International Journal of Computer Science and Engineering (IJCSE)
ISSN 2278-9960
Vol. 2, Issue 3, July 2013, 89-94
© IASET

International Academy of Science, Engineering and Technology
Connecting Researchers; Nurturing Innovations

A STUDY OF MARS, RC6 AND SERPENT

NEETA WADHWA, SYED ZEESHAN HUSSAIN & S. A. M RIZVI

Department of Computer Science, Jamia Millia Islamia, New Delhi, India

ABSTRACT

After DES [Data Encryption Standard] was cracked, the new symmetric encryption standard hunt was started in 1999 and finished in 2001. The process was comprised of two rounds.This paper presents the study of 3 out of 5 finalists of second round of AES [Advanced Encryption Standard] process: MARS, RC6 and SERPENT. It analyzes the structure and working of these algorithms. It also analyzes the three algorithms on the basis of their encryption and decryption time.

KEYWORDS: AES, DES, MARS, RC6, SERPENT, Symmetric Cryptography

INTRODUCTION

When digital communication systems developed, everyone had been using cryptography in his own ways secretly for their personal use. Due to the two major inventions in the world of digital communication, this art became science. First, The publication of the draft DES in the U.S. Federal Register on 17 March 1975. It started the development of Symmetric Cryptography. Second, The publication of the paper 'New Directions in Cryptography' by Whitfield Diffie and Martin Hellman in 1976. This key exchange protocol gave birth to the Asymmetric Cryptography. This work focuses only on Symmetric cryptography.

DES was criticized because of its small key length and cracked by bruteforce attack in late 90's. Then AES process started. First round screened out 15 algorithms and the second round shortlisted 5 algorithms. In final round, the winner, RIJNDAEL, got 86 votes at the third AES conference while SERPENT got 59 votes, TWOFISH got 31 votes, RC6 got 23 votes and MARS got 13 votes. So NIST [National Institute of Standards and Technology] selected Rijndael as the AES. The present study analyzes the following three algorithms.

STRUCTURE

Mars

MARS [1] designed by IBM, it was one of the finalists of AES competition but could not win. Like all AES candidates, it uses 128-bit blocks and supports key sizes of 128, 192 or 256 bits. It uses a variant of the Feistel structure which designers call a "type 3 Feistel network"; it is word oriented cipher, the 128-bit block is treated as four 32-bit sub-blocks or words; each round uses one sub-block as input and modifies all of the other three sub-blocks. It uses data-dependent rotations like RC6. One 9*32 S-box is used; for some operations it is treated as two 8*32 S-boxes. MARS supports user key lengths from 128 bits to 448 bits.

The key expansion procedure expands the user-supplied key array k0, …, kn-1, into a 40-word internal key array K0, …, K39. The range of n is from 4 to 4 32 bit words. The key expansion procedure guarantees that the key words which are used for multiplication do not have any obvious weaknesses. This procedure keeps these words "random", in the sense that no single word has probability much larger than in the uniform distribution. Due to the structure of the key expansion procedure, the performance of MARS is essentially independent of the key-length used.

