Journal of Computer Science 5 (8): 573-578, 2009
ISSN 1549-3636

Message Based Random Variable Length Key Encryption Algorithm

Hamid Mirvaziri, Kasmiran Jumari Mahamod Ismail and Zurina Mohd Hanapi
Department of Electrical, University Kebangsaan Malaysia, Bangi, 43600 Malaysia

Abstract: Problem statement: A block ciphers provides confidentiality in cryptography but cryptanalysis of the classical block ciphers demonstrated some old weaknesses grabbing a partial key in any stage of encryption procedure leads to reconstructing the whole key. Exhaustive key search shows that key generation should be indeterminist and random for each round. Matching cipher-text attack shows that larger size of block is more secure. In order to overcome analysis mentioned above a new algorithm is designed that is based on random numbers and also can defeat time and memory constraints. **Approach:** Dynamic and message dependent key generator was created by producing a random number and it was selected as the size of first chunk. Residual value of second chunk divided by first chunk concatenating with first chunk forms the first cipher as an input for SP-boxes. These processes repeated until whole mesaage get involved into the last cipher. Encrypted messages are not equal under different run. Value of random number should be greater than 35 bits and plaintext must be at least 7 bits. A padding algorithm was used for small size messages or big random numbers. **Results:** Attack on the key generation process was prevented because of random key generation and its dependency to input message. Encryption and decryption times measured between 5 and 27 m sec in 2 GHz Pentium and java platform so time variant and fast enough key generation had been kept collision and timing attacks away due to small seized storage. Long and variable key length made key exhaustive search and differential attack impossible. None fixed size key caused avoidance of replaying and other attacks that can happen on fixed sized key algorithms. **Conclusion:** Random process employed in this block cipher increased confidentiality of the message and dynamic length substitution in proposed algorithm may lead to maximum cryptographic confusion and consequently makes it difficult for cryptanalysis.

Key words: Security, encryption, block cipher, variable key length

INTRODUCTION

In 1997 NIST issued the second public request for an encryption standard and after four years of competition, Rijndael was chosen by NIST in October 2000 and officially became AES in 2001 with US FIPS 197. Although this cipher is now widely deployed and is expected to be the world's predominant block cipher over the next 25 years but this could not stop cryptographers to analysis it.

During AES selection, only branch statements, arithmetic and data-dependent shift were considered vulnerable. In the AES structures each round is weak and identical. The key schedule is also simple with fixed length and every operation is invertible furthermore the key length of AES was also fixed and small for acceptable commercial security. Since the selected algorithms were implemented using only table lookup, X-or and shift operations, there are variety of successful analysis for those algorithms.

Bertoni introduced power attack[1] and after a while this attack implemented and requires about 50 encryption traces and searching 2^{36} possible keys in AES[2]. Kocher *et al*.[3] have described two types of attacks, a Simple Power Analysis (SPA) attack and a Differential Power Analysis (DPA) attack. An SPA attack is described as an attack where a single power consumption signal was used to break a cryptosystem. The information in the power signal is usually very small; thus steps such as executing random dummy code or avoiding memory accesses by processing data in registers can often help to protect against SPA attacks. DPA attack uses statistical analysis to extract information from a power signal. Information that might be undetectable by using SPA can often be extracted using DPA. In the original DPA attack described by Kocher *et al*.[3] the means of the probability distributions are analyzed. The mechanism that enables a DPA attack is the probability distribution function of a point in the power consumption signal that

Corresponding Author: Hamid Mirvaziri, Department of Electrical, University Kebangsaan Malaysia University, Bangi, 43600 Malaysia

can be dependent on the input data, output data and secret key. Most operations performed by classical algorithms have this property, thus most operations are potentially vulnerable to a DPA attack[4].

Some countermeasures against DPA attack is suggested by[5]. They inserted dummy codes, randomized power consumption and balanced data. Goubin *et al*.[6] proposed a new method to protect the DES algorithm from DPA attack. They concluded that nonlinear transformation and random timing shift can prevent general power attacks. Chari *et al*.[7] suggested that not all intermediate data in every round of algorithms need to be masked. For example, they suggest that only the first and last four rounds of DES need to use their scheme to be more secure. NIST also mentioned Rijndael algorithm is not vulnerable to timing attack but Daniel Bernstein has announced successful timing attack against AES by exploiting timing characteristics of table lookups. The statistical attack can even be extended to exploit timing variation of individual bits of the key instead of whole bytes[8]. Neve *et al*.[9] described some condition in which the attack might work and also the limitations of the Bernstein attack. The details of this analysis can be found in[10].

