Published in Africacrypt 2008, Springer Verlag, LNCS 5023

Improving integral attacks against Rijndael-256 up to 9 rounds

Samuel Galice and Marine Minier

CITI / INSA-Lyon
F-69621 Villeurbanne
samuel.galice,marine.minier@insa-lyon.fr

Abstract. Rijndael is a block cipher designed by V. Rijmen and J. Daemen and it was chosen in its 128-bit block version as AES by the NIST in October 2000. Three key lengths - 128, 192 or 256 bits - are allowed. In the original contribution describing Rijndael [4], two other versions have been described: Rijndael-256 and Rijndael-192 that respectively use plaintext blocks of length 256 bits and 192 bits under the same key lengths and that have been discarded by the NIST. This paper presents an efficient distinguisher between 4 inner rounds of Rijndael-256 and a random permutation of the blocks space, by exploiting the existence of semi-bijective and Integral properties induced by the cipher. We then present three attacks based upon the 4 rounds distinguisher against 7, 8 and 9 rounds versions of Rijndael-256 using the extensions proposed by N. ferguson et al. in [6]. The best cryptanalysis presented here works against 9 rounds of Rijndael-256 under a 192-bit key and requires $2^{128} - 2^{119}$ chosen plaintexts and 2^{188} encryptions.
Keywords: block cipher, cryptanalysis, integral attacks, Rijndael-256.

1 Introduction

Rijndael [4] is an SPN block cipher designed by Vincent Rijmen and Joan Daemen. It has been chosen as the new advanced encryption standard by the NIST [7] with a 128-bit block size and a variable key length k, which can be set to 128, 192 or 256 bits. It is a variant of the Square block cipher, due to the same authors [3]. In its full version, the block length b is also variable and is equal to 128, 192 or 256 bits as detailed in [5] and in [10]. We respectively called those versions Rijndael-b. The recommended Nr number of rounds is determined by b and k, and varies between 10 and 14.

Many cryptanalyses have been proposed against Rijndael for the different block sizes and more particularly against the AES. The first attack against all the versions of Rijndael-b is due to the algorithm designers themselves and is based upon the integral (or saturation) property ([3], [4], [12]) that allows to efficiently distinguish 3 Rijndael inner rounds from a random permutation. This attack has been improved by Ferguson et al. in [6] allowing to cryptanalyse an 8 rounds version of Rijndael-b with a complexity equal to 2^{204} trial encryptions and $2^{128} - 2^{119}$ plaintexts and a 9 rounds version using a related-key attack. In [13], S. Lucks presented an other improvement of the Square

Attack using a particular weakness of the key schedule against a 7 rounds version of Rijndael-b where 2^{194} executions are required for a number of chosen plaintexts equal to 2^{32}. H. Gilbert and M. Minier in [8] also presented an attack against a 7 rounds version of Rijndael-b (known under the name of "Bottleneck Attack") using a stronger property on three inner rounds than the one used in the Square Attack in order to mount an attack against a 7 rounds version of Rijndael requiring 2^{144} cipher executions with 2^{32} chosen plaintexts.

Many other attacks ([2], [14]) have been exhibited against the AES using algebraic techniques exploiting the low algebraic degree of the AES S-box. Other attacks that use related keys and rectangle cryptanalysis have been proposed in [9] and in [11]. But none of these attacks exploits new intrinsic structure of the transformations used in Rijndael-b.

This paper describes an efficient distinguisher between 4 Rijndael-256 inner rounds and a random permutation based upon a particular integral (or saturation) property due to a slow diffusion, presents the resulting 7 rounds attacks on Rijndael-256 which are substantially faster than an exhaustive key search for all the key lengths and the corresponding 8 and 9 rounds extension of the previous attacks for $k = 192$ and $k = 256$.

This paper is organized as follows: Section 2 provides a brief outline of Rijndael-b. Section 3 recalls the original Integral property on three inner rounds, investigates the new four rounds property and describes the resulting distinguisher for 4 inner rounds. Section 4 presents 7, 8 and 9 rounds attacks based on the 4 rounds distinguisher of Section 3. Section 5 concludes this paper.

2 A brief outline of Rijndael-b

Rijndael-b is a symmetric block cipher that uses a parallel and byte-oriented structure. The key length is variable and equal to 128, 192 or 256 bits whereas the block length is equal to 128, 192 or 256 bits. The current block at the input of the round r is represented by a $4 \times (b/32)$ matrix of bytes $A^{(r)}$. We give its representation for $b = 256$:

$$A^{(r)} = \begin{array}{|c|c|c|c|c|c|c|c|} \hline a_{0,0}^{(r)} & a_{0,1}^{(r)} & a_{0,2}^{(r)} & a_{0,3}^{(r)} & a_{0,4}^{(r)} & a_{0,5}^{(r)} & a_{0,6}^{(r)} & a_{0,7}^{(r)} \\ \hline a_{1,0}^{(r)} & a_{1,1}^{(r)} & a_{1,2}^{(r)} & a_{1,3}^{(r)} & a_{1,4}^{(r)} & a_{1,5}^{(r)} & a_{1,6}^{(r)} & a_{1,7}^{(r)} \\ \hline a_{2,0}^{(r)} & a_{2,1}^{(r)} & a_{2,2}^{(r)} & a_{2,3}^{(r)} & a_{2,4}^{(r)} & a_{2,5}^{(r)} & a_{2,6}^{(r)} & a_{2,7}^{(r)} \\ \hline a_{3,0}^{(r)} & a_{3,1}^{(r)} & a_{3,2}^{(r)} & a_{3,3}^{(r)} & a_{3,4}^{(r)} & a_{3,5}^{(r)} & a_{3,6}^{(r)} & a_{3,7}^{(r)} \\ \hline \end{array}$$

The key schedule derives $Nr + 1$ b-bits round keys K_0 to K_{Nr} from the master key K of variable length.

