A New Involutory MDS Matrix for the AES

Jorge Nakahara Jr and Élcio Abrahão
(Corresponding author: Jorge Nakahara Jr)

Department of Informatics, UNISANTOS
R. Dr. Carvalho de Mendonça, 144, Santos, Brazil (Email: jorge_nakahara@yahoo.com.br)

(Received July 7, 2006; revised and accepted Nov. 8, 2006)

Abstract

This paper proposes a new, large diffusion layer for the AES block cipher. This new layer replaces the ShiftRows and MixColumns operations by a new involutory matrix in every round. The objective is to provide complete diffusion in a single round, thus sharply improving the overall cipher security. Moreover, the new matrix elements have low Hamming-weight in order to provide equally good performance for both the encryption and decryption operations. We use the Cauchy matrix construction instead of circulant matrices such as in the AES. The reason is that circulant matrices cannot be simultaneously MDS and involutory.

Keywords: AES, involutory transformations, MDS matrices

1 Introduction

The Advanced Encryption Standard (AES) algorithm is an SPN-type cipher designed by J. Daemen and V. Rijmen for the AES Development Process [1]. The original cipher was called Rijndael, and it was selected out of fifteen candidates during the AES Development Process, initiated by the National Institute of Standards and Technology (NIST) in 1997. The AES will become the new *de facto* world standard in symmetric cryptography, as the successor of the Data Encryption Standard (DES) algorithm. In Sep. 2000, Rijndael was officially standardized as FIPS PUB 197 [24]. Rijndael (and the AES) have already been implemented in several programming languages and are embedded in several software systems [29]. The AES is the smallest instance of the Rijndael cipher [14], since the AES operates on 128-bit text blocks, under keys of 128, 192 or 256 bits, for which the cipher iterates ten, twelve and fourteen rounds, respectively.

There are four transformations in a full round of Rijndael: SubBytes (**SB**), ShiftRows (**SR**), MixColumns (**MC**) and AddRoundKey ($\mathbf{AK}_i$). The subscripts i denote the round number. One full round of Rijndael applied to a text block X consists of $\mathrm{AK}_i \circ \mathrm{MC} \circ \mathrm{SR} \circ \mathrm{SB}(X) = \mathrm{AK}_i\ (\mathrm{MC}\ (\mathrm{SR}\ (\mathrm{SB}(X))))$, namely function composition operates in right-to-left order. There is an input transformation, AK_0 prior to the first round, and the last round does not include MC. Further details about AES components can be found in [14].

This paper proposes a new diffusion layer for the AES cipher, that replaces the original SR and MC layers. This new layer consists of a 16×16 involutory MDS matrix, denoted $M_{16\times16}$. The new design was called MDS-AES. Thus, the new round structure of MDS-AES becomes $\mathrm{AK}_i \circ M_{16\times16} \circ \mathrm{SB}(X) = \mathrm{AK}_i\ (M_{16\times16}\ (\mathrm{SB}(X)))$. This design has two main consequences: (1) complete diffusion is achieved in a single round thus improving the overall security of the cipher because the branch number [13, 14] increases from 5 (in the AES) to 17; but (2) lower performance due to the larger number of matrix components.

This paper is organized as follows: Section 2 presents elementary mathematical concepts necessary for further developments in the following sections. Section 3 presents a new MDS matrix for the AES, aimed at replacing the SR and MC layers altogether. The new cipher is called simply MDS-AES. Section 4 presents a security analysis of MDS-AES. Section 5 concludes the paper.

2 Preliminaries

An involutory transformation (or simply involution) f is a self-inverse mapping, namely, $f(x) = f^{-1}(x)$, for all x over the domain of f. The use of involutions in cryptology dates back to Hebrew ciphers such as ATBASH, ALBAM and ATBAH, the German Enigma cipher [18], and more recently, the block ciphers Khazad [3] and Anubis [2]. There are several reasons for the use of involutional mappings. For instance, they reduce the implementation cost of both encryption and decryption operations, and imply that both transformations have the same cryptographic strength. Nonetheless, it is important that algebraic properties due to involutions do not lead to cryptanalytic attacks [6].

