International Journal of Enhanced Research in Science Technology & Engineering, ISSN: 2319-7463
Vol. 3 Issue 1, January-2014, pp: (125-130), Impact Factor: 1.252, Available online at: www.erpublications.com

An Enhanced Data Security Method Using MIPS Encryption Algorithm (MEA)

Dr. Rajiv Srivastava
Director, Ph.D (Computer Science Engineering)
Sagar Institute of Research & Technology, India

Abstract: The MEA is an integral approach of block cipher and transposition cipher method. It takes 64 bit plain text input and produces 64 bit cipher text as in IDEA [13] with modified key schedule to avoid possibilities of weak keys. It further makes transposition of the 64 bit cipher text to 128 bit end cipher text for disk storage. The increased length of end cipher text is a trade-off between the degree of increased security to SI and the nominal cost of storage media in the present state_of_the_art development.

Keywords: Encryption, Decryption, Block cipher, Transposition cipher, Encryption Keys, Decryption Keys.

1. INTRODUCTION

Since computers can be used to quickly break naïve cryptosystems one should use encryption algorithms that are free from and mathematical weaknesses and that are computationally infeasible to break by making cracking more time consuming. At the same time, the computational complexity of encryption and decryption should be within reasonable limits because they represent processing overheads also.

One algorithm that is believed to provide a reasonable compromise among these requirements is based on the Data Encryption Standards (DES) [4,5,10]. For the past 20 years, the best security most of us have heard about has been provided by DES. Although there has been a weakness of hidden trapdoors through s-boxes in DES [6,10], still it has been a good and secure algorithm against the mid seventies technologies. Now with the advent of high speed computers it is facing more criticism for not providing enough security because of its 56 bit key size.

Some other algorithms gave also been developed in due course of time such as An Application of Chinese Remainder Theorem to Multiple-key encryption in Database Systems[3] and A High Performance Encryption Algorithm [11] etc.. These algorithms also face criticism for break due to existing break methods like Brute force, Linear and Differential cryptanalytic methods [14] and the development of high processing speed of computers.

In order to avoid any cryptanalytic attack on cipher text because of small key length in DES [6,10] another well known algorithm IDEA (International Data Encryption Algorithm) [13], on 128 bit key with a block cipher method has been developed. It provides a powerful encryption that resists to a break possibility arising from high speed of computers of today and advanced break methods [14]. This algorithm works on 64 bit plain text input and produces 64 bit cipher text. The design philosophy behind this algorithm is one of the mixing operations from different algebraic groups.

In comparison to DES, the algorithm IDEA seems to be a safer proposal because of its 128 bit key approach but how long it can stand to the challenges posed by cryptanalytic methods and increasing speed of computers is still a question. The security of a computing system is such a challenging field that it demands introduction of newer ideas everyday. The present encryption algorithm named as MIPS[1] Encryption Algorithm (MEA) is a step forward in this direction and provides further resistance to break than IDEA.

We present MIPS Encryption Algorithm (MEA) in section 2. The MEA security is described in section 3, and Conclusion and Discussion is presented in section 4.

[1] The Multilevel Information Protection System (MIPS) is an Information System which provides a relatively higher degree of security to a Sensitive Information (SI). The security to SI in MIPS is given by a MIPS Encryption Algorithm (MEA) and System Run Time Checker (SRTC) : an Authentication module. The MEA works on user supplied 128 bit key whereas SRTC keeps monitoring of all unauthorized access on SI.

2. MIPS Encryption Algorithm (MEA)

The MIPS Encryption Algorithm (MEA) works on symmetric key system and is a modification of IDEA [13] for stronger encryption. It encrypts SI in two passes. In the first pass it encrypts an input of 64 bit plain text (PT) in 64 bit cipher text (CT[2]) using block cipher method with modified key schedule to eliminate weak keys of IDEA. The second pass converts CT in end cipher text (ECT) using transposition cipher method. The ECT then is used for storage of encrypted SI on disk. The increased length of ECT can be seen as a trade-off between the high security provided by this algorithm and the nominal cost of disk storage media in present state-of-the-art development. The various steps of encryption/decryption of plain text in end cipher text are shown as follows:

