Structural Cryptanalysis of the Message Based Random Variable Length Key Encryption Algorithm (MRVLK)

Azam Davahi1,*, Hamid Mirvaziri2, Media Aminian3

1Computer Department, Science and Research Branch, Islamic Azad University, Kerman, Iran,

2Computer Department, faculty of engineering, Shahid Bahonar University, Kerman, Iran,

3Computer Department, Science and Research Branch, Islamic Azad university, Kerman, Iran

azam.davahi@yahoo.com, hmirvaziri@gmail.com, media.aminian@yahoo.com

Article history:
Received July 2014
Accepted August 2014
Available online August 2014

Abstract
This article has presented a Structural cryptanalysis on MRVLK (Message Based Random Variable Length Key Encryption). In this cipher, key length is started from small amount of bits and then will be grown in size. The cipher has variable rounds, random bitwise rotations and dynamic key length that provide resistance to linear and differential cryptanalysis. In spite of these advantages, some disadvantages are observed such as correlation between the ciphertexts in each stage which facilitates structural attack. Even random mechanism such as S-box in this cipher cannot prevent this attack. The attack performs analysis on the final ciphertext and reveals the plaintext of MRVLK by exploiting the fact that the structure of the ciphertext is obvious and weak. The presented attack efficiently utilizes this information and prompts the operations cryptanalysis. Performance of this attack is evaluated in terms of running time. The results show that the original plaintext is achievable to minimal cost.
Keywords: Cryptanalysis, Block Cipher, MRVLK, Structural Attack, Random key.

1. Introduction

Structural cryptanalysis is the branch of cryptology which breaks the ciphertext according to structural weaknesses without having any information about secret key. Structural attacks often show a theoretical understanding of fundamental constructions. Therefore, they are very useful to establish design rules for strong cryptosystems [1]. Block cipher algorithms are divided into two classes: probabilistic and non-probabilistic [2]. The first cryptology algorithm has been proposed by [3]. They have proved that extracting any information from the ciphertext is hard. After that, other approaches have been proposed. In [4], novel approach of efficient multimedia content encryption scheme has been introduced which uses a block of bits rather than bytes or pixels. The proposed block cipher encrypts any type of compressed multimedia content by random substitution using binary tree [5] traversal, row shifting and column shifting. In [6], proposed identity-based secret public keys. This new identity-based approach allows secret public keys to be constructed in a very natural way using arbitrary random strings, eliminating the structure found in, for example, RSA or Diffie-Hellman keys [7]. VBDEM (Variable size Block Encryption using Dynamic-key Mechanism) [8] is one of the other techniques which has been designed with unlimited key size, dynamically changing permutation table based on the encryption key and variable block size for each round and also compression technique based on the key. The employed compression method is only for strengthening its encryption. MRVLK is also presented in 2009 by Mirvaziri et al [9]. MRVLK uses a random and non-deterministic method for encryption and produces the key and invertible S-Boxes for bitwise rotations. This cipher is a non-corresponding cipher with this structure :\( \{0, 1\}^n \rightarrow \{0,1\}^m \) where \( n \) can be smaller or greater than \( m \). Although experiments show that probabilistic algorithms are resistant against various attacks but still they are more likely to attack [10,11]. For example, MRVLK has some structural weaknesses and disadvantages which has caused structural attack to be designed and implemented. In general, this paper discusses performance, strengths and weaknesses of this algorithm and according to its weaknesses a structural attack [12, 13] has been designed and implemented. Finally, attack implementation results have been presented in matlab programming environment.

2. Related Work

The paper in [14] has presented an attack on PRESENT-like ciphers with key-dependent S-boxes. Where the S-boxes are chosen uniformly at random for each round, and where the bit permutation is key-dependent as well, the presented cryptanalysis in [14] can be applied. In [15], a cryptanalysis for block ciphers has proposed that contains secret components, typically S-boxes. In this paper, the proposed attack has been applied on two well known ciphers, AES and Camellia; these ciphers use 8-bit S-boxes but are structurally very different, and our attack adapts accordingly. Also, [16] shows a basic heuristic methodology and a framework for constructing families of distinguishers and introduces differential sets of a special new form dictated by the regular structure of GOST. GOST is a well-known block cipher implemented in standard libraries such as OpenSSL. In GOST two main elementary methods of ciphering the information are used with some modifications [17].