It is important to emphasize that making the diffusion layer of AES an involution does not make the full cipher an involution (e.g. because the S-box is not an involution).

2.1 Finite Fields

A field is a commutative ring (with unity) in which all nonzero elements have a multiplicative inverse [23]. In this paper we are concerned with the finite field GF(2^8) used in AES and related ciphers [2, 3, 14]. For the AES, GF(2^8) is represented as GF(2)/(m(x)), where $m(x) = x^8 + x^4 + x^3 + x + 1$ is an irreducible polynomial over GF(2). A polynomial representation is assumed for every element $a \in$ GF(2^8) in the AES. So, $a = (a_7, a_6, \ldots, a_1, a_0) = \sum_{i=0}^{7} a_i.x^i$, with $a_i \in GF(2)$ for $0 \leq i \leq 7$. A compact representation of an element $x \in$ GF(2^8) uses hexadecimal digits (denoted with subscript $_x$), expressing the coefficients of the polynomial representation. For instance, $x^7 + x^5 + x^3 + x^2 + 1 = AD_x$, and $m(x) = 11B_x$.

Addition in GF(2^8) is simply bitwise exclusive-or, since GF(2^8) has characteristic two. Multiplication in GF(2^8) is just polynomial multiplication modulo $m(x)$.

2.2 Error-correcting Codes

Error-correcting codes have already been suggested in the design of public-key algorithms by McEliece [21]. The use of error-correcting codes, such as MDS codes, in secret-key algorithms has been suggested by Vaudenay in [30].

The Hamming distance between two vectors (or code words) from the n-dimensional vector space $(GF(2^p))^n$ is the number of positions (out of n) by which the two vectors differ. The Hamming weight of a vector (or code word) $u \in (GF(2^p))^n$ is the Hamming distance between u and the null vector in $(GF(2^p))^n$, namely, the number of nonzero positions in u.

A linear $[n, k, d]$-code over GF(2^p) is a k-dimensional subspace of the vector space $(GF(2^p))^n$, where the Hamming distance between any two distinct n-element vectors is at least d, and d is the largest number with this property. A generator matrix G for a linear $[n, k, d]$-code C is a $k \times n$ matrix whose rows form a basis for C. Linear $[n, k, d]$-codes obey the Singleton bound, $d \leq n - k + 1$. A code that meets the Singleton bound, namely, $d = n - k + 1$, is called a Maximum Distance Separable or MDS code. Alternatively, an $[n, k, d]$-error correcting code with generator matrix $G = [I_{k\times k}|A]$, where $I_{k\times k}$ is the $k\times k$ identity matrix, and A is a $k\times(n-k)$ matrix, is MDS if and only if every square submatrix formed from i rows and i columns, $1 \leq i \leq \min\{k, n-k\}$, of A is nonsingular [22].

MDS matrices have become a fundamental component in the design of block ciphers such as SHARK [28], Square [12] and Rijndael [24], to guarantee fast and effective diffusion in a small number of rounds. One approach to obtain MDS matrices is the use of circulant matrices, where each row is a rotated instance (by a single unit) of the neighbouring rows (in the same direction). For example, a 4×4 circulant MDS matrix was used in the block cipher AES [24]. Nonetheless, this matrix is not involutory:

$$\begin{bmatrix} 02_x & 03_x & 01_x & 01_x \\ 01_x & 02_x & 03_x & 01_x \\ 01_x & 01_x & 02_x & 03_x \\ 03_x & 01_x & 01_x & 02_x \end{bmatrix} \cdot \begin{bmatrix} 02_x & 03_x & 01_x & 01_x \\ 01_x & 02_x & 03_x & 01_x \\ 01_x & 01_x & 02_x & 03_x \\ 03_x & 01_x & 01_x & 02_x \end{bmatrix} = \begin{bmatrix} 05_x & 00_x & 04_x & 00_x \\ 00_x & 05_x & 00_x & 04_x \\ 04_x & 00_x & 05_x & 00_x \\ 00_x & 04_x & 00_x & 05_x \end{bmatrix}.$$

