IEICE TRANS. INF. & SYST., VOL.E99–D, NO.4 APRIL 2016

LETTER

A Meet-in-the-Middle Attack on Reduced-Round Kalyna-$b/2b$

Riham ALTAWY[†a)], ***Member***, **Ahmed ABDELKHALEK**[†b)], ***and*** **Amr M. YOUSSEF**[†c)], ***Nonmembers***

SUMMARY In this letter, we present a meet-in-the-middle attack on the 7-round reduced block cipher Kalyna-$b/2b$, which has been approved as the new encryption standard of Ukraine (DSTU 7624:2014) in 2015. According to its designers, the cipher provides strength to several cryptanalytic methods after the fifth and sixth rounds of the versions with block length of 128 and 256 bits, respectively. Our attack is based on the differential enumeration approach, where we carefully deploy a four-round distinguisher in the first four rounds to bypass the effect of the carry bits resulting from the prewhitening modular key addition. We also exploit the linear relation between consecutive odd and even indexed round keys, which enables us to attack seven rounds and recover all the round keys incrementally. The attack on Kalyna with 128-bit block has a data complexity of 2^{89} chosen plaintexts, time complexity of $2^{230.2}$ and a memory complexity of $2^{202.64}$. The data, time and memory complexities of our attack on Kalyna with 256-bit block are 2^{233}, $2^{502.2}$ and 2^{170}, respectively.

key words: *Kalyna, cryptanalysis, meet-in-the-middle attacks, differential enumeration, DSTU 7624:2014*

1. Introduction

Kalyna [1] is an SPN cipher that won the Ukrainian national public cryptographic competition, which aimed to select a block cipher to become the new Ukrainian national encryption standard known as DSTU 7624:2014 [2]. Kalyna supports block sizes of 128-bit, 256-bit, and 512-bit, and key sizes of 128-bit, 256-bit, and 512-bit, where the key size can be equal to or double the block length. In this letter we will refer to a specific version of the cipher as Kalyna-b/k, where b and k denote the employed block and key lengths, respectively. Although the detailed analysis of the resistance of Kalyna to various attacks has not been described by its designers, they concluded that the cipher is sufficiently secure against several cryptanalytic methods after rounds five and six when the block size is 128-bit and 256-bit, respectively (cf. page 14 of [1], [3]).

Demirci and Selçuk were the first to apply the MitM attack on AES [4]. They employed a distinguisher in the middle four rounds and showed that if its input has one byte that takes all the possible values, then each output byte can be evaluated as a function of 25 byte parameters. They also showed that the values of each output byte corresponding to the input byte values form an ordered sequence that can be used as a property to identify the right key guess. The attack has a high memory complexity, which is required to store 2^{200} ordered sequences. Moreover, Dunkelman *et al.* addressed the high memory requirements of the attack in [4] by introducing multisets and differential enumeration [5]. While the use of multisets provides better encoding of the ordered sequences, differential enumeration reduces the number of parameters to 16 bytes, which reduces the memory complexity to 2^{128}. Later on, Derbez *et al.* [6] utilized the rebound approach [7] to further reduce the number of parameters to 10 bytes, thus the memory complexity is reduced to 2^{80}. Moreover, a 9-round attack on AES-192 was presented by Li *et al.* in [8].

In this work, we present a MitM attack on seven round reduced Kalyna-$b/2b$ utilizing the idea of efficient differential enumeration. Kalyna employs a pre- and post-whitening key mixing using modulo 2^{64} addition. Accordingly, we deploy a specific four-round distinguisher that covers the first four rounds, where the active byte is chosen to prevent the propagation of differences to the neighboring bytes. We also exploit the linear relation between odd and even indexed round keys to efficiently recover the last two round keys. The key schedule of Kalyna is designed to make it computationally infeasible to retrieve the master key from the round keys. For that reason, we propose an approach to recover all the round keys using parameters matching. Employing this proposed technique, we use the parameters corresponding to the matching multiset to filter pairs of two consecutive round keys guesses.

2. Specifications of Kalyna

Our attack targets Kalyna-$b/2b$, where the size of the key is double that of the state. Accordingly, in this section, we give a description of the encryption and round key generation procedures of Kalyna-$b/2b$. The encryption procedure updates an $8 \times c$-byte state for 14 and 18 rounds, where c denotes the number of columns in the block state and is equal to 2 and 4 for the 128 and 256-bit block, respectively. The encryption procedure employs a pre- and post-whitening stages using modulo 2^{64} addition applied on the state columns independently. Each round applies the following transformations on the state:

Manuscript received August 6, 2015.
Manuscript revised December 3, 2015.
Manuscript publicized January 22, 2016.