3. MRVLK Block Cipher Algorithm

MRVLK is a family of new probabilistic block ciphers. Variable key selection is dependent on message size and completely random. This algorithm has a variable block size and a variable key size. Because of this, the produced ciphertext can be grown in size. This cipher has variable rounds, random
3.1. Encryption and Decryption Process

At the beginning of MRVLK algorithm, a random number RN will be initialized with a value between 7 and 61. Then, the first chunk of message (M₀) is separated from the main plaintext which is equal to the random number RN. So, in each round, we have a ciphertext (C₀, C₁, ..., Cₙ). At the first round, C₀ and R₀ are equal to M₀ and 0, respectively. R₁ refers to the remainder of the division. Afterwards, the second chunk of message M₁ will be selected (generally the main message is divided into M₀, M₁, M₂, ..., chunks in bits, each chunk is equal to RN, 2RN, 4RN, ..., respectively). Sometimes padding is used to adjust the length of the last chunk.

(1) Mₙ = Kₙ * Cₙ⁻¹ + rₙ

According to the mentioned formula 1, K₁ and R₁ are calculated by M₁/C₀ and M₁% C₀ and then, C₁ is obtained by merging of M₁ and R₁ (C₁ = M₁ || R₁). According to Fig. 1, this process applies to all generated chunks, until the last Cₙ (final ciphertext) has been determined. At the end, final ciphertext (last Cₙ) will be sent to the receiver with the generated key (Key = K₁ || K₂ || ... || Kₙ). In the decryption process, the ciphertext has been separated into two equal chunks (Mₙ and Rₙ). Key is divided into two equal parts, as well. Now, according to the above formula, Ci can be achieved for i = n-1, ..., 1.

---

**Fig. 1 Block Diagram of MRVLK[9].**
3.2. Advantages

MRVLK algorithm is resistant against key exhaustive search and differential [26] attack because of long and variable key length. In non-probabilistic ciphers, key generation is a deterministic function and there is a correlation between the keys in each stage, so generating sub keys are known. Even though, this procedure is chosen carefully and independent from key length. In MRVLK there is no relationship between the keys in each stage and the size of the key which makes MRVLK secure against cache [27].

Table 1. Examples where plaintext and corresponding ciphertext are equal

<table>
<thead>
<tr>
<th>Ciphertext Length</th>
<th>Ciphertext</th>
<th>Plaintext</th>
</tr>
</thead>
<tbody>
<tr>
<td>15</td>
<td>1101101111001</td>
<td>1101101111001</td>
</tr>
<tr>
<td>19</td>
<td>1100001</td>
<td>1100001</td>
</tr>
<tr>
<td>20</td>
<td>111001001</td>
<td>111001001</td>
</tr>
</tbody>
</table>

The cipher has variable rounds, random bitwise rotations and dynamic key length with a well designed key schedule which makes it resistant against linear [26]. Also, variable block size and variable key size in MRVLK, makes this algorithm more powerful against SPA [28], DPA [29] and timing [30, 31] attacks.

3.3. Weakness

Research on MRVLK algorithm indicates that this algorithm, like any other encryption algorithms has some disadvantages. These disadvantages are related to the structure of this algorithm and facilitate cryptanalysis of this algorithm. There are some states in which the plaintext without any changes considered as a ciphertext. Some of these states are as follows:

- If the random number is greater than or equal to the plaintext length, then the plaintext appears without any changes in the output as ciphertext. In other words, the plaintext and the ciphertext are alike.

- If the ciphertext length is odd, then certainly the random number is equal to the ciphertext length. In this case, the ciphertext is equal to plaintext as shown in the second row in table.

- If the ciphertext length is even and the number of zeros in padding is more than half of the ciphertext length, then the random number is equal to the ciphertext length and the plaintext appears in the ciphertext entirely with no change as shown in table.1.

4. Design and Implementation of A Structural Attack on MRVLK

By using the sequence division according to formula 1, we can obtain some plaintexts in each state. So, at the end of this attack, we will have a lot of possible plaintext corresponding to a given ciphertext. So, for increasing the performance, we have to eliminate some of this possible plaintext. Because of this, we need the primary RN. We can send one of the possible plaintext and RN to the MRVLK algorithm as the inputs. If the output of MRVLK is equal to the given cipher text, the considered the plaintext as the input is the real plaintext. On the other hand, we need a way to guess the primary RN. But, we cannot guess the real RN certainly, and we will have a few RNs.