The fact that the AES is not involutory is not due to the its elements. Consider an arbitrary 4×4 circulant matrix (with rows rotated to the right to mimic the MixColumns matrix),

$$A = \begin{bmatrix} a_0 & a_1 & a_2 & a_3 \\ a_3 & a_0 & a_1 & a_2 \\ a_2 & a_3 & a_0 & a_1 \\ a_1 & a_2 & a_3 & a_0 \end{bmatrix}.$$

If A were involutory then $A \cdot A = I_{4\times 4}$, where $I_{4\times 4}$ denotes the 4×4 identity matrix. This equation implies the following two restrictions (where + is exclusive-or):

$$a_0^2 + a_2^2 = 1,$$

and

$$a_1^2 + a_3^2 = 0. \tag{1}$$

If A were MDS, then in particular, the following determinants should be nonzero:

$$\begin{vmatrix} a_1 & a_3 \\ a_3 & a_1 \end{vmatrix} = a_1^2 + a_3^2 \neq 0. \tag{2}$$

The restriction (2) contradicts (1). Similar reasoning would result if the rows were rotated to the right. Thus, we conclude that 4×4 circulant matrices cannot be simultaneously MDS and involutory. Analogous reasoning also applies to larger circulant matrices. For that reason, from now on we consider other MDS construction techniques.

In [17], Junod and Vaudenay suggested some heuristics for constructing low implementation-cost MDS matrices. Their aim was to design matrices with a large number of elements equal to 1 (and other elements of low Hamming-weight), to minimize the implementation overhead. They claim that their construction leads to optimal matrices in the sense of smallest number of xor, table look-up and temporary variables. Nonetheless, their matrices are not involutory.

Another approach for the construction of involutory MDS matrices involves the so called Cauchy matrices [8, 31] used[1] in the block ciphers Khazad and Anubis [2]. Additionally, in these ciphers, the matrix elements were carefully chosen to minimize the number of primitive operations such as exclusive-or, table look-ups, and xtime calls [14]. In this paper, we look for large involutory, MDS matrices whose components also have low Hamming weight. In particular, we look for a 16×16 MDS matrix to provide complete diffusion in a single round (Figure 1).

[1] In those papers such matrices were called Hadamard.

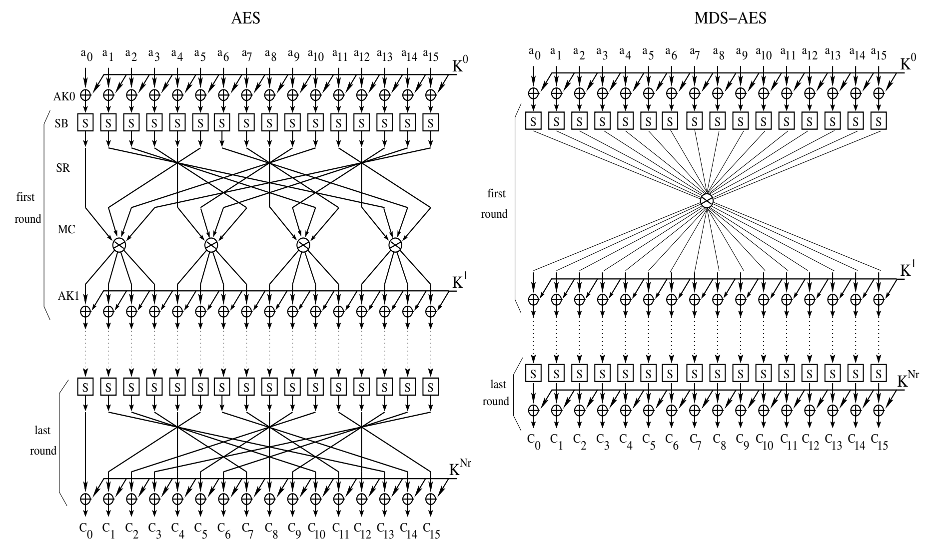

Figure 1: Computational graphs of AES and MDS-AES

Definition 1. *[31] Let x_1, $x_2, \ldots, x_m$, and y_1, $y_2 \ldots, y_n$ be elements in a field F, such that*

(1) $x_1, \ldots, x_m$ are distinct,

(2) $y_1, \ldots, y_n$ are distinct, and

(3) $x_i + y_j \neq 0$ for $1 \leq i \leq m$, $1 \leq j \leq n$.

