

A collisions attack on the 7-rounds Rijndael

Henri Gilbert and Marine Minier

Centre National d'Etudes des T l communications
38-40, rue du G n ral Leclerc
92794 Issy les Moulineaux Cedex 9 - France
email : henri.gilbert@cnet.francetelecom.fr

Abstract

Rijndael is one of the five candidate blockciphers selected by NIST for the final phase of the AES selection process. The best attack of Rijndael so far is due to the algorithm designers. This attack is based upon the existence of an efficient distinguisher between 3 Rijndael inner rounds and a random permutation, and it is limited to 6 rounds for each of the three possible values of the keysize parameter (128 bits, 196 bits and 256 bits). In this paper, we construct an efficient distinguisher between 4 inner rounds of Rijndael and a random permutation of the blocks space, by exploiting the existence of collisions between some partial functions induced by the cipher. We present an attack based upon this 4-rounds distinguisher that requires 2^{32} chosen plaintexts and is applicable to up to 7-rounds for the 196 keybits and 256 keybits version of Rijndael. Since the minimal number of rounds in the Rijndael parameter settings proposed for AES is 10, our attack does not endanger the security of the cipher, indicate any flaw in the design or prove any inadequacy in selection of number of rounds. The only claim we make is that our results represent improvements of the previously known cryptanalytic results on Rijndael.

1 Introduction

Rijndael [DaRi98], a blockcipher designed by Vincent Rijmen and Joan Daemen, is one of the 5 finalists selected by NIST in the Advanced Encryption Standard competition [AES99]. It is a variant of the Square blockcipher, due to the same authors [DaKnRi97]. It has a variable block length b and a variable key length k , which can be set to 128, 192 or 256 bits. The recommended nr number of rounds is determined by b and k , and varies between 10 and 14. In the sequel we will sometimes use the notation Rijndael/ $b/k/nr$ to refer to the Rijndael variant determined by a particular choice of the b , k and nr parameters.

The best Rijndael attack published so far is due to the algorithm designers [DaRi98]. It is a variant of a the "Square" attack, and exploits the byte-oriented structure of Rijndael [DaKnRi97]. This attack is based upon an efficient distinguisher between 3 Rijndael inner rounds and a random permutation. It is stated in [DaRi98] that "for the different block lengths of Rijndael no extensions to 7 rounds faster than exhaustive search have been found".

In this paper we describe an efficient distinguisher between 4 Rijndael inner rounds and a random permutation, and we present resulting 7-rounds attacks of Rijndael/ $b=128$ based on which are substantially faster than an exhaustive key search for the $k = 196$ bits and $k = 256$ bits versions and marginally faster than an exhaustive key search for the $k = 128$ bits version.

This paper is organised as follows. Section 2 provides an outline of the cipher. Section 3 investigates "partial functions" induced by the cipher and describes a distinguisher for 4 inner rounds. Section 4 presents 7-rounds attacks based on the 4-rounds distinguisher of Section 3. Section 5 concludes the paper.

2 An outline of Rijndael/ $b = 128$

In this Section we briefly described the Rijndael algorithm. We restrict our description to the $b=128$ bits blocksize and will consider no other blocksize in the rest of this paper.

The algorithm consists of a key schedule and an iterated encryption function with nr rounds. The key schedule derives $nr + 1$ 128-bit round keys K_0 to K_{nr} from the $k = 128, 196$ or 256 bits long Rijndael key K . Since attacks presented in the sequel do not use the details of the dependence between round keys, we do not provide a description of the key schedule.