[†]The authors are with Concordia Institute for Information Systems Engineering, Concordia University, Montréal, Québec, Canada.
a) E-mail: r_altaw@encs.concordia.ca
b) E-mail: abdelk_a@encs.concordia.ca
c) E-mail: youssef@ciise.concordia.ca
DOI: 10.1587/transinf.2015EDL8174

- SubBytes (SB): A layer of 8-bit substitution boxes.
- ShiftRows(SR): A transformation that cyclically right shifts the rows of the state. The value of the shift is given by $\lfloor \frac{i \cdot b}{512} \rfloor$, where $i = 0, 1, \cdots 7$ and $b = 128$ or 256 denote the row number and state size, respectively.
- MixColumns(MC): A transformation that multiplies the columns of the state independently by an MDS matrix.
- X: A round key mixing layer consisting of xoring the state with the round keys.

In the last round, the X transformation is replaced by a post-whitening modular key addition. In our analysis, we use the following property of the substitution layer:

Proposition 1: Given two non-zero differences in $\mathbb{F}_{256}$, Δ_x and Δ_y, the average number of solutions for $SB(x) \oplus SB(x \oplus \Delta_x) = \Delta_y$ is one.

Key schedule. Even indexed round keys are independently evaluated from the master key, K. The key state is first initialized by a master key dependent value, which undergoes two encryption rounds. Odd indexed round keys are linearly computed from their previous even indexed round keys according to the formula: $K_i = K_{i-1} \lll (b/4 + 24)$. For further details regarding the SBoxes, the linear transformation or the key schedule of other versions, the reader is referred to [1]. The following notation is used throughout the letter:

- x_i, y_i, z_i, and w_i: the 8×2 bytes state after the X or addition modulo 2^{64}, SB, SR, and MC transformations at round i, respectively.
- $x_i[j]$: The j^{th} byte of the state x_i, where $j = 0, 1, \cdots, 15$, and the bytes are indexed column wise.
- x_i^j: The state at round i, and j denotes the order of state x_i within a sequence.
- $x_i[j \cdots k]$: The bytes between the j^{th} and k^{th} positions inclusive of the state x_i.
- Δx_i, $\Delta x_i[j]$: The difference at state x_i and byte $x_i[j]$, respectively.
- $\Delta^j x_i$: The difference at state x_i, and j denotes the order of state x_i within a set or a sequence.

We measure the memory complexity of our attack in b-bit Kalyna-$b/2b$ blocks and the time complexity in reduced-round Kalyna-$b/2b$ encryptions. In the following section, we give the details of our MitM attack on Kalyna-128/256.

3. A Differential Enumeration MitM Attack on Kalyna-128/256

The utilized distinguisher is a truncated differential characteristic such that, when a set of input states from a δ-*set* is used as its input, the set of a given one byte difference of the output state forms an ordered sequence, which can be represented using a multiset. Our middle distinguisher is a truncated differential characteristic such that, when a set of input states from a δ-*set* [9] is used as its input, the set of a given byte difference of the output state forms an ordered sequence, which can be represented using a multiset.

Definition 1 (δ-set of Kalyna-128/256): Let a δ-set be a set of 256 Kalyna-128/256 states where one byte at a particular state position takes all the 2^8 possible values and the rest of the 15 bytes are constants.

Definition 2 (Multisets of bytes): A multiset generalizes the set concept by allowing elements to appear more than once. In our case, a multiset of 256 bytes can take as many as $\binom{2^8 + 2^8 - 1}{2^8} \approx 2^{506.17}$ different values [5].