An $m \times n$ Cauchy matrix over F has element $c_{i,j} = \frac{1}{x_i+y_j}$. The determinant of a square Cauchy matrix is $\dfrac{\prod_{i<j}(x_i - x_j)(y_i - y_j)}{\prod_{ij}(x_i + y_j)}$. Thus, by definition, a square Cauchy matrix is non-singular.

Square Cauchy matrices are unitary ($A^{-1} = A^T$), and symmetric ($A = A^T$, where A^T denotes the transpose of matrix A). These properties guarantee that Cauchy matrices are involutory.

In this paper, we look for matrices that satisfy several restrictions simultaneously:

1) Be MDS;

2) Be involutory;

3) Be 16×16;

4) Each matrix element, in GF(2^8), should have low Hamming weight;

5) The highest-order bits in each matrix element should preferably be in the least significant bit positions.

We call the modified AES cipher with the new MDS matrix substituting the SR and MC layers, simply MDS-AES (Figure 1). One full round of MDS-AES consists of an SB layer, followed by the new MDS matrix, and by the AK$_i$ layer. The output transformation consists of an SB layer followed by AK$_{Nr}$ (the key post-whitening layer) (Figure 1).

Criterion (1) guarantees fast diffusion in a small number of rounds. Restriction (2) aims at equal diffusion power for both encryption and decryption. Restriction (3) is due to the AES block size: 16 bytes. The last two restrictions aim at high performance (in software and hardware implementations). The Hamming weight of each matrix element impacts the number of xor and xtime operations. The higher-order bits in each matrix element affects the number of calls to xtime [14], which stands for multiplication[2] by 02_x in GF(2^8). Nonetheless, due to the size of these matrices, many more primitive operations will be required than in the AES. For instance, the 4×4 MDS matrix in the AES consumes two xtime and four xors per row of the matrix, or 8 xtime and 16 xors per MC matrix, or 32 xtime and 64 xors per round. The new MDS matrix in the MDS-AES design has 16 rows and columns which requires 688 xtime and 272 xors per round. Thus, the price for faster diffusion is lower performance. But, from the security point-of-view, the advantages of MDS-AES are significant (see Section 4).

3 A New Involutory MDS Matrix

We have searched for large involutory MDS matrices to replace the SR and MC layers in every round of the AES. Our search technique followed the Cauchy matrix construction (Section 2.2), in which all elements in a row are distinct. This construction method by itself guarantees that the resulting matrix is MDS and involutory.

Our search algorithm just needs to select the elements of the first row of the 16×16 MDS matrix since the Cauchy matrix construction only depends on this row. Once this row is determined, the remaining ones are simply permutations of the first row. Our first choice for an element in this row is 01_x because it is the smallest nonzero element

[2]This operation can be precomputed for all 256 possible inputs, and the result stored in table.

Table 1: Software performance comparison (estimated)

Cipher	# byte operations **per round**		
	# xtime	# xor	total
AES	32	64	96
MDS-AES	688	272	960

in GF(2^8). Further elements are selected in increasing order, such that

1) The elements are pairwise distinct;

2) The Hamming weight of each element is upper-bounded e.g. at most 4;

3) The highest order bit in every element is preferably in the least significant positions.

If the value does not match the above restrictions then the algorithm looks for the next larger value and apply the same procedure again until the 15th element is determined. The 16th (rightmost) element in the first row of the matrix is simply the exclusive-or of the previous fifteen elements and 1. This 16th element must also be different from the previous elements.