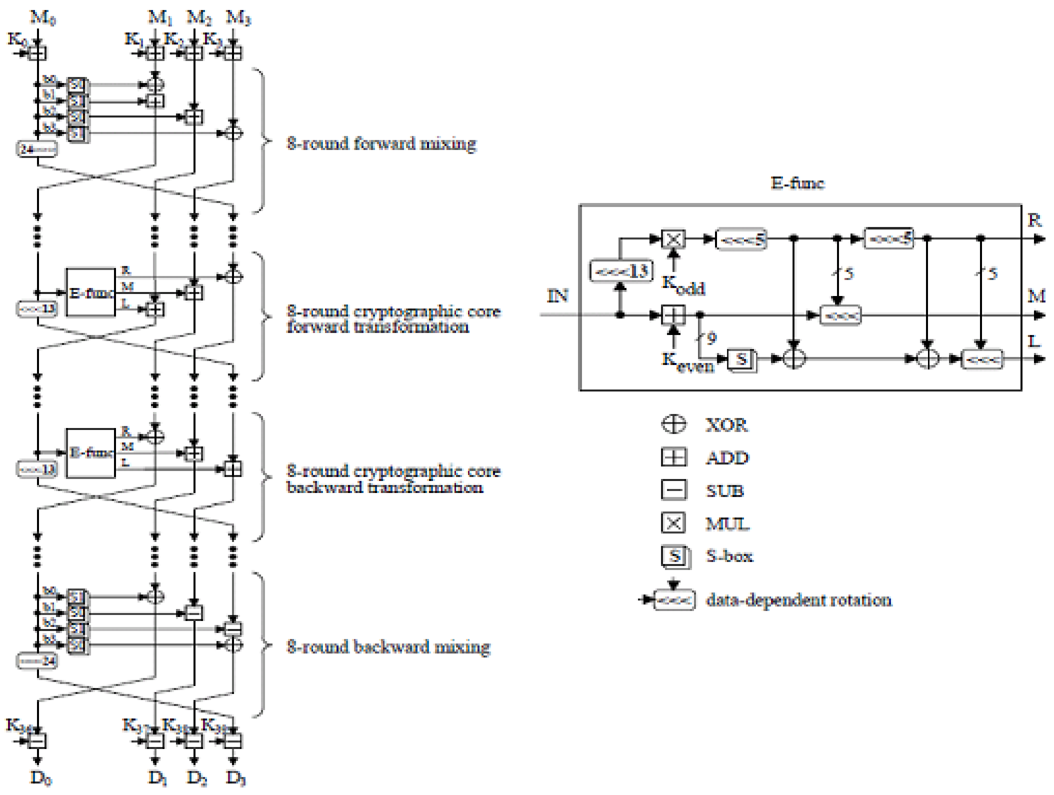
8-round forward mixing
8-round cryptographic core forward transformation
8-round cryptographic core backward transformation
8-round backward mixing
E-func
IN
R
M
L
XOR
ADD
SUB
MUL
S-box
data-dependent rotation

Figure 1: Structure of MARS [1]

In [2] it was shown that the earlier key-schedule of MARS had equivalent keys. That is, pairs of keys which produce the same set of round keys. This was possible also because that key-schedule allowed for keys up to 1248 bits. A new key-schedule was proposed in [3], which is the current key-schedule of MARS. The equivalent keys do not exist for MARS with this new key-schedule.In [4] it was shown that the MARS S-boxes do not satisfy exactly the criteria claimed by the designers. Also, in [5] it was shown that there exist linear relations in the 9 to 32 bit S-box. But these findings have not been utilized in any improvement of cryptanalysis on MARS. Thus they do not affect the security of MARS. In [6], it was claimed that the bound on the best biases in a linear approximation on the core rounds was "only" 2^{-49} where the designers' bound was 2^{-69}. These numbers are only bounds, and do not represent the biases of linear approximations actually determined. It is therefore likely that any linear approximation will have a lower bias. Since a linear attack needs approximately b^{-2} texts to succeed where b is the bias, both numbers are low enough to conclude that MARS is resistant to a linear attack.

RC6

RC6, designed by Dr. Ronald C. Rivest, another strong finalist of second round for consideration as the new Advanced Encryption Standard (AES). It is based on the RC5 block cipher [7]. In May 1997, the U.S. patent office granted the RC5 patent to RSA Data Security (now RSA Security). Since RC5 was proposed, there have been numerous studies of RC5's security [8,11]. Each study has provided a greater understanding of how RC5's structure and components contribute to its security. In a survey article [12], a summary of known cryptanalytic results is given. And RC6 is an evolutionary improvement of RC5, designed to meet the requirements of the Advanced Encryption Standard (AES). In RC6, the number of rounds, the size of the key and the size of the block, are all flexible. RC6 is based on Feistel rounds; but not Feistel rounds operating between the two halves of the block. Instead, the Feistel rounds operate between pairs of quarters of the block, and they are interlocked by the exchange of some data. The key schedule of RC6-w/r/b is practically identical to the key schedule of RC5-w/r/b. Indeed, the only difference is that for RC6-w/r/b, more words are derived from the user-supplied key for use during encryption and decryption. The user supplies a key of length *k* bytes which is then expanded to a set of subkeys. The key schedule of RC6 is described in [13].

RC6 uses 44 subkeys, numbered S0 to S43, each one 32 bits long. The block to be enciphered is divided into four 32-bit integers, A, B, C, and D. The first four bytes enciphered form A, and the convention is little-endian; the first byte enciphered becomes the least significant byte of A. Here f(a) = a x (2a + 1).