More efficient timing attacks against AES directly using cache effects presented by Bonneau and Mironov. They consider a model for attacking AES by using the timing effects of cache-collisions to gather noisy information about the likelihood of relations between key bytes[11]. Table lookup randomization and eliminating special last round table in AES makes timing attack harder. It also ensures a secured cipher with no detectable structure and having different keys result in independent random permutations.

Attackers use cache information to determine bytes of the one round key because knowing all key bytes of the one round key possibly lead to revert the key schedule and computing the cipher key if the key generation process is a function[12]. In some encryption algorithm, intruder tries to observe memory access patterns to learn about the table lookups and table lookups will incur cache misses so it can be avoided if an algorithm do not uses table lookup or mix the order of the table elements several times during each encryption. The memory accesses of software cryptosystems, especially S-box based ciphers like DES and AES that have key-dependent table lookups and the knowledge of the processed message, e.g., in a known-text attack, make it relatively easy to break these ciphers[13].

The adversary will be force to either mount a refined and more complex Cache Based Attack (CBA) or combine the cache attack with some other method to determine the key bytes if CBA exposes as few information as possible that will not be consequence to the complete key. Something that is happened in[14], where a CBA on the first round only reveals 4 bits of each key byte. Hence Osvik *et al*.[14] combine cache attacks on the first and second round of AES that is known as cache-observation attack. The permutations also require to be chosen with lookup efficiency and the choice of permutation need to be sufficiently strong. Bloomer demonstrated that present random permutation techniques do not increase the complexity of CBAs as much as one might expects[12]. Those random permutations do not even prevent the leakage of the complete secret key as proposed in[15]. He considered a modified countermeasure based on random permutations that can use for any encryption algorithm[12]. He also provided a formal notion of security for randomized masking of arbitrary cryptographic algorithms that can prevent side-channel attack if the adversary is able to access a single intermediate result. His randomized masking technique is quite general and it can be applied to arbitrary algorithms using only arithmetic operations over some finite field[16].

Side-channel attacks were not given enough attention in the AES selection process. For example Rijndael makes heavier use of lookup tables than any of the other four AES finalists, which exposes it to multiple side-channel attacks, including timing. By comparison, Serpent[17] uses only tiny 4 by 4 bit S-boxes, which is in fact implemented only by logical operations, making Serpent invulnerable to cache-based side-channel attacks. At that time this was not recognized as an advantage, but it should be clear now that table lookups should be avoided or used with extreme caution. Since the proposed attack focuses attention on the presence of cache miss, the countermeasure has to avoid cache misses related to the computation of the encryption algorithm. In this case, if no information about the secret key can be discovered the attack can not be performed[19].

Unfortunately AES randomization techniques have based their security on heuristics and experiments thus; flaws have been found which make AES randomized implementations still vulnerable to side-channel cryptanalysis[20]. All cryptanalysis that is discussed above lead us to design an algorithm named Message Based Random Variable Length Key Encryption (MRVLK) that is described in the next section and not only resist against those kinds of attacks but also have the ability to use under speed and memory constraints.

Description of MRVLK: A classical block cipher is a substitution cipher with the following structure:

$$\{0,1\}^n \rightarrow \{0,1\}^n$$

The model above has been performed in most classical ciphers with isolated key generation. However there are some restrictions for this model as follows:

Total number of keys are limited and cipher is insecure when n is small and impractical when n is large. Cipher key generations are mostly functional and consequently deterministic. There is a correlation between the keys in each stage so any intruder can find the whole key if he can find partial key in any stage. Almost all classical ciphers have corresponding fix input and output blocks length that is easy to cryptanalyze.