The best matrix found according to these restrictions is the following:

A performance comparison between $M_{16\times16}$ and the AES matrix simply counts the number of elementary operations per round, such as bytewise xors, number of xtime calls and number of table lookups (if xtime is stored in a table). For simplicity, we assume that each of these operations requires a single machine cycle. Thus, one single round of MDS-AES costs the same number of elementary operations as all MDS matrix computations in 9-round AES (under a 128-bit key).

4 Security Analysis

The AES has been intensively analysed since 1997. Nonetheless, the known results apply only to reduced-round variants: differential (DC) and linear (LC) analyses [9], multiset attacks [13, 15], impossible differential (ID) [4, 26, 27], collision [16], boomerang attacks [7] and so on.

A common feature exploited implicitly by all of these attacks on reduced-round AES is the slow diffusion via the combination of SR and MC layers. Notice that both SR and MC operate on 32-bit words at a time. It means that not all output bits depend on all input (plaintext and key) bits after a single round. In AES, for example, all plaintext bits diffuse completely after two rounds [13, p.29]. Key bits, though, diffuses completely after a number of rounds that depends on the user's key size. For 128-bit keys, complete diffusion is achieved after two rounds, and it takes one more round to reach complete diffusion for every 32 bits in the key size [13, p.29]. This design decision (incomplete diffusion in a single round) was probably based on a security-performance trade-off.

If diffusion were complete in a single round of the AES then, all of the known attacks against the AES would have much lower impact. Namely, the corresponding attack distinguishers would be shorter, and the corresponding attacks would affect a smaller number of rounds. Consequently, with complete diffusion, the nominal number of rounds of a cipher could be reduced, compensating the performance overhead due to a larger diffusion matrix.

As an example, (truncated) differential distinguishers covering 4-round AES contain at least 25 active S-boxes [10], as predicted in [13] (dashed lines in Figure 2(a), where a nonzero byte difference is denoted by δ). The new, larger MDS matrix (3) causes the differential distinguisher to contain at least 33 active S-boxes across only three rounds, due to the branch number of $M_{16\times16}$ (Figure 2(b)): sixteen active byte differences in the first round, one active byte difference in the second round and sixteen active byte differences in the third round. By counting the number of active S-boxes, the corresponding probability of the differential distinguishers drops from $(2^{-6})^{25}$ (in AES) to $(2^{-6})^{33}$ (in MDS-AES). This attack implies that at least three rounds are needed for MDS-AES. Similar reasoning applies to (conventional) linear [20] attacks.

Another important attack to consider on MDS-AES is the multiset technique [12], since it is the most effective attack known on (reduced-round instance of) AES. Moreover, in both ciphers, all internal operations are bytewise and bijective. Using the terminology of [12, 15], the propagation of active, passive, and balanced bytes in a multiset distinguisher can be described as follows. One can start with a multiset with one active (plaintext) byte only. The remaining fifteen plaintext bytes are passive. Thus, all bytes at the input to the second round will be active. After two rounds, due to $M_{16\times16}$, all sixteen bytes became balanced. That is the input multiset to the third round. Now, again due to $M_{16\times16}$, all output bytes are not balanced anymore. Thus, the subkey AK_3 can be recovered bytewise by partially decrypting the third round until the end of the second round. This attack can be extended by guessing a full subkey at the top, a trick already used in [12], but at an additional cost of 2^{128} key guesses. This attack implies that at least four rounds are needed for MDS-AES.

A sharp increase in security can be observed regarding the collision attack of Gilbert and Minier [16]. Their attack applies up to 7-round AES (although requiring almost the entire codebook) and depends on incomplete diffusion in a AES round. This attack, thus, is ineffective against MDS-AES, since complete diffusion is achieved in a single round.