The round function, repeated $Nr - 1$ times, involves four elementary mappings, all linear except the first one:

- SubBytes: a bytewise transformation that applies on each byte of the current block an 8-bit to 8-bit non linear S-box (that we call S) composed of the inversion in the Galois Field $GF(256)$ and of an affine transformation.
- ShiftRows: a linear mapping that rotates on the left all the rows of the current matrix (0 for the first row, 1 for the second, 3 for the third and 4 for the fourth in the case of Rijndael-256 as described in [4]).
- MixColumns: another linear mapping represented by a 4×4 matrix chosen for its good properties of diffusion (see [5]). Each column of the input matrix is multiplied by the MixColumns matrix M in the Galois Field $GF(256)$ that provides the corresponding column of the output matrix. We denote by $M_{i,j}$ for i and j from 0 to 3, the coefficients of the MixColumns matrix.
- AddRoundKey: a simple x-or operation between the current block and the subkey of the round r denoted by K_r. We denote by $K_r^{(i,j)}$ the byte of K_r at position (i, j).

Those $Nr - 1$ rounds are surrounded at the top by an initial key addition with the subkey K_0 and at the bottom by a final transformation composed by a call to the round function where the MixColumns operation is omitted.

3 The integral properties

We describe in this section the three inner rounds property named Integral property explained in the original proposal [4] and the new four rounds property of Rijndael-256.

3.1 The Integral property of Rijndael-*b*

This particular property studied in [12] was first used to attack the Square block cipher [3] and holds for the three size of blocks (128, 192 or 256 bits) of the initial version of Rijndael-b. As previously mentioned, we denote by $A^{(r)}$ the input of the round r.

Let us define the set Λ which contains 256 plaintext blocks (i.e. 256 matrices of bytes of size $4 \times (b/32)$) all distinct. Two blocks belong to the same set Λ if they are equal everywhere except on a particular predefined byte (called the active byte). This active byte takes all possible values between 0 and 255:

$$\forall A^{(1)}, A'^{(1)} \in \Lambda : \begin{cases} a_{i,j}^{(1)} \neq a_{i,j}'^{(1)} \text{ for a given } i \text{ and a given } j \\ a_{i,j}^{(1)} = a_{i,j}'^{(1)} \text{ elsewhere} \end{cases}$$

for $0 \leq i \leq 3$ and $0 \leq j \leq (b/32)$.

The Λ set contains then one active byte whereas the other bytes are passive. Notice that this definition could be generalized to several active bytes as we will see in the next subsections. In all the cases, the transformations SubBytes

and AddRoundKey transform a set Λ into another set Λ with the positions of the active bytes unchanged (see [1] and [12] for more details).

Now, if we look at the semi-bijective properties of the internal transformations of Rijndael-b on three rounds - especially the ones of the ShiftRows and of the MixColumns operations -, we could observe the following results (as shown in figure 1):

- The MixColumns of the first round transforms the active byte of the Λ set into a complete column of active bytes.
- The ShiftRows of the second round diffuses this column on four distinct columns whereas the MixColumns converts this to four columns of only active bytes. This stays a Λ set until the input of MixColumns of the third round.
- Until the input of the MixColumns of the third round, the implied transformations constitute a bijection. Then, since the bytes of this Λ set, range over all possible values and are balanced over this set, we have if we denote by $M^{(3)}$ the active blocks belonging to the Λ set at the input of the third MixColumns operation:

$$\bigoplus_{A^{(4)}=MC(M^{(3)}),M^{(3)}\in\Lambda} a_{k,l}^{(4)} = \bigoplus_{M^{(3)}\in\Lambda} \left(2m_{k,l}^{(3)} \oplus 3m_{k+1,l}^{(3)} \oplus m_{k+2,l}^{(3)} \oplus m_{k+3,l}^{(3)}\right) = 0 \quad (1)$$

where MC represents the MixColumns of the third round and k and l taking all possible values.

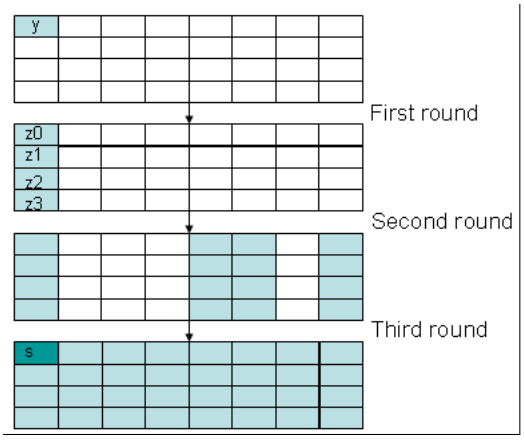

Fig. 1. The three rounds integral property in the case of Rijndael-256: $\bigoplus_{y\in\Lambda} s = 0$

Then, we can easily deduce that each byte at the input of the fourth round is balanced in order to construct an efficient distinguisher between three Rijndael-b inner rounds and a random permutation testing if equality (1) occurs and requiring 256 plaintexts belonging to a same Λ set. Notice also that this property holds for all the possible positions of the active byte.