- Generation of encryption keys to encrypt PT in CT,
- Encryption of PT in CT,
- Encryption of CT in ECT,
- Decryption of ECT to CT,
- Generation of decryption keys to decrypt ECT to CT,
- Decryption of CT to PT

2.1 Generation of Encryption Keys to Encrypt PT in CT: The MIPS Encryption Algorithm is designed to encrypt SI in two passes. In the first pass it encrypts a 64 bit plain text (PT) in 64 bit cipher text (CT). It requires a total of 52 encryption keys with 16 bits each as in IDEA [13]. These 52 encryption keys are generated from user inputted 128 bit key by dividing it into 8 encryption keys with 16 bits each. The 96 bits out of 128 bits i.e. 6 encryption keys are used in round1 of pass1. The details of this round may be found in section 2.2. The remaining 32 bits are the first two encryption keys for round 2. The 64 bits for four remaining encryption keys of round2 are generated from logical rotation and Exclusive-OR operation on encryption keys obtained from user supplied 128 bit key. The third encryption key of round2 is generated from an Exclusive-OR operation of 7 bits logically left rotated first encryption key with logically 8 bits right rotated second encryption key of round1. In general, the i^{th} encryption key ($9 < i < 52$) is generated from an Exclusive-OR operation of 7 bits logically left rotated $(i – 8)^{th}$ encryption key with logically 8 bits right rotated $(i – 7)^{th}$ encryption key.

2.2 Encryption of Plain Text in Cipher Text : Given a 64 bit plain text MEA converts it in a 64 bit cipher text as IDEA[3] with modified key schedule. It uses one logical and two algebraic operations for encryption as follows :

- Exclusive OR i.e. $x \otimes y = z$, $x \otimes z = y$, $y \otimes z = x$
- Addition Modulo 2^{16} (ignoring any overflow) i.e. Addition Modulo 2^{16} of x and y is (x+y) & 65535 (& stands for masking) ;
- Multiplication Modulo $2^{16}+1$ (ignoring any overflow) : We denote this operation as mul and show its result on two numbers x and y. This function is explained below :

```
unsigned mul(x,y)
unsigned x, y ;
{
long int p ;
long unsigned q ;
if (x == 0) { p = 65537 – y} else if (y = = 0) { p = 65537 – x }
else { q = x * y ; p = (q & 65535) – (q >> 16) ; if (p <= 0) p = p + 65537 ; }
return (unsigned) ( p & 65535) } ;
```

The MEA divides 64 bit plain text data block in four sub-blocks as (pt1, pt2, pt3, pt4). It performs the operations as described above on these sub-blocks for eight rounds. After each round it produces four intermediate output sub-blocks as ct11, ct12, ct13, ct14. The sequence of operations in each round is as follows :

(Notations :: $\otimes$: Exclusive OR, $\oplus$: multiplication modulo $2^{16} + 1$ and & : masking)

ct1 = pt1⊕k1 ; ct2 = (pt2 + k2) & 65535 ; ct3 =)pt3 + k3) & 65535 ; ct4 = pt4 ⊕ k4 ;

ct5 = ct1 ⊗ ct3 ; ct6 = ct2 ⊗ ct4 ; ct7 = ct5 ⊕ k5 ; ct8 = (ct8 + ct7) & 65535 ; ct9 = ct8 ⊕ k6;

[2] CT is used for transmission over Computer Networks.

[3] We have changed the notations of IDEA as per our convenience.