We employ a distinguisher in the form of $1 \rightarrow 8 \rightarrow 16 \rightarrow 8 \rightarrow 4$. The distinguisher starts at x_0, where byte $x_0[15]$ takes all possible 2^8 values and ends at z_4, where we evaluate the multiset of one out of the four-byte differences in $z_4[0 \cdots 3]$. We specifically locate the distinguisher in the first four rounds, which enables us to exploit the linear relation between the last two round keys and attack seven rounds. Additionally, we choose the active byte at the beginning of the distinguisher in the most significant byte of the second column to prevent the propagation of the difference to the neighboring bytes, which happens due to the carry propagation resulting from the modular addition key mixing. On the other hand, placing the distinguisher in the middle as in the traditional setting [4] allows us to attack six rounds only. Additionally, we must deal with the probabilistic carry propagation in the analysis of the first round, which reduces the path probability, and hence both the data and time complexities of the attack are increased. It should be noted that while our distinguisher ends with four active bytes, which increases the path probability when we evaluate the multiset from the ciphertext side, this distinguisher does not affect the memory complexity because we store a multiset of the differences in only one of the four active bytes. In other words, since each byte out of the four active bytes at the end of the distinguisher forms an ordered sequence, we can choose any of them to distinguish between key candidates as long as the probability of error is negligible. We denote the δ-*set* at state x_0 by δs, where

$$\delta s = \{x_0^0, x_0^1, \cdots, x_0^{255}\}.$$

We also denote the set of 255 differences at bytes $z_4[0 \cdots 3]$ by ds, where

$$ds = \{\Delta^1 z_4[0 \cdots 3], \Delta^2 z_4[0 \cdots 3], \cdots, \Delta^{255} z_4[0 \cdots 3]\},$$

and $\Delta^l z_4[0 \cdots 3] = z_4^0[0 \cdots 3] \oplus z_4^l[0 \cdots 3]$, for $l = 1, 2, \cdots, 255$. The multiset at z_4 is evaluated by the knowledge of the values of 37 bytes, $\Delta^l y_0[15]$, 8 bytes at $x_1[8 \cdots 15]$, 16 byte at x_2, 8 bytes at $x_3[0 \cdots 3]$, $x_3[12 \cdots 15]$, and 4 bytes at $x_4[0 \cdots 3]$. However, by employing the rebound based differential enumeration technique [6], we deduce that if x_0^0 of the δ-set belongs to a pair of plaintexts that conforms to the differential path in Fig. 1, then the corresponding multiset can be computed by the knowledge of 25 byte parameters only. These parameters are

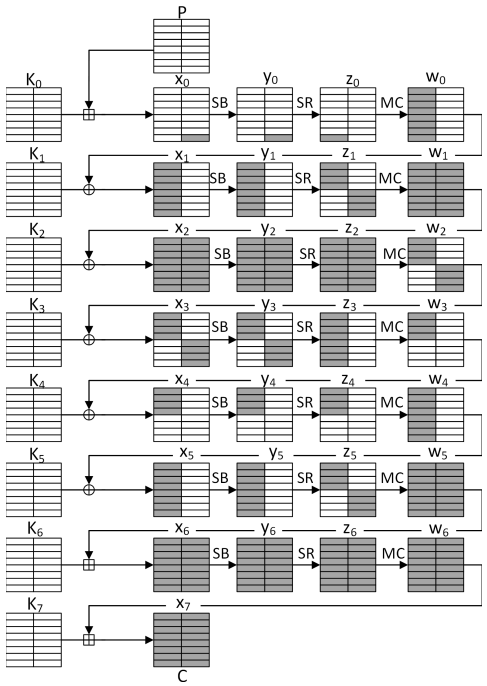

Fig. 1 Differential path used in the attack on the 7-round reduced Kalyna-128/256.

$\Delta y_0[15]$, $x_1[8\cdots15]$, $x_3[0\cdots3]$, $x_3[12\cdots15]$, $x_4[0\cdots3]$, and $\Delta z_4[0\cdots3]$, where $\Delta y_0[15]$ and $\Delta z_4[0\cdots3]$ denote the differences generated by a conforming message pair.

3.1 Attack Procedure

The attack is composed of precomputation and online phases. The online phase is further divided into data collection and key recovery phases.