The encryption function is the composition of numerous elementary block permutations. The current 128-bit block value B is represented by a 4×4 matrix :

$$B = \begin{array}{|c|c|c|c|} \hline b_{0,0} & b_{0,1} & b_{0,2} & b_{0,3} \\ \hline b_{1,0} & b_{1,1} & b_{1,2} & b_{1,3} \\ \hline b_{2,0} & b_{2,1} & b_{2,2} & b_{2,3} \\ \hline b_{3,0} & b_{3,1} & b_{3,2} & b_{3,3} \\ \hline \end{array}$$

Rijndael uses four elementary mappings :

- the σ =ByteSub byte substitution transforms each of the 16 input bytes under a fixed byte permutation P (the Rijndael S-box).
- the ρ =ShiftRow rows shift circularly shifts row i ($i = 0$ to 3) in the B matrix by i bytes to the right.
- the μ =MixColumn is a matrix multiplication by a fixed $GF(2^8)$ matrix
- the κ_r =KeyAddition is a bitwise addition with a 128-bit round key K_r .

The Rijndael cipher is composed by an initial round key addition κ_0 , $nr - 1$ inner rounds and a final transformation. The r th inner round ($1 \leq r \leq nr - 1$) is defined as the $\kappa_r \circ \mu \circ \rho \circ \sigma$ function. The final transformation at the round nr is an inner round without Mixcolumn mapping : FinalRound = $\kappa_{nr} \circ \rho \circ \sigma$. We can thus summarise the cipher as follows:

```

 $B := \kappa_0(B);$ 
For  $r = 1$  to  $nr - 1$ 
 $B := \text{InnerRound}(B);$ 
FinalRound( $B$ );

```

Remarks :

- σ is the single non $GF(8)$ -linear function of the whole cipher.
- The Rijndael S-box P is the composition of the multiplicative inverse function in $GF(8)$ (NB : '00' is mapped into itself) and a fixed $GF(2)$ -affine byte transformation. If the affine part of P was omitted, algebraic methods (e.g. interpolation attacks) could probably be considered for the cryptanalysis of Rijndael.
- The $\mu \circ \rho$ linear part of Rijndael appears to have been carefully designed. It achieves a full diffusion after 2 rounds, and the Maximum Distance Separability (MDS) property of μ prevents good differential or linear "characteristics" since it ensures that two consecutive rounds involve many active S-boxes.

3 Distinguishing 4 inner rounds of Rijndael/ $b=128$ from a random permutation

3.1 Notation

Figure 1 represents 4 consecutive inner round functions of Rijndael associated with any 4 fixed unknown 128-round keys. Y, Z, R, S represent the input blocks of the 4 rounds and T represents the output of the 4th round. We introduce short notations for some particular bytes of Y, Z, R, S, T , which play a particular role in the sequel : $y = Y_{0,0}$, $z_0 = Z_{0,0}$, $z_1 = Z_{1,0}$, $z_2 = Z_{2,0}$, $z_3 = Z_{3,0}$, and so on. Finally we denote by c the $(c_0 = Y_{1,0}, c_1 = Y_{2,0}, c_2 = Y_{3,0})$ triplet of Y bytes.

Let us fix all the Y bytes but y to any 11-uple of constant values. So the c triplet is assumed to be equal to a constant (c_0, c_1, c_2) triplet, and the 12 $Y_{i,j}$, $i=1$ to 3, $j=0$ to 3 are also assumed to be constant. The 4 Rijndael rounds represented on Figure induce one byte to one byte functions that associate each of the Z, R, S, T bytes introduced above with the y input byte. Thus we denote by $z_0^c[y]$ to $z_3^c[y]$, $r_0^c[y]$ to $r_3^c[y]$, $s^c[y]$, $t_0^c[y]$ to $t_3^c[y]$, the z_i, r_i, s, t_i byte value associated with a c constant and one $y \in 0..255$ value.