$ct10 = (ct7 + ct9)\ \&\ 65535$; $ct11 = ct1 \otimes ct9$; $ct12 = ct3 \otimes ct9$; $ct13 = ct2 \otimes ct10$;
$ct14 = ct4 \otimes ct10$;

Here, the intermediate output after round1 is the four sub-blocks ct11, ct12, ct13 and ct14. The input data block for round2 is produced by swapping two inner sub-blocks i.e. ct12 and ct13. Thus the input data block for round2 is (pt1, pt2, pt3, pt4) such that :
$pt1 = ct11$; $pt2 = ct13$; $pt3 = ct12$; $pt4 = ct14$;

This input (pt1, pt2, pt3, pt4) is encrypted by using the encryption keys of round2 with a similar set of operations as performed above in round1. This process of encryption should be repeated for 8 rounds. The final output after round8 will have following operations :

$ct1 = pt1 \oplus k1$; $ct2 = (pt2 + k2)\ \&\ 65535$; $ct3 = (pt3 + k3)\ \&\ 65535$; $ct4 = pt4 \oplus k4$;

Thus, MEA outputs 64 bit cipher text (ct1, ct2, ct3, ct4) from the plain text (pt1, pt2, pt3, pt4) at the end of pass 1.

2.3 Encryption of Cipher Text in End Cipher Text : The pass 2 of the MEA converts 64 bit cipher text in 128 bit end cipher text (figure 1). We apply transposition cipher method in this pass. The input for this pass is the end product of pass1 i.e. cipher text $\{(ct_1, ct_2, ct_3, ct_4)|$ where each ct_i is of 16 bits$\}$. We apply 2^8 modulo operation on each 16 bit sub-block of cipher text to split it into two components. Likewise all four sub-blocks of cipher text are split as under :

$ct1 = (ct_{11}, ct_{12})$; $ct_2 = (ct_{21}, ct_{22})$; $ct_3 = (ct_{31}, ct_{32})$; $ct_4 = (ct_{41}, ct_{42})$,

Here $ct_{i1} = ct_i \bmod 2^8$ and $ct_{i2} = ct_{i1} \otimes ((ct_i - ct_{i1}) / 2^8)$, [$\otimes$: Exclusive – OR].

Thus the input block produced for transposition in pass2 is a 128 bit block (ct_{11}, ct_{12}, ct_{21}, ct_{22}, ct_{31}, ct_{32}, ct_{41}, ct_{42}). The positions of these sub-blocks are assigned values 1,2,3,,,,,,8 sequentially for reference in a transposition key. This data block is now transposed to increase intricacy of encryption using a 128 bit transposition key supplied by the user. As an example if a user inputs a transposition key as "1,4,5,8,3,2,7,6" then the end cipher text produced is (ct_{11}, ct_{22}, ct_{31}, ct_{42}, ct_{21}, ct_{12}, ct_{41}, ct_{32}).

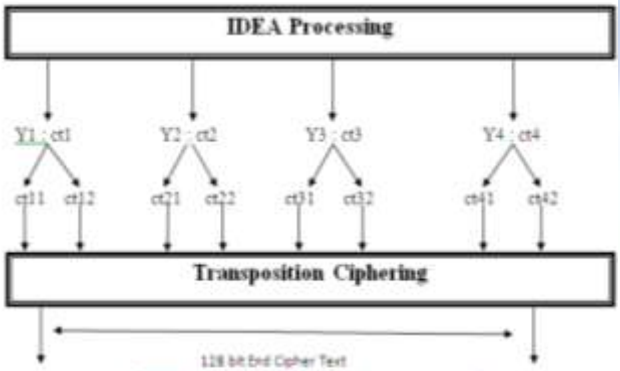

Figure 1. Application of Transposition cipher method

There are total 40320 transportation keys which can be used in pass2. Each transposition key k_i is a set of eight digits like $k_i = \{i_1, i_2, i_3, i_4, i_5, i_6, i_7, i_8 \mid i_j$ are distinct digits with $1 <= i_j <= 8$ $\}$. Indeed, each i_j represents a sub-key from user inputted transposition key. However, a repetition of sub-keys may also be allowed with the condition that each transposition key has atleast six distinct sub-keys. The remaining two repeated sub-keys should then be replaced by the sub-keys s_i obtained from $S = \{s_1, s_2 \mid s_i \in \{1,2,, ,7,8\} - \{ i_1, i_2, \ldots, i_m \mid i_m$'s are distinct sub-keys in a transportation key$\} \}$ such that if $s_1 < s_2$ then s1 replaces at the first duplicating position from the left or vice-versa.