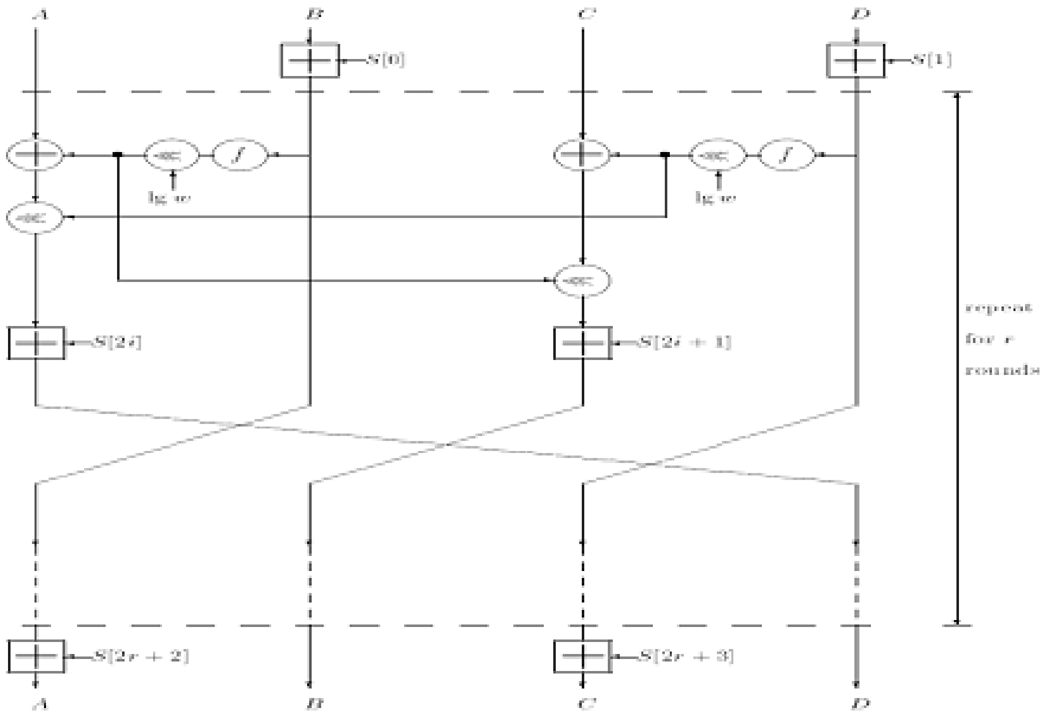

Figure 2: Structure of RC6 [13]

RC6 is a fully parameterized family of encryption algorithms. A version of RC6 is specified as RC6-w/r/b where the word size is w bits, encryption consists of a nonnegative number of rounds r, and b denotes the length of the encryption key in bytes. For all variants, RC6-w/r/b operates on units of four w-bit words using the following six basic operations.

a + b integer addition modulo 2^w , a - b integer subtraction modulo 2^w, a ⊕□b bitwise exclusive-or of w-bit words, a x b integer multiplication modulo 2^w, a<<<b rotate the w-bit word a to the left by the amount given by the least significant lgw bits of b, a>>>b rotate the w-bit word a to the right by the amount given by the least significant lgw bits of b. The base-two logarithm of w will be denoted by lgw.

The work of [14] proposed an area optimized hardware architecture to the RC5 core into a Field Programmable Gate Array (FPGA) device with fewer resources than the conventional one. But the encryption throughput was found *less* than the conventional architecture, and it did not propose any modifications to the conventional system architecture.

Serpent

SERPENT [15] is based on the class of substitution-permutation networks (SPN) having 32 rounds. It operates on 128 bit blocks of data and a 256 bit external key (The key is variable but internally it is required to be 256-bits therefore they pad any supplied key that is less than 256-bits with a 1 followed by the required number of 0s to make the key 256bits). Each round requires its special 128-bit round key; since the last round needs two keys, total of 33 different round keys are required and these are generated from the external key in a separate key schedule. The transformation flow is divided into 32 uniform rounds repeated over the data block with each round consisting of (nearly identical) sequence of elementary operations. It does its computations in little-endian. After key mixing the result is then passed through the s-boxes. There are 32 rounds and 8 sboxes, where a single s-box is used for each round so each s-box is used four times.