MRVLK has variable block length and a variable key length cipher can be started from large bits and then grow in sizes. The cipher has variable rounds, random bitwise rotations and dynamic key length with a well designed key schedule that provides resistance against linear and differential cryptanalysis. This cipher is non corresponding cipher with the following structure:

$$\{0,1\}^n \rightarrow \{0,1\}^m$$

where, n can be smaller or greater than m, i.e., it is an encryption method with no correspondence. MRVLK with a dynamic and dependent key generator tries to overcome power attack by randomization of the data and nonlinear substitution therefore no information about the secret key can be discovered and this attack can not perform. Variable key selection is completely random and message dependent but once the key created it is the only one that can decrypt the cipher text. The key-dependent S-boxes are used because they offer protection against known statistical attacks. The bit rotations prevent attacks based on the byte structure and the dynamic length of the cipher is far from analysis to be able to break.

Almost all key generations in previous ciphers are functional and deterministic but proposed key generation, encryption and decryption in MRVLK is probabilistic, time variant and storage efficient. MRVLK is performed by cipher length randomization techniques that can prevent side-channel cryptanalysis. Time canonization and randomization are also implemented due to random initial message length selection; the way of timing attack avoiding[15]. Key scheduling and method is brought in next sections.

Key schedule: There is no modulo addition and multiplication to fit the encryption for low power devices. There are two types of shift and rotating operations, fixed and data dependent. In most of AES candidates fixed rotating or shift operation has been used. In MRVLK data dependent rotating operations is used as in Mars and RC6. As a matter of fact fixed S-boxes (e.g., DES) allow attackers to study S-boxes and discover weak points but in key-dependent S-boxes, attacker doesn't know how the S-boxes act. Complexity of keyed S-box depends on the length of the key i.e., takes longer to set up for a key, since S-boxes have to be built for each key and generally key-dependent S-boxes are used because they offer adequate protection against known statistical attacks and are likely to offer protection to any unknown similar attacks. Key bits rotation is added to prevent potential attacks that relied solely on the byte structure. Key bit length can be very challengeable in symmetric key encryption algorithms specially when it is not fixed length and depends on a random number. We have designed a very thorough key schedule to prevent related-key and weak-key attacks.

In order to design a sufficient key length the algorithm start to find large bit length keys due to security condition and key length will be increased dramatically based on the input message length. This key schedule confirms that security of the key is proportional to Y!/X! where X is bit length of the key at starting point and Y is whole key bit length and it is bigger than 2^y, the security of classical algorithms when X is small and Y is large enough. Larger key length that provides more security in algorithms can be achieved with a random number bigger than 60 bits. Figure 1 is demonstrating key length changes versus different random numbers as K_1. Although there is a key length peak in the range of 30-33 bits but it is better to choose size of random number more than 60 bits as it has been discussed before. In the other side very large cipher text length is not recommended because of memory constraint. As it is shown in Fig. 2 changes in cipher text length is similar to key length but smaller cipher text can be obtained by random numbers more than 33 bits and less than 64 bits in length.