2.4 Decryption of End Cipher Text to Cipher Text : The decryption of end cipher text to cipher text is done using the same transposition key which was used in encryption of end cipher text from cipher text due to our symmetric-key-system encryption methodology. The sub-keys of transposition are arranged in ascending order to retain positions of data sub-blocks of the end cipher text like original 128 bit data block produced before transposition in pass2. The two 16 bits data sub-blocks of thus obtained 128 bit data block are paired as (1,2), (3,4), (5,6) and (7,8) and then converted into one block of 16 bits using the reverse steps of the operations as described in section 2.3. Thus, the output produced at the end of decryption of ECT to CT is a 64 bit cipher text which indeed is same as end product of pass1.

2.5 Decryption of Cipher Text to Plain Text: The decryption logic of cipher text to plain text after decryption of ECT to CT is bit tricky and requires some doing. The application of decryption keys is position wise [Table1-2] and is based on criterion of swapping of two inner sub-blocks of input data block from first to eight rounds in pass1 (section 2.2). The generation of decryption keys is based on the inverse of functions used for encryption (section 2.2), therefore, the decryption keys are additive and multiplicative inverse of encryption keys used in pass1 and can be understood as follows :

- **Decryption key produced from an additive inverse of an encryption key :** The decryption key corresponding to i^{th} encryption produced from an additive inverse is $2^{16} - i^{th}$ encryption key.

- **Decryption key produced from a multiplicative inverse of an encryption key :** The decryption key corresponding to an encryption key produced from a multiplicative inverse is based on Euclidean God Algorithm. The details of this algorithm to find a multiplicative inverse of an encryption key, say xin, can be understood as follows. It returns k as multiplicative inverse.

```
unsigned inv (xin)
usigned xin
if (xin == 0) b2 = 0
else {n1 = 65537 ; n2 = xin ; b2 = 1 ; b1 = 0
do {
r = 65537 mod xin ; q = (n1 – r) / n2 ;
if (r == 0) { if (b2 < 0) b2 = 65537 + b2 }
else n1 = n2 ; n2 = r ; t = b2 ; b2 = b1 – q * b2 ; b1 = t ;
} while (r! = 0) ;
K = (unsigned) b2 ;
return k ;
}
```

The decryption of 64 bit cipher text (CT) to 64 bit plain text (PT) follows the same logic as of encryption of CT from PT using decryption keys in place of encryption keys. The application of decryption keys (position wise) can be understood from Table 1-2.