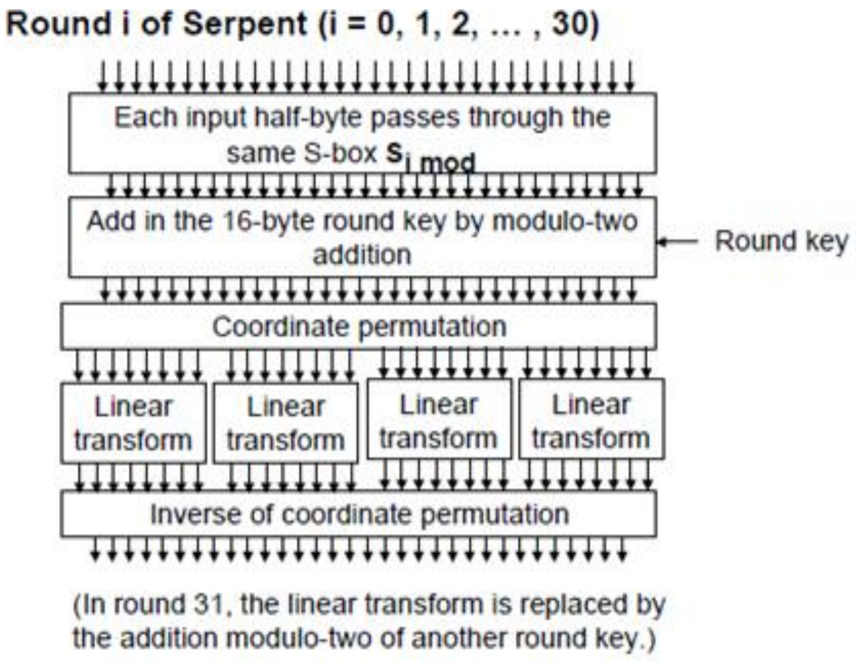
Round i of Serpent (i = 0, 1, 2, ... , 30)
Each input half-byte passes through the same S-box $S_{i \bmod}$
Add in the 16-byte round key by modulo-two addition
Round key
Coordinate permutation
Linear transform
Linear transform
Linear transform
Linear transform
Inverse of coordinate permutation
(In round 31, the linear transform is replaced by the addition modulo-two of another round key.)

Figure 3: Structure of SERPENT[15]

Serpent encrypts a 128-bit plaintext *P* to a 128-bit ciphertext *C* in 32 rounds under the control of 33 128-bit subkeys. The cipher consists of the following basic steps:

An initial permutation *IP*, 32 rounds, each consisting of a key mixing operation, a pass through S-boxes, and (in all but the last round) a linear transformation. In the last round, this linear transformation is replaced by an additional key mixing operation,a final permutation *FP*.

The initial and final permutations do not have any cryptographic significance. They are used to simplify an optimized implementation of the cipher and to improve its computational efficiency. Serpent is much faster than DES. Its design supports a very efficient bitslice implementation, and the fastest version at the time of the competition ran at over 45 Mbit/sec on a 200MHz Pentium (compared with about 15 Mbit/sec for DES). Because of the large number of rounds it is slow as compared to other finalists but it had been claimed most secure [16]. Serpent is the best of the AES finalists in hardware - even with the full 32 rounds. An independent team produced implementations for the Xilinx XCV1000 FPGA of RC6, Rijndael, Serpent and Twofish. Serpent was the only finalist for which a fully pipelined implementation could be fitted into a single chip. Serpent was also by far the fastest, achieving a throughput of 5.04 Gbit/sec, versus 2.40 Gbit/sec for RC6, 1.94 Gbit/sec for Rijndael and 1.71 Gbit/sec for Twofish. An NSA study of ASIC costs predicts 8.03 Gbit/sec for Serpent versus 5.163 for Rijndael, 2.171 for RC6 [17]. Even on Pentium, using this benchmark, Serpent is the third fastest algorithm when one combines the published cycle count figures from Gladman [18] and Osvik [19], and fourth fastest combining Worley et al [20] and Osvik [19].

SIMULATION RESULTS

The algorithms are implemented according to their standard specifications in .Net environment using C#, on windows XP OS. In the experiment the pdf and text files of different size ranges between 2MB to 20MB are encrypted and their encryption time is calculated.

For the experiment, Intel Pentium® Dual Core 2.50GHz CPU with 4GB RAM (DDR2 DRAM frequency 399.0MHz) is used.

Table 1: Encryption Time (in Milliseconds) of MARS, RC6 and SERPENT

File Size (Mb)	Mars	Rc6	Serpent
2.77	98	67	74
3.21	111	71	79
5.76	122	73	86
6.43	129	76	89
8.83	136	79	94
9.45	142	80	97
11.65	156	83	106
14.34	162	94	118
18.77	188	102	121

Table 2: Decryption Time (in Milliseconds) of MARS, RC6 and SERPENT

File Size (Mb)	Mars	Rc6	Serpent
2.77	101	66	80
3.21	109	72	84
5.76	126	71	93
6.43	134	74	99
8.83	145	77	106
9.45	149	81	110
11.65	159	84	115
14.34	169	91	128
18.77	200	108	149

CONCLUSIONS

All the finalists had non-Feistel structure with their own strengths. RC6 had simple structure whereas MARS and SERPENT had complex designs. RC6 takes smallest amount of time for both encryption and decryption process. So RC6 is fastest followed by SERPENT and MARS in terms of encryption/decryption speed.