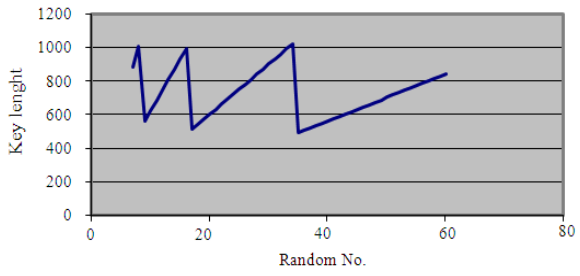

Fig. 1: Key length changes in different random numbers

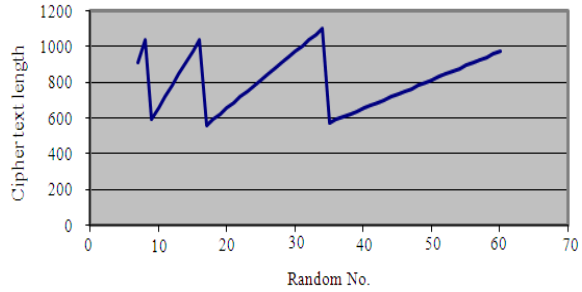

Fig. 2: Ciphertext changes in different random numbers

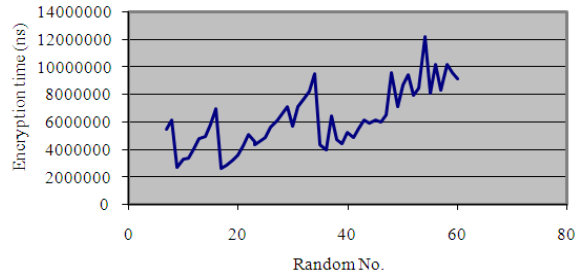

Fig. 3: Encryption time versus random number changes

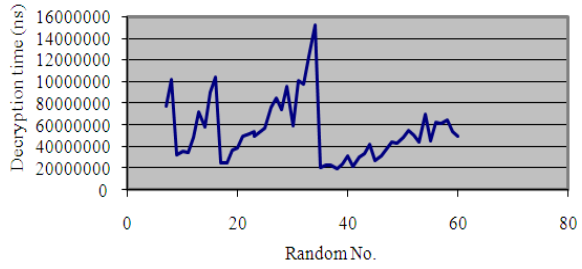

Fig. 4: Dncryption time versus random number changes

When the size of random number varies between 7 and 35 for example, there is a gradually increase in encryption and decryption time but after that total time will decreases sharply. The decline that is demonstrated in Fig. 3 and 4 describes that random number should not be less than 35 bits in size. Normally intruders need more time to find out larger numbers and since aggregation time is important for some devices like smartcards so it is recommended not to choose less than 35 bits for its size.

Most of cache attacks try to find the key during key generation phase because in classical ciphers re-keying is a deterministic function i.e., there is a correlation between the keys in each stage so re-keying is known even though this procedure chosen carefully and independent from key length[8] but in MRVLK there is no correlation between the keys in each stage and the size of the key so cache and differential key analysis does not expected.

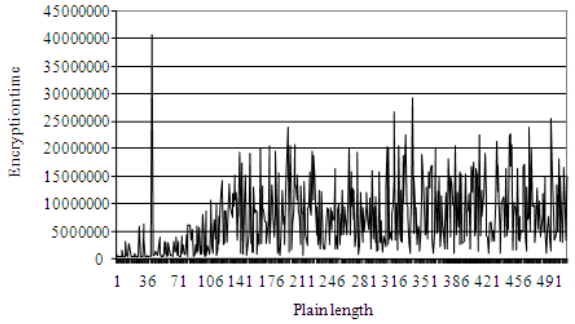

Fig. 5: Encryption time in different message length

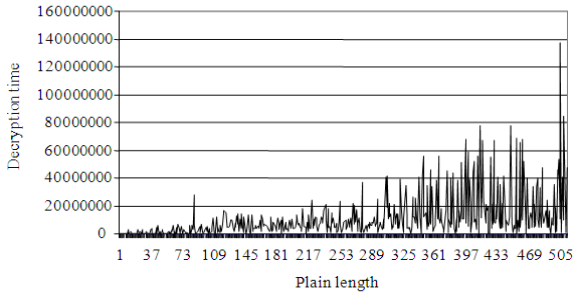

Fig. 6: Decryption time in different message length

The encryption algorithm and key schedule must be designed one after another; subtle changes in one affect the others. It is not enough to design a strong round function and then insert a strong key schedule onto it; both must work together. Cipher implementation with circumstance condition makes it very hard to mount any statistical cryptanalysis. For large messages, performance of the key schedule is minor compared to performance of the encryption and decryption processes. For smaller messages, key process can overwhelm encryption speed so in MRVLK design, we tried to find the best message chunks to balance between the time and memory storage. As it is shown in Fig. 5 the best size of plain text is between 221 and 291 bits as chunks (average 264) to make the algorithm as fast as possible similar graph and conclusion repeated for the size of cipher text Fig. 6; even it can be implemented in any classical mode like CBC or ECB[18].

MATERIALS AND METHODS

Padding: This cipher follows one of the Ferguson and Schneier methods for message padding that comes below.

Let P be the plaintext, l(P) be the length of P in bytes and b be the block size of the block cipher in bytes that is arbitrary in MRVLK.