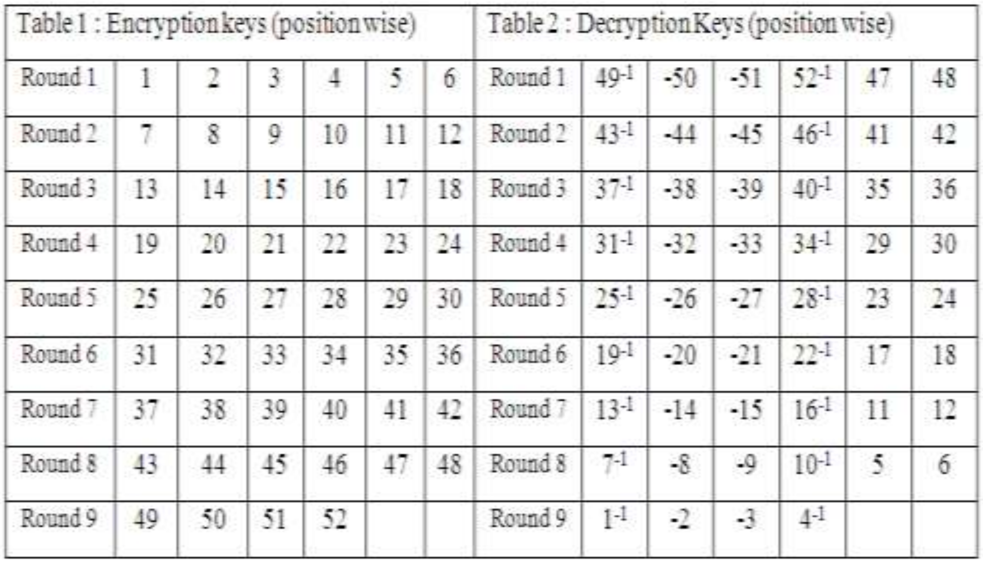

Table 1 : Encryption keys (position wise)							Table 2 : Decryption Keys (position wise)						
Round 1	1	2	3	4	5	6	Round 1	49^{-1}	-50	-51	52^{-1}	47	48
Round 2	7	8	9	10	11	12	Round 2	43^{-1}	-44	-45	46^{-1}	41	42
Round 3	13	14	15	16	17	18	Round 3	37^{-1}	-38	-39	40^{-1}	35	36
Round 4	19	20	21	22	23	24	Round 4	31^{-1}	-32	-33	34^{-1}	29	30
Round 5	25	26	27	28	29	30	Round 5	25^{-1}	-26	-27	28^{-1}	23	24
Round 6	31	32	33	34	35	36	Round 6	19^{-1}	-20	-21	22^{-1}	17	18
Round 7	37	38	39	40	41	42	Round 7	13^{-1}	-14	-15	16^{-1}	11	12
Round 8	43	44	45	46	47	48	Round 8	7^{-1}	-8	-9	10^{-1}	5	6
Round 9	49	50	51	52			Round 9	1^{-1}	-2	-3	4^{-1}		

3. MEA Security

MIPS Encryption Algorithm (MEA) is devised in such a manner in such a manner that it will run from 16 bit machine onwards. The testing of MEA has been performed on the range of machines including latest Intel Core Processors and 64 bit RISC machines and has been successful. The former data encryption algorithm DES requires only 2^{56} encryptions for break. A million chips capable of testing a million keys per second can break DES in 20 hours. If one can design a chip capable of testing a billion keys per second and uses a billion of them to solve the problem then it will take 10^{13} years to break IDEA. Our computations reflect that with the same design specifications of a set of chips as to break encryption of IDEA, a time of $2x10^{13}$ years will be required to break MEA using a method of brute force but the question is that can one really design a machine with an array of one billion chips with such required capabilities. But still MEA is very new and is open to challenges.

4. Conclusion and Discussion

We tested MEA using a number of common approaches of an attacker and observed that MEA provides relatively higher degree of security. This algorithm seems to be better secured due to the introduction of transportation ciphering in pass2 (section 2.3). It indeed intricates the work of a cryptanalyst by adding a search for a transportation key which was used to encrypt CT to ECT since we split 16 bits of cipher text into two parts where one part is a modulo 2^8 of cipher text sub-block and other part is Exclusive-OR of the remainder with quotient of CT with same operation. We further transpose thus produced data sub-blocks by transposition. Hence while decrypting, an analyst may never be sure of finding a correct code even after trying all combinations of transposition because some false combinations may also be produced due to the fact that we have split each sub-block of cipher text into two parts which can be represented in ASCII code.

In fact, splitting a CT in two parts is an attempt to discourage the application of method of brute force which normally assures solution. Likewise the application of differential cryptanalysis [14] may not be an easy task for attacker. Even the methods like known text may not be a successful attack on MEA and so is the case with the guess method.

Although the security of an Information System may demand some ongoing changes and reforms in the present algorithm to meet upcoming challenges, but at present the algorithm MEA in MIPS looks secured for break. One may not deny the fact that security locks are broken time to time motivating researchers for further work in this field. Who knows what break may come out tomorrow for MEA.

References

[1]. Payal Maggo and Rajender Singh Chhillar., “Lightweight Image Encryption Scheme for Multimedia Security.”, International Journal of Computer Applications 71(13):43-48, June 2013.

[2]. R. Gupta, A. Aggarwal & S. K. Pal, "Design and Analysis of New Shuffle Encryption Schemes for Multimedia", Defence Science Journal, Vol. 62, No. 3, May 2012, pp. 159-166.

[3]. Pankaj Rakheja, Amanpreet kaur, “A Unique Cryptographic Mechanism for Encoding Data Using DNA Structure”, in International conference on Network Communication and Computers (ICNCC 2011) organized and sponsored by IACSIT, The Institute of Electrical and Electronics Engineers (IEEE), Singapore Institute of Electronics and other organizations.