Concerning impossible differential (ID) [4, 27] attacks, any truncated differential (with probability one) involving two rounds must involve at least 17 active S-boxes, because of the branch number of $M_{16\times16}$ matrix. Using the meet-in-the-middle (MITM) technique [4] we concluded that any pair of truncated differentials E_0 and E_1

$$M_{16\times16} = \left[\begin{array}{cccccccccccccccc}
01_x & 03_x & 04_x & 05_x & 06_x & 07_x & 08_x & 09_x & 0a_x & 0b_x & 0c_x & 0d_x & 0e_x & 10_x & 02_x & 1e_x \\
03_x & 01_x & 05_x & 04_x & 07_x & 06_x & 09_x & 08_x & 0b_x & 0a_x & 0d_x & 0c_x & 10_x & 0e_x & 1e_x & 02_x \\
04_x & 05_x & 01_x & 03_x & 08_x & 09_x & 06_x & 07_x & 0c_x & 0d_x & 0a_x & 0b_x & 02_x & 1e_x & 0e_x & 10_x \\
05_x & 04_x & 03_x & 01_x & 09_x & 08_x & 07_x & 06_x & 0d_x & 0c_x & 0b_x & 0a_x & 1e_x & 02_x & 10_x & 0e_x \\
06_x & 07_x & 08_x & 09_x & 01_x & 03_x & 04_x & 05_x & 0e_x & 10_x & 02_x & 1e_x & 0a_x & 0b_x & 0c_x & 0d_x \\
07_x & 06_x & 09_x & 08_x & 03_x & 01_x & 05_x & 04_x & 10_x & 0e_x & 1e_x & 02_x & 0b_x & 0a_x & 0d_x & 0c_x \\
08_x & 09_x & 06_x & 07_x & 04_x & 05_x & 01_x & 03_x & 02_x & 1e_x & 0e_x & 10_x & 0c_x & 0d_x & 0a_x & 0b_x \\
09_x & 08_x & 07_x & 06_x & 05_x & 04_x & 03_x & 01_x & 1e_x & 02_x & 10_x & 0e_x & 0d_x & 0c_x & 0b_x & 0a_x \\
0a_x & 0b_x & 0c_x & 0d_x & 0e_x & 10_x & 02_x & 1e_x & 01_x & 03_x & 04_x & 05_x & 06_x & 07_x & 08_x & 09_x \\
0b_x & 0a_x & 0d_x & 0c_x & 10_x & 0e_x & 1e_x & 02_x & 03_x & 01_x & 05_x & 04_x & 07_x & 06_x & 09_x & 08_x \\
0c_x & 0d_x & 0a_x & 0b_x & 02_x & 1e_x & 0e_x & 10_x & 04_x & 05_x & 01_x & 03_x & 08_x & 09_x & 06_x & 07_x \\
0d_x & 0c_x & 0b_x & 0a_x & 1e_x & 02_x & 10_x & 0e_x & 05_x & 04_x & 03_x & 01_x & 09_x & 08_x & 07_x & 06_x \\
0e_x & 10_x & 02_x & 1e_x & 0a_x & 0b_x & 0c_x & 0d_x & 06_x & 07_x & 08_x & 09_x & 01_x & 03_x & 04_x & 05_x \\
10_x & 0e_x & 1e_x & 02_x & 0b_x & 0a_x & 0d_x & 0c_x & 07_x & 06_x & 09_x & 08_x & 03_x & 01_x & 05_x & 04_x \\
02_x & 1e_x & 0e_x & 10_x & 0c_x & 0d_x & 0a_x & 0b_x & 08_x & 09_x & 06_x & 07_x & 04_x & 05_x & 01_x & 03_x \\
1e_x & 02_x & 10_x & 0e_x & 0d_x & 0c_x & 0b_x & 0a_x & 09_x & 08_x & 07_x & 06_x & 05_x & 04_x & 03_x & 01_x
\end{array}\right].$$

for an ID attack might cover at most three rounds (two rounds in the top-down direction and one round from the bottom up, or vice-versa), otherwise, there would be no contradiction in between E_0 and E_1, or the differentials would not hold with certainty. It means that an ID distinguisher (using the MITM) can cover at most three rounds of MDS-AES (compared to four rounds in AES). Moreover, in order to apply this distinguisher in an attack on 4-round MDS-AES, a full round subkey (128 bits) would need to be guessed at once (because of the $M_{16\times16}$ matrix), making the attack impractical (and not significant compared, for instance, with a multiset attack).