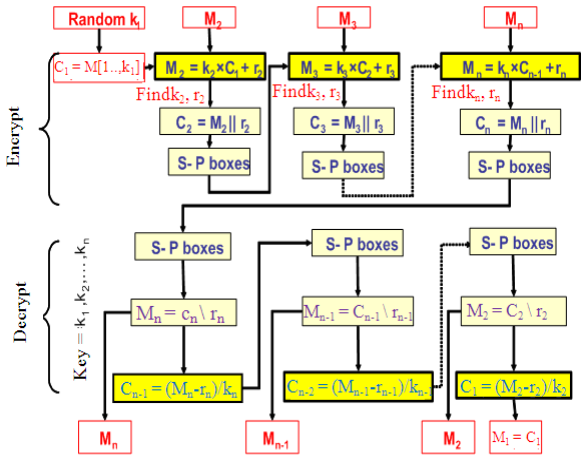

Fig. 7: Block diagram of MRVLK

Append to P a single byte with a value of 128 and then as many zero bytes as necessary to make the overall size a multiple of b.

Encryption: Encryption process starts with generating a random number as K_1 chosen as the size of M_1 equals to C_1 as the first selected message and cipher respectively and r_1 equals zero. The rest of algorithm chooses M_2 bigger than M_1 so that $M_2 = K_2 * C_1 + r_2$ and C_2 concatenating M_2 with r_2. Decryption procedure obviously decipher encrypted message that is C_n. The key and random values are required for deciphering procedure. There is no encryption process if the first key element is bigger than message chunks in size (repetition needed) and it should not be smaller than 35 bits as mentioned before. The algorithm needs at least 7 bits data (that is a character size in ASCII format) to provide a decryption procedure. Length of first chunk is called K_1 and length of second chunk must be at least two times greater than the first chunk and since cipher C_2 consists of second chunk of the message concatenated with residual division of the second chunk per first chunk (size of residual value is chosen same as first chunk for obvious reasons) C_2 length should be three times greater than size of random number and for next stage size of ciphers (C_2, C_3,…) is additive. Diagram of the algorithm is shown in Fig. 7.

RESULTS

Cipher speed (cycles per byte encrypted) is considered as a performance indicator. Because the NIST's AES contest platform of choice was the Intel Pentium, the authors concentrated on that platform. Performance versus other ciphers is demonstrated in Table 1 and encryption time is shown on Table 2.

Table 1: MRVLK-Performance Vs other block ciphers (on a Pentium)

Algorithm	Key length	Width (bits)	Rounds	Average Clks/byte
MRVLK	Variable	Variable	Variable	2.5-13.5
Twofish	Variable	128	16	18.1
Blowfish	Variable	64	16	19.8
Square	128	128	8	20.3
RC5	Variable	64	32	24.8
Cast-128	128	64	16	29.5
DES	56	64	16	43.0
Serpent	128,192,256	128	32	45.0
Safer-128	128	64	8	52.0
Feal-32	64,128	64	32	65.0
IDEA	128	64	8	74.0
Trip-DES	112	64	48	116.0

Table 2: MRVLK-time encryption MRVLK Vs AES (on a Pentium)

64 byte plaintext	Encryption time (m sec)	Decryption time (m sec)
MRVLK	3-12	2-15

Variable time is due to different size of random number. MRVLK's round function encrypts at very few clock cycles per block. Any changes to the procedure evaluated in terms of calculating the cipher for next stage, so that performance would be kept constant.

DISCUSSION

Key generation prevents collision and timing attacks. Long and variable key length prevents exhaustive key search and differential attacks. Non fixed size key avoid replaying in authentication and attacks that can happen on the fixed sized key algorithms. Dynamic length substitution will lead to maximum cryptographic confusion i.e., makes relationship between cipher text and key as complex as possible and finally dynamic length transposition will lead to maximum diffusion i.e., dissipates statistical structure of plaintext over the cipher text to makes the transformation as complex as possible.

CONCLUSION

MRVLK is efficiently implemented and resisted against known attacks because of changing the key and block size in each round where the key generation is an independent process and time variant so it is reliable, fast, complex and hard enough to resists against existing attacks and can implement where there are speed and memory constraints. Differential related key attack is based on key relation so there is no chance of gaining the key by using this method.