[4]. Modugu, R., Yong-Bin Kim, Minsu Choi, “Design and performance measurement of efficient IDEA (International Data Encryption Algorithm) crypto-hardware using novel modular arithmetic components “,Instrumentation and Measurement Technology Conference (I2MTC), 2010 IEEE

[5]. Nabarun Bagchi, “Secure BMP Image Steganography Using Dual Security Model (I.D.E.A, image intensity and Bit Randomization)and Max-Bit Algorithm” International Journal of Computer Applications 1(21):18–22, February 2010.

[6]. Chong Fu, Zhen-chuan Zhang , Ying-yu Cao. “ An improved image encryption algorithm based on chaotic maps” Third International Conference on Natural Computation. 2007, Vol. 13, pp.189-193.

[7]. Michalski, A., Buell, D., Gaj, K., “High-throughput reconfigurable computing: design and implementation of an IDEA encryption cryptosystem on the SRC-6E reconfigurable computer”Field Programmable Logic and Applications, 2005. Page(s): 681 - 686

[8]. J. Chen, “A DNA-based, biomolecular cryptography design,” in IEEE International Symposium on Circuits and Systems (ISCAS), 2003, pp. 822–825

[9]. Sanchez-Avila, C. Sanchez-Reillol, “The Rijndael block cipher (AES proposal) : a comparison with DES” 2001 IEEE 35th International Carnahan Conference on Security Technology

[10]. M.P. Leong, O.Y.H. Cheung, K.H. Tsoi and P.H.W. Leong, “A Bit-Serial Implementation of the International Data Encryption Algorithm IDEA,” 2000 IEEE Symposium on Field-Programmable Custom Computing Machines, IEEE (2000), pp. 122-131.

[11]. M.D.Abrams, H.J.Podell (eds.), “Computer and Network Security”, IEEE Computer Society”, 1987, Washington, DC.

[12]. Biham Ali, “New Type of Cryptanalytic Attacks Using Related Keys”, Jcryptol, Vol 7, No.4, Fall 1994, pp 229-242.

[13]. Rodeny H.Cooper, Willaim Hyslop and Wayne Patterson,” An Application of the Chinese Remainder Theorem to Multiple-key Encryption in Data Base Systems”, Proc IFIP/Sec84, Canada, Spet 84, pp 553-556.

[14]. “Data Encryption Standards”, Federal Information Processing Standards Publication 46, National Bureau of Standards, 1977, Washington.

[15]. D.E.Denning, Cryptography & Data Security, Addison Wesley Publication, 1982, Reading, M.A..

[16]. W.Deffie, M.E. Hellman, “Exhaustive Cryptanalysis of NBS Data Encryption Standards”, Computer, Vol.10, June 1977, pp 74-84.

[17]. E.B.Fernandez, R.C.Summer and C.Wood, Data Security & Integrity, Reprinted 1983, Addison Wesley Publishing Co..

[18]. M.E.Hellman, “DES Will be Totally Insecure Within Ten Years”, IEEE of Software Spectrum, Vol.16 July 1978, pp 40-41.

[19]. C.J.Holloway, “Controlling the Use of Cryptographic Keys”, Computer Security (UK), Vol 14, No.7, 1995, pp 587-598.

[20]. Gerd E.Keiser, “Local Area Networks”, 1989, MGH International Edition.

[21]. W.E.Madryga, A High Performance Encryption Algorithm”, Proc IFIP/Sec84, Canada, Spet84, pp 557-570.

[22]. J.Reid, “Open Systems Security : Traps And Pitfalls”, Computer Security (UK), Vol 14, No.6, 1995, pp 496-517.

[23]. Bruce Schneier, “A IDEA Encryption Algorithm”, Dr Dobbs Journal, Dec1993, pp 50-56.

[24]. Bruce Schneier, “Differential and Linear Cryptanalysis”, Dr.Dobbs Journal (USA), Vol 21, No.1, Jan 1996, pp 42-48.

ER PUBLICATIONS