3.2 An improvement of the saturation property for 4 rounds of Rijndael-256

In the case of Rijndael-256, we describe in this section a particular and stronger property on three rounds of Rijndael-256 and how to extend this property to four rounds for a particular Λ set with three active bytes.

A stronger three rounds property. Suppose now that the same Λ set than the previous one with one active byte, say y at byte position (i, j), is defined. Let us see how this set crosses three Rijndael-256 inner rounds:

- The MixColumns of the first round transforms the active byte of the Λ set into a complete column of active bytes (called $z_0, \cdots, z_3$ in figure 2).
- The ShiftRows of the second round diffuses this column on four among eight distinct columns whereas the MixColumns converts this to four columns among eight of only active bytes.
- The ShiftRows of the third round diffuses those four columns into the eight columns of the current block but the $(j + 2 \mod 8)$-th and the $(j + 6 \mod 8)$-th columns only depend on one byte each, say $a^{(2)}_{i+2 \bmod 4,j}$ and $a^{(2)}_{i+1 \bmod 4,j}$. Using the notations of figure 2, we could say that at the end of the third round, the third and the seventh columns only depend respectively on the byte $a^{(2)}_{2,0} = z_2$ and $a^{(2)}_{1,0} = z_1$; thus, the bytes of those two columns bijectively depend on the y value and each of those two columns represent a Λ set.

So, we have demonstrated that two particular columns stay two different Λ sets at the output of the third round.

How to exploit this property ? Because two complete Λ sets remain for two particular columns at the end of the third round, we want to find a way to exploit this particular property on four rounds of Rijndael-256 by adding one round at the end of the three previous rounds. We always consider here that the input of the four rounds is a Λ set with one active byte y.

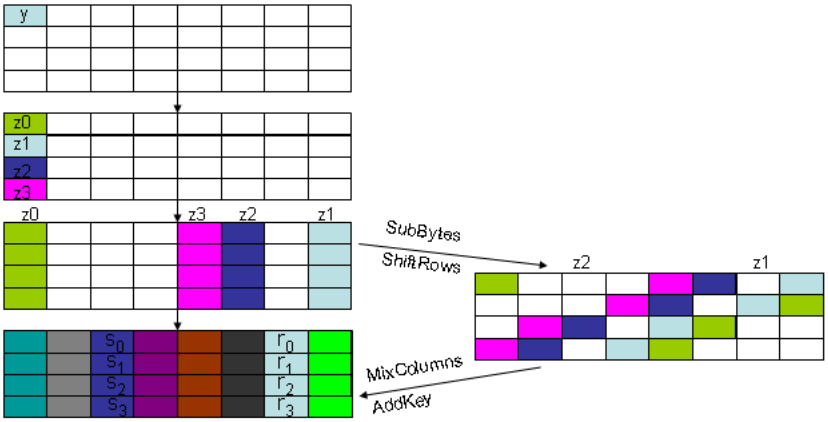

Fig. 2. The three rounds property in the case of Rijndael-256: the bytes $s_0, \cdots, s_3$ only depend on z_2 and the bytes $r_0, \cdots, r_3$ only depend on z_1

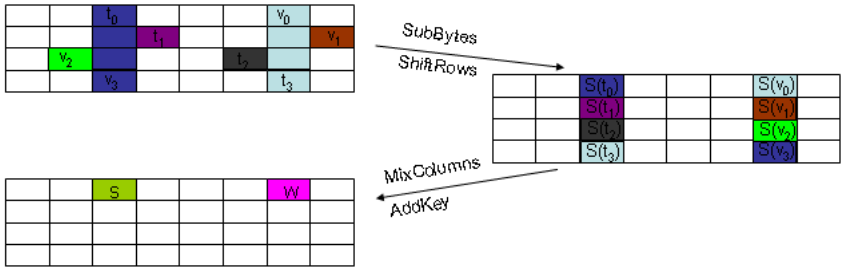

Fig. 3. The fourth round added after the three previous ones

Using the notation of figure 3, we could write the output bytes s and w at the end of the fourth round according to the output bytes of the third round, noticing that t_0, v_3, v_0 and t_3 belongs to two Λ sets:

$$s = 2 \cdot S(t_0) \oplus 3 \cdot S(t_1) \oplus S(t_2) \oplus S(t_3) \oplus K_4^{(0,2)}$$
$$w = 2 \cdot S(v_0) \oplus 3 \cdot S(v_1) \oplus S(v_2) \oplus S(v_3) \oplus K_4^{(0,6)}$$