Concerning a boomerang attack [6], the construction of truncated differentials for a boomerang distinguisher has the same drawbacks as in the ID attack, namely, any pair of truncated differentials for a boomerang will have many active byte differences due to the $M_{16\times16}$ matrix. This phenomenon happens both for differentials going in the top-down and the bottom-up directions, independent of the initial number of nonzero byte differences. Therefore, boomerang distinguishers for MDS-AES are expected to be much shorter (two or three rounds) than the ones for the AES (five rounds), and thus not relevant compared to a multiset attack.

Based on the security analysis described previously, at least five rounds are recommended for MDS-AES.

5 Conclusions

This paper presented a new 16×16 MDS matrix for the AES cipher. This design was called MDS-AES. The new matrix replaces the SR and MC layers altogether, and provides complete diffusion in a single round, thus improving the overall security, since the branch number of the new matrix is 17, compared to 5 in the AES. Moreover, the involutory nature of the new matrix allows equally fast diffusion for both the encryption and decryption operations. The new matrix was found after a heuristic search for matrices that satisfy all of the following properties: (1) MDS, (2) involutory, (3) 16×16, (4) have elements with low Hamming weight, and (5) the highest-order bits in each matrix element are in the least significant bit positions.

The MDS-AES construction shows quite good resistance against differential, linear, multiset, collision, impossible differential and boomerang attacks. Given the attacks in Table 2, we conclude that (1) the resilience of MDS-AES is higher than that of the AES; (2) the best attack (meaning higher number of rounds and lowest computational resources) on reduced-round MDS-AES seems to be the multiset attack, which is the best known attack on reduced-round AES.

Notice that complete diffusion by itself cannot prevent other attacks such as slide and advanced slide attacks [5], nor mod-n attacks [19]; but the key schedule of the AES already avoid round self-similarity which is crucial for these attacks; we assume that MDS-AES uses the same key schedule algorithms of AES. Complete diffusion alone also does not prevent algebraic attacks [11]; a suggested countermeasure is to use an S-box with a more elaborate algebraic representation in GF(2^8), e.g. the S-box of Skipjack [25], whose algebraic representation is not as simple as that of the AES (see Appendix), and which looks random, namely, it does not seem to be derived from the inversion mapping in GF(2^8). Moreover, the Skipjack S-box has similar differential and linear profiles as the AES S-box.

It is left as an open problem if other 16×16, involutory MDS matrices can be found with elements having lower Hamming weight than $M_{16\times16}$, with consequently lower implementation costs.

The only drawback of MDS-AES is the performance penalty due to the larger number of primitive operations (xor, xtime, table look-up, temporary variables) implied by the larger number of matrix components, making it much slower than the AES. This fact indicates that the new design is mostly of theoretical interest.