Precomputaion phase: In this phase, we build a lookup table that contains 2^{200} multisets of the 255 difference in one of the byte differences in z_4. Using the rebound approach, we iterate over the 2^{200} possible values of the 25 bytes $\Delta y_0[15]$, $x_1[0\cdots7]$, $x_3[0\cdots3]$, $x_3[12\cdots15]$, $x_4[0\cdots3]$, and $\Delta z_4[0\cdots3]$ to construct 2^{200} multisets of differences. The procedure can be summarized as follows:

- For each of the 2^{200} values of $\Delta y_0[15] \parallel x_1[0\cdots7] \parallel x_3[0\cdots3], x_3[12\cdots15] \parallel x_4[0\cdots3] \parallel \Delta z_4[0\cdots3]$, evaluate the value of x_2 as follows:
  1. Linearly propagate $\Delta y_0[15]$ forward to evaluate $\Delta x_1[0\cdots7]$.
  2. Using $x_1[0\cdots7]$, deduce Δx_2.
  3. Compute $\Delta y_3[0\cdots3]$, $\Delta y_3[12\cdots15]$ using $\Delta z_4[0\cdots3]$ and $x_4[0\cdots3]$.
  4. Using $x_3[0\cdots3]$, $x_3[12\cdots15]$, evaluate Δy_2.
  5. Find x_2, such that $SB(x_2) \oplus SB(x_2 \oplus \Delta x_2) = \Delta y_2$. According to proposition 1, we get one solution on average.
- Having the value of x_2, we can now compute the 255 unordered differences $\Delta^i z_4[0\cdots3]$ in ds for $i = 1, 2, \cdots, 255$ as follows:
  1. Set $\Delta^i y_0[15] = i$. As the SBox is a permutation over $\mathbb{F}_{256}$, the sequence of $\Delta y_0[15]$ corresponds to the unordered sequence of $\Delta x_0[15]$ of the *delta*-set.
  2. Linearly propagate $\Delta^i y_0[15]$ forward and compute the value of $\Delta^i x_1[0..7]$.
  3. Using the value of $x_1[0\cdots7]$ and $\Delta^i x_1[0\cdots7]$, pass the substitution layer with certainty and evaluate $\Delta^i x_2$.
  4. Using the value of x_2 and $\Delta^i x_2$, evaluate $\Delta^i x_3[0\cdots3]$, $\Delta^i x_3[12\cdots15]$.
  5. Propagate $\Delta^i x_3[0\cdots3]$, $\Delta^i x_3[12\cdots15]$ with the knowledge of $x_3[0\cdots3]$, $x_3[12\cdots15]$ through the Sboxes to evaluate $\Delta^i x_4[0\cdots3]$.
  6. Using $x_4[0\cdots3]$ and $\Delta^i x_4[0\cdots3]$, compute $\Delta^i z_4[0\cdots3]$.
- Now we get 255 4-byte differences in z_4 corresponding to the 255 values of the δs, so we pick one active byte position from the four active bytes in z_4 and represent its 255 difference values using a multiset. In other words, we represent a sequence of one out of the four generated byte differences in z_4, (i.e., $\Delta z_4[j]$, $j = 0, 1, 2, 3$) using a multiset and store it in a hash table.

Data collection In this stage, we query the encryption oracle with structures of chosen plaintexts to get enough pairs such that one of them conforms to the whole truncated differential path. The employed structure is a set of 256 plaintexts, which are equal in the least significant 15 bytes and the most significant byte takes all the possible 256 values. For each structure, we variate the most significant state byte and set the remaining 15 bytes to a constant value, thus one active byte in the plaintext propagates to one active byte in x_0 with certainty. A structure results in about $\frac{2^8\times(2^8-1)}{2} \approx 2^{15}$ pairs. While a chosen plaintext pair follows the forward path from the plaintext to state x_0 with certainty, the probability that its corresponding ciphertext pair conforms to the backward path from the ciphertext to state z_4 is 2^{-96}. This probability is due to the $16 \rightarrow 8$ and $8 \rightarrow 4$ transitions through the inverse MixColumn transformation in rounds six and five, respectively. Accordingly, it is expected that when trying 2^{96} plaintext pairs, a ciphertext pair corresponding to one of them follows the path in Fig. 1. Since, each structure provides 2^{15} pairs, one requires about 2^{81} structures. All in all, we ask for the encryption of $2^{81}\times2^8 = 2^{89}$ chosen plaintexts to get the required 2^{96} pairs.