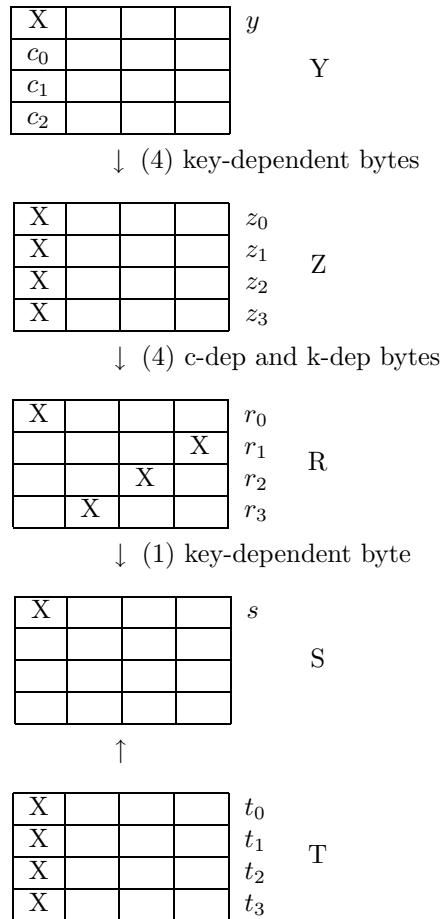


Figure 1: 4 inner rounds of Rijndael

3.2 The 3-rounds distinguisher used in the Rijndael/b=128 designers' attack

The Rijndael designers' attack is based upon the observations that : - z_0 to z_3 are one to one functions of y and the other Z bytes are constant - r_0 to r_3 bytes are one to one functions of y (as well as the 12 other R bytes) - s is the XOR of four one to one functions of y and thus $\sum_{y=0}^{255} s[y] = 0$.

Thus 3 consecutive inner rounds of Rijndael have the distinguishing property that if all Y bytes but y are fixed and y is taken equal to each of the 256 possible values, then the sum of the 256 resulting s values is equal to zero.

This leads to a 6-rounds attack (initial key addition followed by 5 inner rounds followed by final round). Under an assumption on 4 initial key addition key bytes, an initial round (initial key addition followed by 1 inner round can be added on top) and two additional rounds can be added at the end (namely one additional inner round followed by one final round). Under an assumption on 5 additional keybytes (4 final round key bytes and one combination of the additional inner round bytes), this attack need 2^{32} plaintext with a complexity equal to 2^{80} encryption.

3.3 A 4-rounds distinguisher for Rijndael/b = 128

We now analyse in detail the dependency of the one byte to one byte functions introduced in Section 3.1 in the c constant and the expanded key. We show that the $s^c[y]$ function is entirely determined by a surprisingly small number of unknown bytes, which either only depend upon the key or depend upon both the key and the c value. This provides an efficient test for distinguishing 4 inner rounds of Rijndael from a random permutation.

The construction of the proposed distinguisher is based upon the following observations, which are illustrated in Figure 1.

Property 1 : At round 1, the $y \rightarrow z_0^c[y]$ one to one function is independent of the value of the c triplet and is entirely determined by one key byte. The same property holds for z_1, z_2, z_3 . This is because inside at the output of the ShiftRow function inside the first inner round the c_0 to c_2 constants only affect columns 1 to 3 of the current block value, whereas the z_0 to z_3 bytes entirely depend upon column 0.

Property 2 : At round 2, the $r_0[y]$ value is one to one function of $z_0[y]$, and the $r_0[y] \rightarrow z_0[y]$ is entirely determined by one single unknown constant byte that is entirely determined by c and the second inner round key . For similar reasons, the one to one relation between $r_2[y]$ and $z_2[y]$ (resp. $r_3[y]$ and $z_3[y]$) is entirely determined by one single unknown constant byte. Consequently, the $r_0[y]$ to $r_3[y]$ are one to one functions of y and these four functions are entirely determined by the 4 c -independent and key-dependent constant unknown bytes introduced in property (1) and the 4 c - and k -dependent bytes.