4.1. Generation Random Numbers for Decryption

As mentioned in the previous section, some relationships between M and R (Cn =Mn || Rn) facilitate cryptanalysis for attacker. In this way, attacker can obtain the random number RN through the above
relation to decrypt the ciphertext. Operations of this attack for obtaining the RNs will start immediately after receiving the ciphertext. To do this, if ciphertext length is odd, cipher text length is assigned to RN and then cryptanalysis process will start to obtain the plaintext. If cipher text length is even, attacker will check whether the number of added zeros due to padding in the last chunk is more than half of cipher text length or not. If this circumstance is satisfied, then RN will be equal to cipher text length. Therefore, cryptanalysis process will be started to obtain the plaintext. If the number of added zeros to ciphertext is less than half of the cipher text length and also less than 61, then we have more than one RN. In such circumstances, the first RN is equal to the cipher text length. To obtain the other possible RN, ciphertext length is divided by 2. Divisions continue until the result is even and is greater than 7. If RN is more than 61, then the first RN cannot be equal to cipher text length. So division will start directly to achieve possible random numbers. After each division, its results will be placed in the list of possible random numbers. Corresponding pseudo-code of obtaining random number is shown in Fig. 2. The “zerocount” variable in Fig. 2 refers to the zeros which have added to the last chunk Mn. In MRVLK, the count of zero of padding has been kept in a field at the end of the ciphertext in a hexadecimal format.

4.2. Breaking the Ciphertext
For breaking the ciphertext, first he checks whether any of the obtained random numbers are equal to ciphertext length and if this is satisfied, the plaintext will be equal to the same ciphertext. Otherwise, Mn and rn will be extracted from the final ciphertext and C will be specified according to \( r_n = M_n \mod C_{n-1} \). Therefore, in each stage, we will obtain a set of possible Cs. This process will be applied to each possible C. This procedure will be continued till the length of C0 is equal to one of given random number which has obtained in section 3.1. There is a corresponding M for each C, so plaintext can be achieved by merging the obtained M’s in each stage. The corresponding pseudo-code of ciphertext breaking is shown in Fig.3.

```
function [RN]=Random Number(ciphertext)
    C=ciphertext
    len=length(C)
    i=1
    if (mod(len,2) ~=0)
        RN=len
    else
        hlen=len/2
        if (hlen<=zerocount)
            RN[1,i]=len
        else
            if (len<=61)
                RN[1,i]=len
                i=i+1
            end
        while (mod (hlen,2)==0 && hlen/2>=7)
            hlen=hlen/2
            RN[1,i]=hlen
            i=i+1
        end
    end

Fig. 2 pseudo-code of obtaining random number.
```
function [plaintext]=hack(ciphertext)
cipher = substring(ciphertext, 0, length(ciphertext))
n = length(cipher)
M = substring(cipher, 0, n/2)
r = substring(cipher, n/2, n-1)
c = 0
while (c <= M)
    if (mod(M, c) == r)
        plaintext = [plaintext || M]
        hack(c)
    end
    c = c + 1
end

Fig. 3 pseudo-code of ciphertext breaking.

5. Results

In this section outputs of proposed attack have been presented and then evaluated based on various parameters in terms of running time and performance. It should be noted that the required time to break S-Box is not considered in total running time. It is preferred to discuss all the results in detail in case of original research paper. To explain observed data you can use figures, graphs and tables.

5.1. Data Verification

Since this attack may generate several plaintexts, each of the plaintexts with the given key is considered as the input of MRVLK. If the output of MRVLK is equal to the analytical ciphertext, then the original plaintext is obtained. Some examples of attack results are demonstrated in Table 2. Table 2 shows the output of the attack for a given plain text. For example, for “Notebook” as the plain text, we will have 4 outputs. In mentioned attack, when one of the three states occurs, the original plaintext is obtained by cryptanalysis with a comparison and without any division. Hence, time length of attack is too small. Otherwise, time length of attack will depend on two parameters: the difference and length of the ciphertext. “L-R” columns in Tables 3, 4 and 5 represent the difference between right and left parts of the ciphertext. L indicates the left part and R indicates the right part. For example, if the ciphertext is “000000011011010010000000000001001101” then the left part(L) is “000000011011010010000000000001001101” and the right part(R) is “00000000001001101101” and the decimal equivalent for each part are 1641 and 72, respectively. Therefore, the L – R or (the difference between these two) is 1564.

5.2. Ciphertext length effect on time of attack

The effect of ciphertext length on time length of attack is shown in Table 3. As observed in this table, if the difference of left and right sides are equal to each other in several ciphertexts, ciphertext length has the greatest effect on the duration of attack. The attack time for shorter ciphertext is smaller. As the ciphertext length increases, the time length of attack increases as well. Time attack for different ciphertext lengths has shown in Fig. 4.
5.3. The Effect of Difference Between Left and Right Parts on the Time Length of The Attack

The effect of the changes in the difference in ciphertexts on the time of the attacks are shown in Table 4 and Fig. 5. In this table, when the lengths of all ciphertexts are equal to each other, effective parameter on the period time of attack is the ciphertext difference.