Key recovery: In this stage, for each plaintext pair (P_0, P'_0), we pick P_0 and construct the rest of the 255 plaintexts in its *delta*-set by $P_i = P_0 \oplus i$ for $i = 1, 2, \cdots, 255$. Then, we get their corresponding 256 ciphertexts C_i for $i = 0, 1, \cdots, 255$, partially decrypt them using guesses for K_7 and K_6 to get the 255 differences $\Delta^i z_4[0\cdots3]$, evaluate the multiset and match it with the precomputed table. In this stage, we exploit the linear relation between even

and odd indexed round keys to identify the right K_7 and K_6 by guessing the 128-bit K_7 only and getting K_6 candidates for free. The probability of a false match is given by $2^{200+96+128-467.6} = 2^{-43.6}$ which is negligible. Note that the probability of randomly having a match in the table is $2^{-467.6}$ (and not $2^{-506.7}$ which is the numbers of different values a multiset can represent) because the number of ordered sequences associated to a multiset is not constant (the proof is provided in section 4 of [10]).

Attack Complexity: The memory requirement of the attack is given by $2^{200} \times 512/128 = 2^{202}$ 128-bit blocks. The data complexity is 2^{89} chosen plaintexts. The time complexity of the offline phase is due to performing 2^{200} partial encryptions of 256 messages, which is equivalent to $2^{200+8} \times 5/7 \approx 2^{208}$ encryptions. The time complexity of the online phase to recover K_7 and K_6 is given by $2^{96+128+8} \times 2/7 \approx 2^{230.2}$.

3.2 Recovering the Remaining Round Keys

Even indexed round keys are generated independently from the master key through 2 rounds of encryption, which makes the recovery of the master key computationally infeasible. For this reason, we recover all the eight round keys using the plaintext pair and its corresponding ciphertext pair, which resulted in a match in the precomputed table and the set of parameters used in the offline computation of the matching multiset. In the precomputation phase, we store the values of the parameters $x_1[0 \cdots 7]$, x_2, $x_3[0 \cdots 3], x_3[12 \cdots 15]$, and $x_4[0 \cdots 3]$ along with the multiset in the table entries. Then, we perform incremental filtering to recover the remaining round key. In other words, we guess K_5 and evaluate x_4. It is expected that 2^{96} candidates survive parameters matching. Then, we evaluate K_4 candidates from K_5 and filter the 2^{96} (K_5, K_4) candidates by computing x_3 and comparing it with the retrieved parameters. We repeat this incremental filtering with K_3, K_2, K_1, and K_0 candidates until one guess for the six round keys remains. The memory complexity is slightly increased since we store an additional 36 bytes in each entry and accordingly is given by $2^{200} \times (512 + 36 \times 8)/128 = 2^{202.64}$ 128-bit blocks. The time complexity of this stage is equal to $2^{128} \times 1/7 + 2^{96} \times 2/7 + 2^{160} \times 3/7 + 2^{32} \times 4/7 + 2^{128} \times 1/7 \approx 2^{159}$. Consequently, the online time complexity of the attack remains the same and is given by $2^{230.2}$.

4. Analysis of the Attack on Kalyna-256/512

Our MitM attack can also be applied on Kalyna-256/512, where the state has four columns. The attack steps are similar to the steps on Kalyna-128/256 except the precomputation phase. As depicted in Fig. 2, the distinguisher ends in two active bytes in z_4. Accordingly, we only need 21 byte parameters, which are $\Delta y_0[31]$, $x_1[16 \cdots 23]$, $x_3[0, 1]$, $x_3[26, 27]$, $x_3[20, 21]$, $x_3[14, 15]$, $x_4[0, 1]$, and $\Delta z_4[0, 1]$. We assume the independence of both the ordered sequences generated by individual bytes and their corresponding multisets. Accordingly, we store two multisets of differences in the two active bytes of z_4 so that the probability of false matches is very low. Hence, the memory complexity of the attack is $2^{168} \times 1024/256 = 2^{170}$. The data complexity is determined according to the path probability, which is equal to 2^{-240}. Thus the data complexity of the attack is given by $2^{225+8=233}$ chosen plaintexts. The time complexity of the offline phase is due to performing 2^{168} partial encryptions on 256 messages, which is equivalent to $2^{168+8} \times 5/7 \approx 2^{176}$ encryptions. The time complexity of the online phase is given by $2^{240+256+8} \times 2/7 \approx 2^{502.2}$. The probability of a false match is given by $2^{168+240+256-467.6\times2} = 2^{-271.2}$, which is negligible.