Property 3 : At round 3, the s byte can be expressed as a function of the r_0 to r_3 bytes and one c -independent and key-dependent unknown constant. Consequently, the $s^c[y]$ function is entirely determined by 5 c -ind and key-dep constant and 4 key-dep and c -dep constants

Property 4 : Let us consider the decryption of the fourth inner round : s can be expressed as $s = p^{-1}[(0E.t_0 + 0B.t_1 + 0D.t_2 + 09.t_3) + k_5]$ where p represents the single S-box. In other words $0E.t_0 + 0B.t_1 + 0D.t_2 + 09.t_3$ is a one to one function of s , and that function is entirely determined by 1 single key byte k_5 . Thus $0E.t_0 + 0B.t_1 + 0D.t_2 + 09.t_3$ is a function of y that is entirely determined by 6 unknown bytes which only depend upon the key and by 4 additional unknown bytes which depend both upon c and the key.

Properties (3) and (4) provides an efficient 4-rounds distinguisher. We can restate property (4) in saying that the $s^c[y]$ function is entirely determined (in a key-dependent manner) by 4 c -dependent bytes. Let us make the heuristic assumption that these 4 unknown c -dependent bytes behave as a random function of the c triplet of bytes. By the birthday paradox, given a C set of about $2^{16}c$ triplet values, there exist with a non negligible probability two distinct c' and c'' in C such that the $s^{c'}[y]$ and $s^{c''}[y]$ functions induced by c' and c'' are equal (i.e. in other words such that the $(s^{c'}[y])_{y=0..255}$ and $(s^{c''}[y])_{y=0..255}$ lists of 256 bytes are equal). Property (4) provides a method to test such a "collision", using the t_0 to t_3 output bytes of 4 inner rounds : c' and c'' collide if and only if $\forall y \in [0, \dots, 255]$, $0E.t_0^{c'} + 0B.t_1^{c'} + 0D.t_2^{c'} + 09.t_3^{c'} = 0E.t_0^{c''} + 0B.t_1^{c''} + 0D.t_2^{c''} + 09.t_3^{c''}$. Note that it is sufficient to test the above equality on a limited number of y values (say 16 for instance) to know with a quite negligible "false alarms" probability whether c' and c'' collide.

The proposed 4-round distinguisher uses the above collision test is the following manner :

- select a C set of about $2^{16}c$ triplet values and a subset of $0..255$, say for instance a Λ subset of 16 y values.
- for each c triplet value, compute the $L_c = (0E.t_0^c + 0B.t_1^c + 0D.t_2^c + 09.t_3^c)_{y \in \Lambda}$. We claim that such a computation of 16 linear combinations of the outputs represents substantially less than one single Rijndael operation.
- check whether two of the above lists, $L_{c'}$ and $L_{c''}$ are equal. The 4 round distinguisher requires about 2^{20} chosen inputs Y , and since the collision detection computations (based on the analysis of the corresponding T values) require less operations than the 2^{20} 4-inner rounds computations, the complexity of the distinguisher is less than 2^{20} Rijndael encryptions.

Note that property (4) also provides a method to distinguish 4 inner round from a random permutation, using $N \leq 256$ plaintexts and $2^{80} N$ operations (note that a value such as $N = 16$ is far sufficient in practice). However, the birthday paradox provides another, more efficient, distinguisher.

4 An attack of the 7-rounds Rijndael/ $b=128$ cipher with 2^{32} chosen plaintexts

In this Section we show that the 4-inner round distinguisher of Section 3 provides attacks of the 7-rounds Rijndael for the $b=128$ blocksize and the various keysizes. We present two slightly different attacks. The first one (cf Section 4.2 hereafter) is substantially faster than an exhaustive search for the $k=196$ and $k=256$ keysizes, but slower than exhaustive search of the $k=128$ bits keysizes. The second attack (cf Section 4.2) is dedicated to the $k=128$ keysizes, and is marginally faster than an exhaustive search for that keysizes.