More formally, we obtain:

$$\begin{aligned} a^{(5)}_{0,j+2 \bmod 8} &= 2S(a^{(4)}_{0,j+2 \bmod 8}) \oplus 3S(a^{(4)}_{1,j+3 \bmod 8}) \\ &\quad \oplus S(a^{(4)}_{2,j+5 \bmod 4}) \oplus S(a^{(4)}_{3,j+6 \bmod 4}) \oplus K_4^{(0,2)} \\ a^{(5)}_{0,j+6 \bmod 8} &= 2S(a^{(4)}_{0,j+6 \bmod 8}) \oplus 3S(a^{(4)}_{1,j+7 \bmod 8}) \\ &\quad \oplus S(a^{(4)}_{2,j+1 \bmod 8}) \oplus S(a^{(4)}_{3,j+2 \bmod 8}) \oplus K_4^{(0,6)} \end{aligned} \quad (2)$$

If we use the notations of figure 3 and if we consider as in the previous subsection that the input of the four rounds is a Λ set with one active byte,

say y, then we have: $\bigoplus_{y\in\Lambda} 2S(t_0) = 0$ and $\bigoplus_{y\in\Lambda} S(t_3) = 0$ because t_0 and t_3 belongs to the same Λ set at the end of the third round. Thus, we could write:

$$\begin{aligned}\bigoplus_{y\in\Lambda} s &= \bigoplus_{y\in\Lambda} (2S(t_0) \oplus 3S(t_1) \oplus S(t_2) \oplus S(t_3)) \\ &= 2\bigoplus_{y\in\Lambda} S(t_0) \oplus 3\bigoplus_{y\in\Lambda} S(t_1) \oplus \bigoplus_{y\in\Lambda} S(t_2) \oplus \bigoplus_{y\in\Lambda} S(t_3) \\ &= 0 \oplus 3\bigoplus_{y\in\Lambda} S(t_1) \oplus \bigoplus_{y\in\Lambda} S(t_2) \oplus 0\end{aligned} \tag{3}$$

The same property holds for w.
So, we want to find a way to obtain

$$3\bigoplus_{y\in\Lambda} S(t_1) \oplus \bigoplus_{y\in\Lambda} S(t_2) = 0 \tag{4}$$

considering that t_1 depends on z_1 and on z_3 and that t_2 depends on z_2 and on z_0. Thus, more input blocks are required to satisfy this equality. A good solution to produce such equality is to take $(256)^2$ Λ sets to completely saturated the values of z_1 and z_3 for t_1 and of z_0 and z_2 for t_2. To produce a such number of plaintexts, let us define the following Λ set with three active bytes - say y, n, p as denoted in figure 4 - at the positions (i, j), $(i + 1 \mod 4, j)$ and $(i + 2 \mod 4, j)$. More formally, we could write this new Λ set as follows:

$$\forall A^{(1)}, A'^{(1)} \in \Lambda : \begin{cases} a^{(1)}_{i,j} \neq a'^{(1)}_{i,j}, a^{(1)}_{i+1 \bmod 4,j} \neq a'^{(1)}_{i+1 \bmod 4,j} \text{and} \\ a^{(1)}_{i+2 \bmod 4,j} \neq a'^{(1)}_{i+2 \bmod 4,j} \\ \text{for a given } i \text{ and a given } j \\ a^{(1)}_{i,j} = a'^{(1)}_{i,j} \text{ elsewhere} \end{cases}$$

Using such a Λ set with 2^{24} elements generated from three different active bytes (say y, n and p) belonging to a same input column, at the end of the fourth round, equality (4) is verified and we then could write using equality (3), $\bigoplus_{y,n,p\in\Lambda} s = 0$ and $\bigoplus_{y,n,p\in\Lambda} w = 0$. More formally, we have:

$$\bigoplus_{y,n,p\in\Lambda} a^{(5)}_{i,j+2 \bmod 8} = 0 \tag{5}$$

$$\bigoplus_{y,n,p\in\Lambda} a^{(5)}_{i,j+6 \bmod 8} = 0 \tag{6}$$

for all $i \in \{0..3\}$.

We performed some computer experiments which confirm the existence of those properties for arbitrarily chosen key values. The complete property is

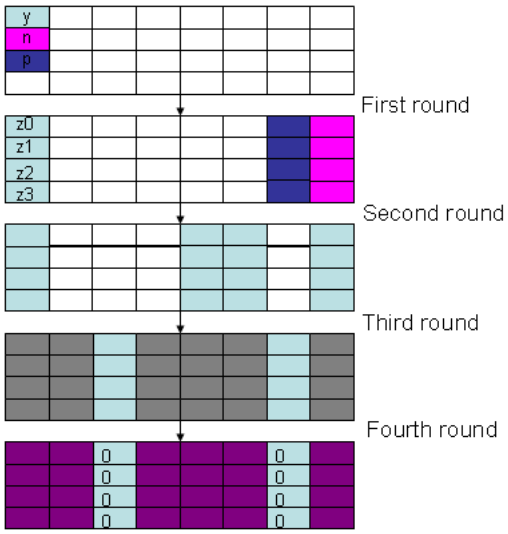

Fig. 4. The four Rijndael-256 rounds property

represented on figure 4. Notice also that when only two bytes - say y and n - at position (i, j) and $(i+1 \mod 4, j)$ are saturated, the corresponding properties are less strong and could be written as partial sums:

$$\bigoplus_{y,n\in\Lambda} a^{(5)}_{0,j+2 \bmod 8} = \bigoplus_{y,n\in\Lambda} a^{(5)}_{3,j+2 \bmod 8}$$
$$\bigoplus_{y,n\in\Lambda} a^{(5)}_{0,j+6 \bmod 8} = \bigoplus_{y,n\in\Lambda} a^{(5)}_{3,j+6 \bmod 8}$$

3.3 The 4 rounds Distinguisher

Then, we can easily use equality (5) or equality (6) at the input of the fifth round in order to construct an efficient distinguisher between four Rijndael-256 inner rounds and a random permutation testing if equality (5) or (6) occurs and requiring 2^{24} plaintexts belonging to a same Λ set with three active bytes at positions (i, j), $(i + 1 \mod 4, j)$ and $(i + 2 \mod 4, j)$.