5.4. A Ciphertext difference and length effects on time length of attack

The effect of changes in length and difference of ciphertext on attack time is shown in table 5. Experiments show that if both parameters of length and difference are variable, then effective parameters on the period time of attack, is the difference between left and right side of ciphertext. In this case, it is possible to have a ciphertext with greater length and smaller attacking time as can be seen in table 5 (Examples 1 and 3). Comparing effectiveness of ciphertext length and difference on the period time of attack is shown in Fig. 6 and Fig. In these Figures the period time of attack increases with increase in difference and without considering ciphertext length changes. Thus, it can be concluded that the period time of attack is shorter for the ciphertext with the smaller difference when both length and difference are changing.
6. Conclusion

In this paper, a new method to attack MRVLK block cipher algorithm was presented. Structural weaknesses are applied in proposed attack, and it does not require any knowledge of secret key. This attack is composed of two operations: obtaining random number and breaking the ciphertext. After the implementation of this attack on MRVLK, results of the attack is described on different examples to verify its data and, its performance is evaluated regarding the running time. This evaluation confirms that the period time of attack changes in accordance with changes in weight difference and length, so that the period time of attack is shorter for smaller ciphertext weight difference and shorter lengths. In the future, we intend to suggest a solution to resist MRVLK against this new attack and improve the algorithm security.

Table 2. Implementation results of attack on MRVLK

<table>
<thead>
<tr>
<th>Plain Text</th>
<th>CipherText</th>
<th>Attack Output</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>01100001</td>
<td>A</td>
</tr>
<tr>
<td>Davahli</td>
<td>110000111010001101100110100111100001111011000000000100010</td>
<td>avahli, Davahli</td>
</tr>
<tr>
<td>Notebook</td>
<td>000000000011011101110111011111111111101000000011000</td>
<td>otebook, Notebook, 4otebook, Davahli</td>
</tr>
</tbody>
</table>
Table 3. Ciphertext length effect on time length of attack

<table>
<thead>
<tr>
<th>Ciphertext</th>
<th>CipherText Length</th>
<th>L-R</th>
<th>Attack Time (ms)</th>
</tr>
</thead>
<tbody>
<tr>
<td>000000011001101001 0000000000010011101</td>
<td>36</td>
<td>1564</td>
<td>15.8346</td>
</tr>
<tr>
<td>000000000011011001 0000000000010011101</td>
<td>40</td>
<td>1564</td>
<td>19.7968</td>
</tr>
<tr>
<td>00000000000000001101001 0000000000000001001101</td>
<td>42</td>
<td>1564</td>
<td>20.5960</td>
</tr>
<tr>
<td>00000000000000001101001 0000000000000000001001101</td>
<td>44</td>
<td>1564</td>
<td>24.4170</td>
</tr>
</tbody>
</table>

Table 4. Ciphertext weight difference effect on time length of attack

<table>
<thead>
<tr>
<th>Plain Text</th>
<th>Ciphertext</th>
<th>L-R</th>
<th>Attack Time (ms)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1110100110 0001111000000000000010001000</td>
<td>12528</td>
<td>47.4479</td>
<td></td>
</tr>
<tr>
<td>1110100110 01011100001</td>
<td>12992</td>
<td>50.5415</td>
<td></td>
</tr>
<tr>
<td>1100100110 111111110001000000000000010001</td>
<td>14000</td>
<td>53.4761</td>
<td></td>
</tr>
<tr>
<td>1101110111 0101101101000000000010011100</td>
<td>15070</td>
<td>57.7548</td>
<td></td>
</tr>
</tbody>
</table>

Table 5. Ciphertext length effect on time length of attack

<table>
<thead>
<tr>
<th>Ciphertext</th>
<th>CipherText Length(bit)</th>
<th>L-R</th>
<th>Attack Time (ms)</th>
</tr>
</thead>
<tbody>
<tr>
<td>000000000011011001001 0000000000010011101</td>
<td>42</td>
<td>1564</td>
<td>20.5960</td>
</tr>
<tr>
<td>000000000000000001101001 0000000000000000001001101</td>
<td>44</td>
<td>1568</td>
<td>24.4170</td>
</tr>
<tr>
<td>11011111111010000000000000110000</td>
<td>28</td>
<td>14300</td>
<td>53.4761</td>
</tr>
<tr>
<td>11011111111011000000000000110000</td>
<td>30</td>
<td>16264</td>
<td>57.7548</td>
</tr>
</tbody>
</table>

References