The 7-rounds Rijndael is depicted at Figure 2. X represents a plaintext block, and V represents a cipher block. In Figure 2 the 4 inner rounds of Figure 1 are surrounded by one initial $X \rightarrow Y$ round (which consists of an initial key addition followed by one first inner round), and two final rounds (which consists of one $T \rightarrow U$ inner round followed by an $U \rightarrow V$ final round).

Our attack method is basically a combination of the 4-round distinguisher presented in Section 3 and an exhaustive search of some keybytes (or combinations of keybytes) of the initial and the two final rounds. In the attack of Section 4.1 we are using the property that in the equations provided by the 4-rounds distinguisher there is a variables separations in terms which involve one half of the 2 last rounds keybytes and terms which involve a second half of the 2 last round key bytes to save a 2^{80} factor in the exhaustive search complexity. In the attack of Section 4.2, we are using precomputations on colliding pairs of c values to test each 128-bits key assumption with less operations than one single Rijndael encryption.

4.1 An attack of the 7-rounds Rijndael/ $b=128/k=196$ or 256 with 2^{32} chosen plaintexts and a complexity of about 2^{140}

We now explain the attack procedure in some details, using the notation introduced in Figure 2. We fix all X bytes but the four bytes x_0 to x_3 equal to 12 arbitrary constant values. We encrypt the 2^{32} plaintexts obtained by taking all possible values for the x_0 to x_3 bytes, thus obtaining 2^{32} V ciphertext blocks. We are using the two following observations :

Property 5 : If 4 keybytes added with the x_0 to x_3 bytes in the initial key addition are known (let us denote them by $k_{ini} = (k_0, k_1, k_2, k_3)$, then it is possible to partition the 2^{32} plaintexts in 2^{24} subsets of 256 plaintext values satisfying the conditions of Section 3, i.e. such that the corresponding 256 Y values satisfy the following conditions :

- the y byte takes 256 distinct values (that are known up to an unknown constant first round key byte which is not required for the attack).
- the $c = (c_0, c_1, c_2)$ triplet of bytes is constant ; moreover, each of the 2^{24} subsets corresponds to a distinct c value (the c value corresponding to

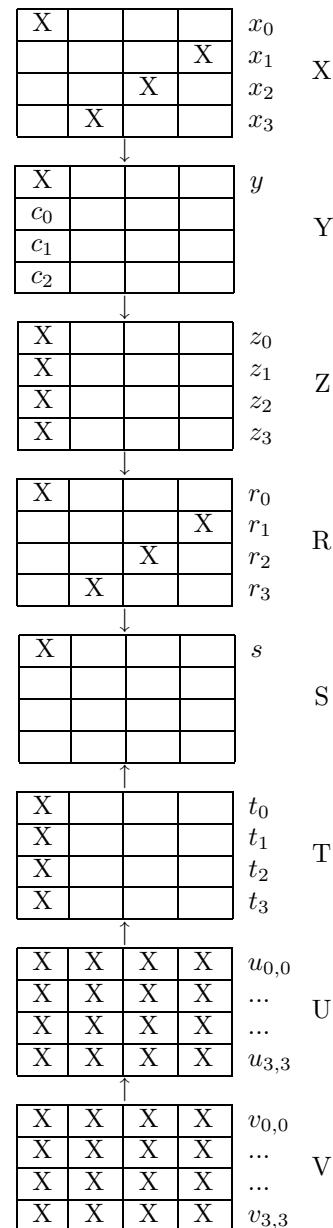


Figure 2: 7-rounds Rijndael

each subset is known up to three constant first round keybytes which are not required for the attack).

- the 12 other Y bytes are constant and their constant values $Y_{i,j}$ for $i=1..3$ and $j=0..3$ is the same for all subsets.

Note that the same property is used in the Rijndael designers' attack.