The existence of such property for Rijndael-256 is not really surprising even if it has never been observed before. This property is due to a slower diffusion in Rijndael-256 than in Rijndael-128 (the AES) and in Rijndael-192. Note also that this particular property doe not work for the AES and Rijndael-192: there is no particular output byte after the third round that only depends on one particular byte of the corresponding input and we do not find such a property for the AES and Rijndael-192.

4 The proposed attacks

We could use the properties previously described on four Rijndael-256 inner rounds to mount elementary attacks against 7, 8 and 9 rounds versions of Rijndael-256. To attack 7 rounds of Rijndael-256, we use first the extension by one round at the beginning proposed in [6] and the partial sums technique described in [6] with equality (5) to add two rounds at the end of our four rounds distinguisher. To extend this 7 rounds attack by one round at the end and/or by one round at the beginning, we directly apply the techniques proposed in [6] and the weakness of the Rijndael key-schedule proposed in [13].

4.1 The 7 rounds attack

Extension at the Beginning. Usually and as done in [4] and in [13], to extend the distinguisher that use equality (1) by one round at the beginning, the authors first choose a set of 2^{32} plaintexts that results in a Λ set at the output of the first round with a single active byte (see figure 5). This set is such that one column of bytes at the input of the first MixColumns range over all possible values and all other bytes are constant. Then, under an assumption of the four well-positioned key bytes of K_0, a set of 256 plaintexts that result in a Λ set at the input of the second round is selected from the 2^{32} available plaintexts.

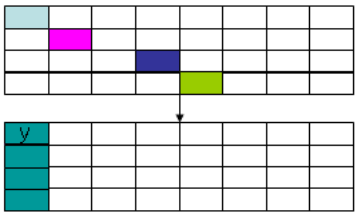

Fig. 5. The extension by one round at the beginning.

Instead of guessing four bytes of the first subkey K_0, the authors of [6] simply use all the 2^{32} plaintexts that represents in our case 2^8 Λ sets with three active bytes (2^8 groups of 2^{24} encryptions that vary only in three bytes of $A^{(1)}$). Then, for some partial guesses of the key bytes at the end of the cipher, do a partial decryption to a single byte of $A^{(5)}$, sum this value over all the 2^{32} encryptions and check for a zero result.

This first improvement save a factor 2^{32} corresponding with 4 exhaustive key bytes of K_0 compared to the attack proposed in [4] using always 2^{32} plaintexts/ciphertexts.

Note also, as done in [10], that we could see this extension as a distinguisher with one more round implying a Λ set with 4 active bytes that results after the first round into an other Λ set with a complete column of active bytes.

Partial sums technique. Using the method of [6], we could use the equality (5) to attack a 7 rounds version of Rijndael-256 by adding one round at the beginning using the technique previously described and adding two rounds at the end using the two rounds extension proposed in [6] and described in figure 6. We describe here the original attack and then directly apply it in our case.

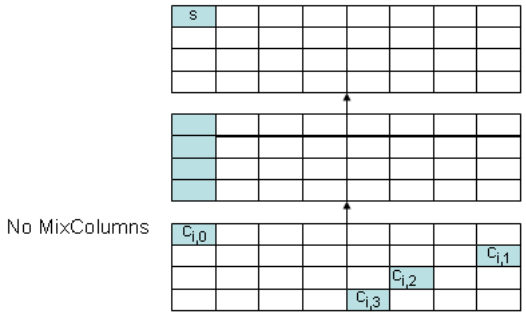

Fig. 6. The extension by two rounds at the end proposed in [6], considering that the last round does not contain a MixColumns operation.

This extension works in the original paper on a 6 rounds version of Rijndael and looks at a particular byte of $A^{(5)}$ that verifies (1) and how it relates to the ciphertext. First, the authors rewrite the cipher slightly by putting the AddRoundKey before the MixColumns in round 5. Instead of applying MixColumns and then adding K_5, they first add in K'_5, which is a linear combination of four bytes of K_5, and then apply MixColumns. Under this assumption, it is easy to see that any byte of $A^{(5)}$ depends on the ciphertext, on four bytes of K_6 and one byte of K'_5 considering that the sixth round is the last one and does not contain a MixColumns operation. Then, only the five key bytes of the two last rounds remain unknowns.

Moreover, the authors improve the complexity of their attack using a technique called "partial sums" to sequentially decipher the two last rounds (the last not containing the MixColumns operation) according the values of the five unknown key bytes. To use the three rounds distinguisher given by equation (1), they compute from the i-th ciphertext c_i:

$$\sum_i S^{-1}\left[S_0\left[c_{i,0} \oplus k_0\right] \oplus S_1\left[c_{i,1} \oplus k_1\right] \oplus S_2\left[c_{i,2} \oplus k_2\right] \oplus S_3\left[c_{i,3} \oplus k_3\right] \oplus k_4\right] \quad (7)$$

where S_0, S_1, S_2, S_3 represent the inverse of the S-box S multiplied by a component of InvMixColumns, $c_{i,j}$ the byte number j of c_i; $k_0, \cdots, k_3$ the four bytes of K_6 and k_4 the implied byte of K'_5.