REFERENCES

1. C. Burwick, D. Coppersmith, E. D'Avignon, R. Gennaro, S. Halevi, C. Jutla, S. M. Matyas, L. O'Connor, M. Peyravian, D. Safford and N. Zunic, *MARS – a candidate cipher for AES,* NIST AES Proposal, Jun 98. http://csrc.nist.gov/encryption/aes/round2/AESAlgs /MARS/mars.pdf, Aug. 1999.
2. M. Saarinen, *A Note Regarding the Hash Function Use of MARS and RC6*, available online from http://www.jyu.fi/ mjos/, 1999
3. N. Zunic, *Suggested 'tweaks' for the MARS cipher*, a proposed modification, submitted on May 15, 1999.
4. L. Burnett, G. Carter, E. Dawson, W. Millan, *Efficient Methods for Generating MARS-like S-boxes*, In proceedings of FSE'2000 to be published by Springer Verlag.
5. L. Knudsen and H. Raddum, *Linear approximations to the MARS S-box*, AES Round 2 public comment, April 7, 2000.
6. M. Robshaw and Y. Yin, *Potential Flaws in the Conjectured Resistance of MARS to Linear Cryptanalysis*, AES Round 2 public comment, April 27, 2000.

7. R.L. Rivest, *The RC5 encryption algorithm*, In B. Preneel, editor, Fast Software Encryption, volume 1008 of Lecture Notes in Computer Science, pages 86-96, 1995,Springer Verlag.
8. B.S. Kaliski Jr. and Y.L. Yin, *On differential and linear cryptanalysis of the RC5 encryption algorithm*, Advances in Cryptology - Crypto '95, Springer-Verlag (1995), 171-183.
9. L.R. Knudsen and W. Meier, *Improved differential attacks on RC5*, *Advances in Cryptology - Crypto '96*, Springer-Verlag (1996), 216-228.
10. A. Biryukov and E. Kushilevitz, *Improved cryptanalysis of RC5*, Advances in Cryptology - Eurocrypt '98, Springer Verlag (1998).
11. A. A. Selcuk, *New results in linear cryptanalysis of RC5*, Proceedings of 5th International Workshop on Fast Software Encryption, Springer Verlag (1998), 1-16.
12. Y.L. Yin, *The RC5 encryption algorithm: two years on, CryptoBytes* (3) 2 (Winter 1997).
13. R.L. Rivest, M.J.B. Robshaw R. Sidney and Y.L. Yin, *The RC6 Block Cipher*,v1.1, August 20, 1998. Available at www.rsa.com/rsalabs/aes/.
14. N. Sklavos, C. Machas, O. Koufopavlou, *Area Optimized Architecture and VLSI Implementation of RC5 Encryption Algorithm*, In Proceedings of the *IEEE ICECS '2003*, vol. 1, Dec. 2003.
15. R. Anderson, E. Biham, L. Knudsen, *Serpent: A Proposal for the Advanced Encryption Standard*, The First Advanced Encryption Standard (AES) Candidate Conference, Ventura, California, August 20–22, 1998 (http://www.cl.cam.ac.uk/~rja14/serpent.html), 1998.
16. R.J.Anderson, E. Biham, L.R. Knudsen, Serpent and Smartcards, in *Cardis 98*, Springer Verlag (2000) pp. 257-264; also available at http://www.cl.cam.ac.uk/~rja14/serpent.html
17. B.Weeks, M.Bean, T.Rozylowicz, C.Ficke, *Hardware Performance Simulations of Round 2 Advanced Encryption Standard Algorithms*, In the proceedings of the 3rd AES Candidate Conference.
18. B.Gladman, *Implementation Experience with AES Candidate Algorithms*, in Proceedings of the 2nd AES Candidate Conference (NIST, 1999) pp 7-14
19. DA Osvik, *Speeding Up Serpent*, In the proceedings of the 3rd AES Candidate Conference
20. J Worley, B Worley, T Christian, C Worley, AES Finalists on PA-RISC and IA64: Implementations and Performance", to appear in the proceedings of the 3rd AES Candidate Conference