REFERENCES

1. Bertoni, G., L. Breveglieri, M. Monchiero, G. Palermo and V. Zaccaria, 2005 AES power attack based on induced cache miss and countermeasure. Proceeding of the International Conference on Information Technology: Coding and Computing, Apr. 4-6, IEEE Computer Society, USA., pp: 586-591. DOI: 10.1109/ITCC.2005.62
2. Acıicmez, O. and C.K. Koc, 2006. Trace driven cache attack on AES. Lecture Notes Comput. Sci., 4377: 271-286. DOI: 10.1007/11967668_18
3. Kocher, P., J. Jaffe and B. Jun, 1999. Differential power analysis. Lecture Notes Comput. Sci., 11: 388-397. http://direct.bl.uk/bld/PlaceOrder.do?UIN=065676280&ETOC=RN&from=searchengine
4. Messerges, T.S., 2001. Securing the AES finalists against power analysis attacks. Lecture Notes Comput. Sci., 1978: 293-301. DOI: 10.1007/3-540-44706-7
5. Daemen, J. and V. Rijmen, 1999. Resistance against implementation attacks: A comparative study of the AES proposals. http://csrc.nist.gov/encryption/aes/round1/conf2/aes2conf.htm
6. Goubin, L. and J. Patarin, 1999. DES and differential power analysis-the duplication method. Proceedings of the Workshop on Cryptographic Hardware and Embedded Systems, Aug. 12-13, ACM Press, Springer-Verlag, pp: 158-172. http://portal.acm.org/citation.cfm?id=752372
7. Bernstein, D.J., 2005. Cache-timing attacks on AES. University of Illinois, Chicago. http://cr.yp.to/antiforgery/cachetiming-20050414.pdf
8. Canteaut, A., C. Lauradoux and A. Seznec, 2006. Understanding cache attacks. Technical Report. ftp://ftp.inria.fr/INRIA/publication/publi-pdf/RR/RR-5881.pdf
9. Neve, M., J.P. Seifert and Z. Wang, 2006. A refined look at Bernstein's AES sidechannel analysis. Proceedings of ACM Symposium on Information, Computer and Communications Security, Mar. 21-24, ACM Press, Taipei, Taiwan, pp: 369-369. http://portal.acm.org/citation.cfm?id=1128887
10. Neve, M., 2006. Cache based vulnerabilities and SPAM analysis. PhD Thesis, Faculty of Science and Application, UCL, Belgium. http://www.cryptologie.be/document/Publications/thesis.pdf
11. Bonneau, J. and I. Mironov, 2006. Cache-collision timing attacks against AES. Proceeding of the Workshop on Cryptographic Hardware and Embedded Systems, Oct. 10-13, Yokohama, Japan, pp: 462-462. http://cat.inist.fr/?aModele=afficheN&cpsidt=19689020
12 .Blomer, J. and V. Krummel, 2007. Analysis of countermeasures against access driven cache attacks on AES. Lecture Notes Comput. Sci., 4876: 96-106. DOI: 10.1007/978-3-540-77360-3
13. Acıicmez, O., 2007. Yet Another Micro Architectural Attack: Exploiting I-cache. Proceedings of the 2007 ACM Workshop on Computer Security Architecture, Nov. 2-2, Fairfax, Virginia, USA., pp: 11-18. http://portal.acm.org/citation.cfm?id=1314469
14. Osvik, D.A., A. Shamir and E. Tromer, 2006. Cache attacks and countermeasures: The case of AES. Lecture Notes Comput. Sci., 3860: 1-20. DOI: 10.1007/11605805
15. Brickell, E., G. Graunke, M. Neve and J.P. Seifert, 2006. Software mitigations to hedge AES against cache-based software side channel vulnerabilities. Cryptology ePrint Archive, Report 2006/052. http://eprint.iacr.org/2006/052.pdf
16. Handschuh, H. and A. Hasan, 2004. Provably secure masking of AES. Lecture Notes Comput. Sci., 3357: 69-83. DOI: 10.1007/b105103
17. Shoichi Hirose, 2006. How to construct double-block-length hash functions. Proceedings of the 2nd Cryptographic Hash Workshop, Aug. 24-25, Santa Barbara, California, pp: 1-17. http://csrc.nist.gov/groups/ST/hash/documents/HIROSE_slide.pdf
18. Mao, W., 2003 Modern Cryptography: Theory and Practice, Prentice-Hall, ISBN: 10: 0-13-066943-1 pp: 648.
19. BlueKrypt, 2009. Cryptographic key length recommendation. http://www.keylength.com
20. Blomer, J., J.G. Merchan and V. Krummel, 2004. Provably secure masking of AES. http://eprint.iacr.org/2004/101/