To improve the general complexity of the attack, they associate the following partial sums to each ciphertext c:

$$x_k := \sum_{j=0}^{k} S_j \left[c_j \oplus k_j\right]$$

for k from 0 to 3. They use the transformation $(c_0, c_1, c_2, c_3) \rightarrow (x_k, c_{k+1}, \cdots, c_3)$ to sequentially determine the different values of k_k and to share the global computation into 4 steps of key bytes search (always testing if equation (1 happens) with 2^{48} operations for each one corresponding with 2^{50} S-box lookups for each set of 2^{32} ciphertexts, corresponding with 2^{24} particular Λ sets of plaintexts with one active byte (see [6] for the details of the complexities). To discard false alarms (i.e. bad keys that pass the test), they need to repeat this process on 6 different Λ sets. Then, the general complexity of the partial sums attacks against a 6 rounds version of Rijndael-b is about 2^{44} encryptions (considering that 2^8 S-box applications are roughly equivalent with one trial encryption) using $6 \cdot 2^{32}$ plaintexts.

The corresponding 7 rounds attacks Applying the first extension at the beginning and the partial sums technique, we could directly mount an attack against 7 rounds of Rijndael-256 using the equality (5) and the corresponding four Rijndael-256 rounds distinguisher: we test if equality (5) holds for $A^{(6)}$ by summing on the 2^{32} values of the partial decryptions corresponding with the 2^{32} plaintexts that represent in our case 2^8 Λ sets with three active bytes. Then, we exploit the partial sums technique on the four corresponding bytes of K_7 and the implied byte of K'_6. For a set of 2^{32} ciphertexts, the cost of the four steps of the deciphering process is exactly the same than in the previous attack and is about 2^{50} S-box lookups. We need to repeat the process using around 6 different sets of 2^{32} ciphertexts to detect false alarms as in [6]. Then, the total number of S-box lookups is 2^{52} corresponding with 2^{44} encryptions, always considering that 2^8 S-box applications is roughly equivalent with one trial encryption.

4.2 The 8 rounds attack

The naive approach. As done in [6] and in [13], we could directly improve the previous 7 rounds attack by adding one round at the end. To express a single byte of $A^{(6)}$ in the key and the ciphertext, we could extend equation (5) to three levels at the end with 16 ciphertexts bytes and 21 key bytes. However, the partial sums technique is only helpful during the last part of the computation.

For a 192-bit master key, we first guess the required 112 bits of the last round 256-bit subkey. The two last bytes of this subkey required for the computations could be directly deduced from the other 112 bits due to the weakness of the key schedule described in [13]: if we know some bytes of the subkey at position i and $i-1$,we directly deduce those at position $i-Nk$ with $Nk=6$ for a 192-bit master key. (Note that this property is only true for some particular positions of the byte $a_{i,j}^{(6)}$, for example if $i=0$ and $j=2$.) Thus after guessing the 14 required bytes of this subkey, we could directly use the partial sums technique requiring about 2^{50} S-box lookups. Thus, the total cost for a structure of 2^{32} ciphertexts is about 2^{162} S-box lookups. As noticed in [6], we need to process three structures of 2^{32} ciphertexts before we start eliminating guesses for the last round key, so the overall cost of this 8-rounds attack is on the order of 2^{164} S-box lookups or about 2^{156} trial encryptions.

For a 256-bit master key, the alignment in the key schedule is different and guessing the eighth round subkey does not give any information about round keys of round 7 and of round 6. So, we could not improve the general complexity of the attack. Working in a similar fashion as before, we first guess the 128 bits of the last round key and using the partial sums technique, compute the four bytes of K_7 and the byte of K_6' for each of the 2^{32} ciphertexts belonging to a same structure. Thus, the complete cost of this attack is about 2^{178} S-box lookups for one structure. As noticed in [6], we need to process five structures of 2^{32} ciphertexts before we start eliminating guesses for the last round key, so the overall cost of this 8-rounds attack is on the order of 2^{180} S-box lookups or about 2^{172} trial encryptions.

The herd technique. In [6], the authors develop a technique to improve their 6 rounds attack by adding one round at the beginning. This new attack require naively the entire codebook of 2^{128} known plaintexts that could be divided into 2^{96} packs of 2^{32} plaintexts/ciphertexts that represent 2^{24} Λ sets with one active byte after this first round. But this property could not be directly exploited because in this case even the wrong keys pass the test at the end of the fifth round since equality (1) holds on for the 2^{120} Λ sets.

Instead, they use a particular byte at the end of the first round, say $a_{a,b}^{(2)}$ different from the four bytes of the Λ set with a fixed value x (see figure 7). With $a_{a,b}^{(2)}=x$, they obtain a set of 2^{120} possible encryptions composed of 2^{88} packs, where each pack contains 2^{24} groups of Λ sets. They call this structure with 2^{120} elements a *herd*. If they sum up equality (1) on a herd, then the property is only preserved for the correct key.