Property 6 : Each of the t_0, t_1, t_2, t_3 bytes can be expressed as a function of four ciphertext bytes and five unknown key bytes (i.e. 4 of the final round key bytes and one linear combination of the penultimate round key bytes). Therefore, we can "split" the $t^c[y] = 0E'.t_0^c[y] + 0B'.t_1^c[y] + 0D'.t_2^c[y] + 09'.t_3^c[y]$ combination of the four $t_i^c[y]$ bytes considered in the 4-rounds distinguisher as the XOR of two terms $\tau_1^c[y]$ and $\tau_2^c[y]$ which can each be expressed as a function of 8 ciphertext bytes and 10 unknown key bytes, namely $\tau_1 = 0E'.t_0^c[y] + 0B'.t_1^c[y]$ and $\tau_2 = 0D'.t_2^c[y] + 09'.t_3^c[y]$. We denote in the sequel by k_{τ_1} those 10 unknown keybytes which allow to derive τ_1 from 8 bytes of the V ciphertext, and by k_{τ_2} those 10 keybytes that allow to derive τ_2 from 8 bytes of the V ciphertext.

We perform an efficient exhaustive search of the k_{ini} , k_{τ_1} and k_{τ_2} keys in the following way :

- For each of the 2^{32} possible k_{ini} assumptions, we can select 256^2 subsets (of 256 X plaintexts each) associated with one distinct c constant each. Note that the c value associated with a subset and the y values associated with each of the X plaintexts of a subset are only known up to unknown keybits, but this does not matter for our attack. We can denote by c^* and y^* the known values which only differ from the actual values by fixed unknown key bits.
- Now for each subset associated with a c^* constant triplet, based on the say the 16 ciphertexts associated with the $y^* = 0$ to $y^* = 15$ values, we can precompute the $(\tau_1^c(y^*))_{y^*=0..15}$ 16-tuple of bytes for each of the 2^{80} possible k_{τ_1} keys. We can also precompute the $(\tau_2^c(y^*))_{y^*=0..15}$ 16-tuple for each of the 2^{80} possible k_{τ_2} keys.

Based on this precomputation, for each (c^*, c''^*) pair of distinct c^* values :

- We precompute a (sorted) table the $(\tau_1^{c'}(y^*) \oplus \tau_1^{c''}(y^*))_{y^*=0..15}$ 16-tuple of bytes for each of the 2^{80} possible k_{τ_1} keys (the computation of each 16-tuple just consists in xoring two precomputed values)
- For each of the 2^{80} possible values of the k_{τ_2} key, we compute the $(\tau_2^{c'}(y) \oplus \tau_2^{c''}(y))_{y=0..15}$ 16-tuple of bytes associated with the observed ciphertext, and check whether this t-uple belongs to the precomputed table of 16-tuple $(\tau_1^{c'}(y^*) \oplus \tau_1^{c''}(y^*))_{y^*=0..15}$. If for a given k_{τ_1} value there exists a k_{τ_2} value such that $(\tau_1^{c'}(y^*) \oplus \tau_1^{c''}(y^*))_{y^*=0..15} = (\tau_2^{c'}(y) \oplus \tau_2^{c''}(y))_{y=0..15}$, (i.e. $t^{c'}[y] = t^{c''}[y]$ for each of the y values associated with $y^* = 0$ to 16 , this represents an alarm). The equality between the t bytes associated with c'

and c'' is checked for the other $y*$ values. If the equality is confirmed, this means that a collision between the $s^c[y]$ functions associated with c' and c'' . This provides 20 bytes of information on the last and penultimate key values, since with overwhelming probability, the right values of k_{ini} , k_{τ_1} and k_{τ_2} have then been found.

Since a random subset 256^2 of the 256^4 possible $s^c[y]$ functions has been tested, the probability of the above procedure to result in a collision, and thus to provide k_{ini} , k_{τ_1} and k_{τ_2} is high (say about $1/2$). In other words, the success probability of the attack is about $1/2$.

