assumed to be supported, up to 64 bits, at the ideal latency. Full pipelining of the fetch and multiply operations was expected. Multiple operations can be issued in a vector, VLIW, or data-flow manner.

The total number of micro-operations that are required in some encryption algorithms, counting vector operations as a single micro-operation, was investigated in [15]. The results are summarized in Tab.1 for the 128-bit key and 128-bit block size.

Cipher	Micro-operations
Rijndael	616
Crypton	736
Twofish	600
RC6	296
MARS	528
Cast256	824
Serpent	1328

Tab.1. Total number of micro-operations for the 128-bit key and 128-bit block size

As far as FEA-M is concerned, under the assumption that 0 and 1 of matrices obey uniform distribution, one encryption on 64×64 plaintext needs 64×2 XOR operation, two Boolean matrix multiplication operations and one exponent operation. In fact, $K^i = K^{i-1} \cdot K$. Therefore, the exponent operation K^i can be transferred into one multiplication of K^{i-1} and K if the previous exponent K^{i-1} is stored.

On basis of Property 2, one Boolean matrix operation needs $\frac{64}{2} \cdot 64$ XOR operations. Therefore, for 128-bit block size, the total number of micro-operations is

$$\frac{(3 \times \frac{64 \times 64}{2} + 2 \times 64) \cdot 128}{64 \times 64} = 196$$

It means FEA-M encrypts one bit with about 1.5 XOR operations. FEA-M is much faster encryption algorithm than others listed in Tab.1.

VII. Conclusion

We have developed a fast encryption algorithm for multimedia data, FEA-M. It encrypts 512 bytes of data with a 512 bytes of key at a time. Our cryptanalysis has shown that FEA-M satisfies basic confusion and diffusion design criteria. The structure of FEA-M facilitates both hardware and software implementations. Computation complexity comparison has shown that FEA-M is much faster encryption algorithm than others. It needs only about 1.5 XOR operations to encrypt one bit plaintext.

We believe FEA-M can be used to secure many multimedia applications, such as digital video and audio transmissions.

Multimedia data security is challenging. We hope interested parties can offer their valuable comments on FEA-M.

References

[1] "Data encryption standard", FIPS PUB 46, National Bureau of Standards, Washington, D.C., Jan. 1997.
[2] X. Lai, "On the design and security of block cipher", Konstanz, Germany: Hartung-Gorre, 1992.
[3] R. Anderson, E. Biham, and L. Knudsen, "Serpent: A Proposal for the Advanced Encryption Standard", 1998. AES submission.
[4] C. Burwick, D. Coppersmith, E. DAvignon, R. Gennaro, S. Halevi, C. Jutla, S. M. Matyas Jr., L. OConnor, M. Peyravi, D. Stafford, and N. Zunic, "MARS - a candidate cipher for AES", IBM Corporation, June 1998. AES submission.
[5] J. Daernen and V. Rijmen, "AES Proposal: Rijndael", June 1998. AES submission.
[6] C. H. Lim, "CRYPTON: A New 128-bit Block Cipher", 1998. AES submission.
[7] R. L. Rivest, M.J.B. Robshaw, R. Sidney, and V. L. Yin, "The RC6 Block Cipher", 1998. AES submission.
[8] B. Schneier, J. Kelsey, D. Whiting, D. Wagner, C. Hall, and N. Ferguson, "Twofish: A 128-Bit Block Cipher", June 1998. AES submission.
[9] R. Rivest, A. Shamir, L. Adleman, "A method for obtaining digital signatures and public key cryptosystem", Communication of ACM, Feb. 1978.
[10] B. Macq, J. Quisquater, "Cryptology for digital TV broadcasting", Proc. of IEEE, 83(6), 1995.
[11] L. Tang, "Methods for encrypting and decrypting MPEG video data efficiently", Proc. of the ACM Multimedia'96, Boston, Nov. 1996.
[12] C. Shi, B. Bhargava, "Light-weight MPEG video encryption algorithm", Proc. of Multimedia'98, Jan. 1998.
[13] C. E. Shannon, "Communication theory of secret systems", Bell Syst. Tech. J., Vol.28, 1949.
[14] J. L. Massey, "An introduction to contemporary cryptology", Proc. IEEE, Vol.76, No.5, May 1988.
[15] G. Graunke, "Yet another performance analysis of the AES candidates", http://csrc.nist.gov/encryption/aes/round1 /pubcmnts.htm.