Thus, they notice that this particular byte $a_{a,b}^{(2)}$ depends on only four bytes of plaintext, say $(p_4,\cdots,p_7)$ and on four bytes of the key K_0. As done for the partial sums technique, they could share the key exhaustive search on the four

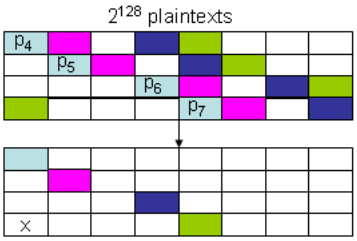

Fig. 7. The herd technique: adding one more round at the beginning.

key bytes of K_0 required to entirely determine the value of $a_{a,b}^{(1)}$ in a three-phase attack using 2^{64} counters m_y for the first phase, 2^{32} counters n_z for the second whereas the third phase filters information for key guesses.

The attack works as follows: in the first phase, the counter m_y is incremented at bit level according the 64-bit value $y = (c_0, \cdots, c_3, p_4, \cdots, p_7)$; in the second phase, the four bytes of K_0 are guessed to compute $a_{a,b}^{(2)}$ and to share the counters into herds; then select a single herd and update n_z by adding $z = (c_0, \cdots, c_3)$ for each y that is in the good herd; in the third phase, guess the five key bytes of K_7 and of K_6' to decrypt each z to a single byte of $A^{(6)}$, sum this byte over all the 2^{32} values of z (with multiplicities) and check for zero. This last phase must be repeated for each initial guess of the four bytes of K_0.

The first phase requires about 2^{120} trial encryptions and the rest of the attack has a negligible complexity compared to it (see [6] for some details about the attack complexity). Then, the total complexity of this attack is 2^{120} trial encryptions and 2^{64} bits of memory using 2^{128} chosen plaintexts. The authors provide another improvement of their attack remarking that the four plaintext bytes $(p_4, \cdots, p_7)$ and the four guessed key bytes of K_0 define four bytes of $A^{(1)}$. So they can create 2^{24} smaller herds with 2^{104} elements by fixing three more bytes of $A^{(1)}$ to reduce the plaintext requirements to $2^{128} - 2^{119}$ texts.

So, we could directly apply this attack to an 8 rounds version of Rijndael-256 using the particular equality (5) by adding two rounds at the beginning and two rounds at the end using $2^{128} - 2^{119}$ plaintexts that will be separated into herds during the second phase of the attack. However, in the previous case, they consider that all the codebook is known due to the huge amount of plaintexts required. So, they do not take into account the ciphering process. This is not our case and we first need to cipher $2^{128} - 2^{119}$ chosen plaintexts among the 2^{256} possible values with four active columns that lead to 2^{24} herds with 2^{104} elements at the end of the first round. Then, the complexity of the attack itself is the same but the total cost is dominated by the $2^{128} - 2^{119}$ trial encryptions. Notice that the same problem remains for Rijndael-192.

4.3 The 9 rounds attack

As done in [6], we could use the herd technique (with 2^{23} undamaged herds) combined with the partial sums technique to mount an attack against a 9 rounds version of Rijndael-256. In this case, we guess four bytes of K_0 and the 21 subkey bytes - 16 bytes of K_9, 4 bytes of K_8 and one byte of K'_7 - required to add three rounds at the end of the 4 rounds distinguisher. We always consider that this distinguisher is extended with one round at the beginning summing on sets with 2^{32} elements. The attack then works as follows: first, construct 2^{23} undamaged herds of 2^{104} elements using $2^{128} - 2^{119}$ plaintexts; guess the four key bytes of K_0 to determine a particular herd; then apply the partial sums technique to this set to compute each x_k and to obtain a single byte of $A^{(7)}$ depending on 16 bytes of the ciphertext and 21 subkey bytes; then use the fact that summing the 2^{104} values on a single byte of $A^{(7)}$ will yield zero (from equality (5)) for the good key. The required storage is about 2^{104} bits and the total complexity of this attack is about $2^{32} \cdot 2^{170} = 2^{202}$ trial encryptions for one herd and a 256-bit key (see [6] for the details of the complexity of the attack). We need to test four herds before discarding the first bad keys and at least 26 herds to get exactly the good key (with a decreasing complexity). Then, the total complexity of this attack is about 2^{204} trial encryptions.

This attack could only work for a 256-bit key. However, in the case of a 192-bit key, using the weakness of the key-schedule described in section 4.2, we know that we could preserve 2 bytes of the exhaustive search of K_9 that are directly determined by the 14 others. Then, we could save a 2^{16} factor from the previous attack and we obtain a complexity of about $2^{204-16} = 2^{188}$ trial encryptions for the same number of plaintexts and the same required storage.

5 Conclusion

In this paper, we have presented a new particular property on four rounds of Rijndael-256 that relies on semi-bijective properties of internal foldings. Then we have built the best known attack against a 9 rounds version of Rijndael-256 requiring for a 192-bit keys 2^{188} trial encryptions with $2^{128} - 2^{119}$ plaintexts. We have summed up in table 1 all known results concerning the attacks against Rijndael-b.