Once k_{ini} , k_{τ_1} and k_{τ_2} have been found, the 16-bytes final round key is entirely determined and the final round can be decrypted, so one is left with the problem of cryptanalysing the 6-round version of Rijndael. One might object that the last round of the left 6-round cipher is not a final round, but an inner round. However, it is easy to see that by applying a linear one to one change of variable to U and the 6th round key (i.e. replacing U by a U' linear function of U and K_6 by a linear function K' of K), the last round can be represented as a final round (i.e. U' is the image of T by the final round associated with K'). Thus we have in fact left with the cryptanalysis of the 6-round Rijndael, and the last round subkey is easy to derive. The process of deriving the subkeys of the various rounds can then be continued (using a subset of the 2^{32} chosen plaintexts used for the derivation of k_{ini} , k_{τ_1} and k_{τ_2}), with negligible additional complexity, until the entire key has eventually been recovered.

4.2 An attack of the 7-rounds Rijndael/ $b=128/k=128$ 2^{32} chosen plaintext

We now outline a variant of the former attack that is dedicated to the $k=128$ bits version of Rijndael and is marginally faster than an exhaustive search. This attack requires a large amount of precomputations.

As a matter of fact, it can be shown that the 4 c -dependent bytes that determine the mapping between four $z_i^c[y]$ bytes and the four $r_i^c[y]$ are entirely determined by 12 key-dependent (and c -independent) bytes. For each of the 256^{12} possible values of this $\phi(K)$ 12-tuple of bytes, we can compute colliding c' and c'' triplets of bytes (this can be done performing about 256^2 partial Rijndael computations corresponding to say 256^2 distinct c values and looking for a collision. One can accept that no collision be found for some $\phi(K)$ values : this just means that the attack will fail for a certain fraction (say $1/2$) of the key values). We store c' and c'' (if any) in a table of 256^{12} $\phi(K)$ entries.

Now we perform an exhaustive search of the K key. To test a key assumption, we compute the k_{ini} , k_{τ_1} , k_{τ_2} and $\phi(K)$ values. Then we find the (c', c'') pair of colliding c values in the precomputed table, compute the two associated $c'*$ and $c''*$ values, determine which two precomputed lists $(V^{c'}[y])_{y*=0..15}$ $(V^{c''}[y])_{y*=0..15}$ of 16 ciphertext values each are associated with $c'*$ and $c''*$, and

finally compute the associated $(t^{c'}[y])_{y*=0..15}$ and $(t^{c''}[y])_{y*=0..15}$ bytes by means of partial Rijndael decryption. The two values associated with $y* = 0$ are first computed and compared. The two values associated with $y* = 1$ are only computed and compared if they are equal, etc, thus in average only two partial decryption are performed. If the two lists of 16 t bytes are equal, then there is an alarm, and the K is further tested with a few plaintexts and ciphertexts.

We claim than the complexity of the operations performed to test one K key is marginally less than one Rijndael encryption.

5 Conclusion

We have shown existence of collisions on 4-rounds of Rijndael, that permit us to create a distinguisher between 4-inner rounds of Rijndael and a random permutation and to mount an attack on a 7-rounds version of Rijndael for any key-lengths. This attack is not practicable because its complexity is very high even if the number of plaintexts to cipher is small. The complete version of Rijndael on ten rounds is not in danger.

Our future works concern performance of keyderivation : how to avoid testing each $(c'^*, c''*)$ pair separately or take opportunity of the keyschedule ? We will investigate other block values. We will work too on an amelioration of the extentions at end by using algebraic properties of S-box. This paper should not be used as an argument against Rijndael in the AES competition.

References

- [AES99] <http://www.nist.gov/aes>
- [DaKnRi97] J. Daemen, L.R. Knudsen and V. Rijmen, "The Block Cipehr Square". In *Fast Software Encryption - FSE'97* , pp. 149-165, Springer Verlag, Haifa, Israel, January 1997.
- [DaRi98] J. Daemen, V. Rijmen, "AES Proposal : Rijndael", *The First Advanced Encryption Standard Candidate Conference*, N.I.S.T., 1998.