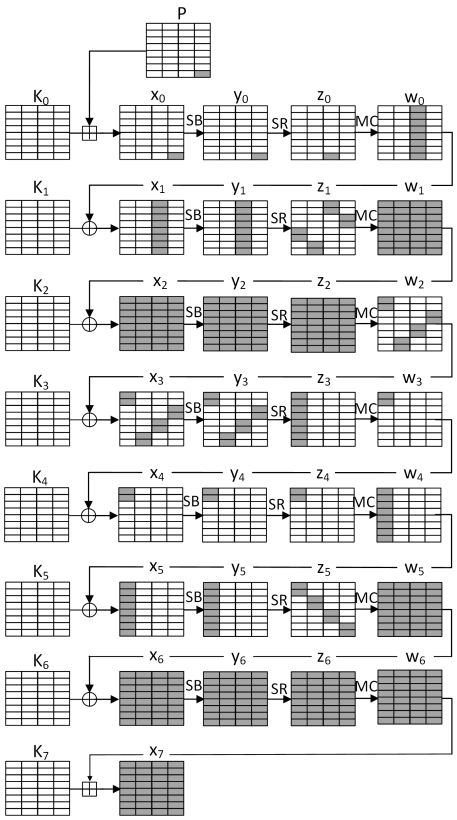

Fig. 2 Differential path used in the attack on the 7-round reduced Kalyna-256/512.

5. Conclusion

We have presented a MitM attack on the new Ukranian standard encryption algorithm Kalyna reduced to seven rounds. According to the security analysis performed by the designers of the cipher, Kalyna is resistant to various cryptanalytic methods after rounds five and six of the 128-bit and 256-bit block versions, respectively. Our results are considered the first steps towards the public cryptanalysis of the new Ukrainian encryption standard.

References

[1] R. Oliynykov, I. Gorbenko, O. Kazymyrov, V. Ruzhentsev, O. Kuznetsov, Y. Gorbenko, O. Dyrda, V. Dolgov, A. Pushkaryov,

R. Mordvinov, and D. Kaidalov, "A new encryption standard of Ukraine: The Kalyna block cipher." Cryptology ePrint Archive, Report 2015/650, 2015. http://eprint.iacr.org/

[2] R. Oliynykov, I. Gorbenko, O. Kazymyrov, V. Ruzhentsev, O. Kuznetsov, Y. Gorbenko, O. Dyrda, V. Dolgov, A. Pushkaryov, R. Mordvinov, and D. Kaidalov, "DSTU 7624:2014. National standard of Ukraine. Information technologies. Cryptographic data security. Symmetric block transformation algorithm. Ministry of economical development and trade of Ukraine," 2015.

[3] R. Oliynykov, I. Gorbenko, O. Kazymyrov, V. Ruzhentsev, Y. Gorbenko, and V. Dolgov, "A new encryption standard of Ukraine: The block cipher "Kalyna" (DSTU 7624:2014)," 2015. Online presntation: http://www.slideshare.net/oliynykov/kalyna-english, accessed: 25-July-2015.

[4] H. Demirci and A. Selçuk, "A meet-in-the-middle attack on 8-round AES," Proc. FSE, ed. K. Nyberg, LNCS, vol.5086, pp.116–126, Springer, 2008.

[5] O. Dunkelman, N. Keller, and A. Shamir, "Improved single-key attacks on 8-round AES-192 and AES-256," Proc. ASIACRYPT, ed. M. Abe, LNCS, vol.6477, pp.158–176, Springer, 2010.

[6] P. Derbez, P.-A. Fouque, and J. Jean, "Improved key recovery attacks on reduced-round AES in the single-key setting," Proc. EUROCRYPT, ed. T. Johansson and P. Nguyen, LNCS, vol.7881, pp.371–387, Springer, 2013.

[7] F. Mendel, C. Rechberger, M. Schläffer, and S.S. Thomsen, "The rebound attack: Cryptanalysis of reduced Whirlpool and Grøstl," Proc. FSE, ed. O. Dunkelman, LNCS, vol.5665, pp.260–276, Springer, 2009.

[8] L. Li, K. Jia, and X. Wang, "Improved single-key attacks on 9-round AES-192/256," Proc. FSE, ed. C. Cid and C. Rechberger, LNCS, vol.8540, pp.127–146, Springer, 2015.

[9] J. Daemen and V. Rijmen, "AES proposal: Rijndael," 1998.

[10] O. Dunkelman, N. Keller, and A. Shamir, "Improved single-key attacks on 8-round AES-192 and AES-256," Journal of Cryptology, vol.28, no.3, pp.397–422, 2015.