In [6], the authors also present a related key attack against a 9 rounds version of the AES. Moreover, in [9] and in [11], two related key rectangle attacks have been proposed against the AES under keys of length 192 and 256 bits. We do not find a way to extend the attacks that use related keys against Rijndael-256. The main problem in this case comes from the higher number of 32-bit key words that must be generated to construct 256-bit subkeys: we do not find a key pattern that sufficiently preserves an integral property.

Cipher	nb rounds	Key size	Data	Time Complexity	source
AES	6	(all)	2^{32} CP	2^{72}	[4] (Integral)
	7	(all)	$2^{128} - 2^{119}$ CP	2^{120}	[6] (Part. Sum)
	8	(192)	$2^{128} - 2^{119}$ CP	2^{188}	[6] (Part. Sum)
	8	(256)	$2^{128} - 2^{119}$ CP	2^{204}	[6] (Part. Sum)
	9	(256)	2^{85} RK-CP	2^{224}	[6] (Related-key)
	10	(192)	2^{125} RK-CP	2^{182}	[11] (Rectangle)
Rijndael-192	6	(all)	2^{32} CP	2^{72}	[4] (Integral)
	7	(all)	$2^{128} - 2^{119}$ CP	$2^{128} - 2^{119}$	[6] (Part. Sum)
	8	(192)	$2^{128} - 2^{119}$ CP	2^{188}	[6] (Part. Sum)
	8	(256)	$2^{128} - 2^{119}$ CP	2^{204}	[6] (Part. Sum)
Rijndael-256	6	(all)	2^{32} CP	2^{72}	[4] (Integral)
	7	(all)	$2^{128} - 2^{119}$ CP	$2^{128} - 2^{119}$	[6] (Part. Sum)
	7	(all)	6×2^{32} CP	2^{44}	this paper
	8	(all)	$2^{128} - 2^{119}$ CP	$2^{128} - 2^{119}$	this paper
	9	(192)	$2^{128} - 2^{119}$ CP	2^{188}	this paper
	9	(256)	$2^{128} - 2^{119}$ CP	2^{204}	this paper

Table 1. Summary of Attacks on Rijndael-b - CP: Chosen plaintexts, RK: Related-key

References

1. Alex Biryukov and Adi Shamir. Structural cryptanalysis of SASAS. In Birgit Pfitzmann, editor, *EUROCRYPT*, volume 2045 of *Lecture Notes in Computer Science*, pages 394–405. Springer, 2001.
2. Nicolas Courtois and Josef Pieprzyk. Cryptanalysis of block ciphers with overdefined systems of equations. In *ASIACRYPT*, volume 2501 of *Lecture Notes in Computer Science*, pages 267–287, 2002.
3. J. Daemen, L.R. Knudsen, and V. Rijmen. The block cipher Square. In *Fast Software Encryption'97*, Haifa, Israël, pages 149–165. Lectures Notes in Computer Science 1267, Springer-Verlag, 1997.
4. J. Daemen and V. Rijmen. AES proposal: Rijndael. In *The First Advanced Encryption Standard Candidate Conference*. N.I.S.T., 1998.
5. J. Daemen and V. Rijmen. *The Design of Rijndael*. Springer-Verlag, 2002.
6. Niels Ferguson, John Kelsey, Stefan Lucks, Bruce Schneier, Michael Stay, David Wagner, and Doug Whiting. Improved cryptanalysis of Rijndael. In Bruce Schneier, editor, *FSE*, volume 1978 of *Lecture Notes in Computer Science*, pages 213–230. Springer, 2000.
7. FIPS 197. Advanced Encryption Standard. Federal Information Processing Standards Publication 197, 2001. U.S. Department of Commerce/N.I.S.T.
8. Henri Gilbert and Marine Minier. A collision attack on 7 rounds of Rijndael. In *AES Candidate Conference*, pages 230–241, 2000.
9. Seokhie Hong, Jongsung Kim, Sangjin Lee, and Bart Preneel. Related-key rectangle attacks on reduced versions of SHACAL-1 and AES-192. In Henri Gilbert and Helena Handschuh, editors, *FSE*, volume 3557 of *Lecture Notes in Computer Science*, pages 368–383. Springer, 2005.
10. Jorge Nakahara Jr., Daniel Santana de Freitas, and Raphael Chung-Wei Phan. New multiset attacks on Rijndael with large blocks. In Ed Dawson and Serge Vaudenay, editors, *Mycrypt*, volume 3715 of *Lecture Notes in Computer Science*, pages 277–295. Springer, 2005.

11. Jongsung Kim, Seokhie Hong, and Bart Preneel. Related-key rectangle attacks on reduced AES-192 and AES-256. In Alex Biryukov, editor, *FSE*, volume 4593 of *Lecture Notes in Computer Science*, pages 225–241. Springer, 2007.
12. Lars R. Knudsen and David Wagner. Integral cryptanalysis. In Joan Daemen and Vincent Rijmen, editors, *FSE*, volume 2365 of *Lecture Notes in Computer Science*, pages 112–127. Springer, 2002.
13. Stefan Lucks. Attacking seven rounds of Rijndael under 192-bit and 256-bit keys. In *AES Candidate Conference*, pages 215–229, 2000.
14. Sean Murphy and Matthew J. B. Robshaw. Essential algebraic structure within the AES. In *CRYPTO*, volume 2442 of *Lecture Notes in Computer Science*, pages 1–16, 2002.