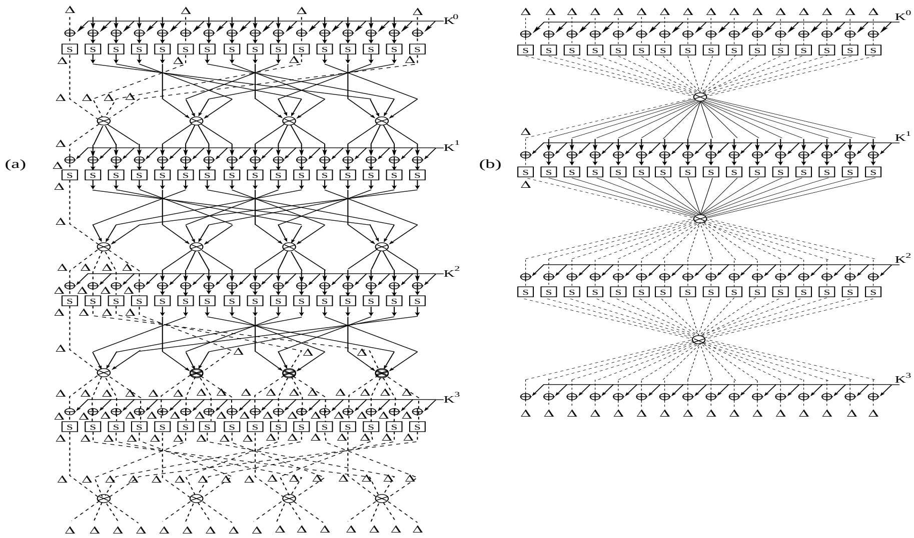

Figure 2: (a) Differential trail (dashed line) for AES, and (b) for MDS-AES

Table 2: Comparison of attacks on (reduced-round) AES and MDS-AES

Cipher	DC	LC	Multiset	Collision	ID	Boomerang
AES	2^{150} CP (4 rounds)	2^{150} KP (4 rounds)	2^9 CP (4 rounds)	$\approx 2^{128}$ CP (7 rounds)	$2^{29.5}$ CP (5 rounds)	2^{39} CPACC (5 rounds)
MDS-AES	2^{198} CP (3 rounds)	2^{198} KP (3 rounds)	2^9 CP (3 rounds)	ineffective	ineffective	ineffective

CP: Chosen-Plaintext; KP: Known-Plaintext.

A topic for further research is the determination of larger involutory MDS matrices (not of the Cauchy type) for Rijndael-160, Rijndael-192, and Rijndael-224. We could not derived such matrices because their dimensions are not powers of 2 (to fit the block size of the latter).

Acknowledgements

Author funded by FAPESP under contract # 2005/02102-9.

References

[1] AES, *The Advanced Encryption Standard Development Process*, 1997. (http://csrc.nist.gov/encryption/aes/)

[2] P. S. L. M. Barreto and V. Rijmen, "The ANUBIS block cipher," in *1st NESSIE Workshop*, Heverlee, Belgium, Nov. 2000.

[3] P. S. L. M. Barreto and V. Rijmen, "The KHAZAD legacy-level block cipher," in *1st NESSIE Workshop*, Heverlee, Belgium, Nov. 2000.

[4] E. Biham and N. Keller, "Cryptanalysis of reduced variants of Rijndael," in *3rd AES Conference*, New York, USA, 2000.

[5] A. Biryukov and D. Wagner, "Slide attacks," in *6th Fast Software Encryption Workshop*, LNCS 1636, pp. 245-259, Springer-Verlag, 1999.

[6] A. Biryukov, "Analysis of involutional ciphers: Khazad and Anubis," in *10th Fast Software Encryption Workshop*, LNCS 2887, pp. 45-53, Springer-Verlag, 2003.

[7] A. Biryukov, "The Boomerang attack on 5 and 6-round reduced AES," in *Proceedings of AES4 Conference*, LNCS 3373, pp. 11-15, Springer-Verlag, 2004.

[8] J. Bloemer, M. Kalfane, M. Karpinski, R. Karp, M. Luby, and D. Zuckerman, *An XOR-based Erasure-Resilient Coding Scheme*, ICSI TR-95-048, Aug. 1995.

[9] J. H. Cheon, M. Kim, K. Kim, J. -Y. Lee, and S. W. Kang, "Improved impossible differential cryptanalysis of Rijndael and crypton," in *Proceedings of ICISC 2001*, LNCS 2288, pp. 39-49, Springer-Verlag, 2001.

[10] D. Coppersmith, "The data encryption algorithm and its strength against attacks," *IBM Journal on Research and Development*, vol. 38, no. 3, pp. 243-250, 1994.

[11] N. T. Courtois and J. Pieprzyk, "Cryptanalysis of block ciphers with overdefined systems of quadratic equations," in *Advances in Cryptology (Asiacrypt'02)*, LNCS 2501, pp. 267-287, Springer-Verlag, 2002.

[12] J. Daemen, L. R. Knudsen, and V. Rijmen, “The block cipher SQUARE,” in *4th Fast Software Encryption Workshop*, LNCS 1267, pp. 149-165, Springer-Verlag, 1997.

[13] J. Daemen and V. Rijmen, “AES proposal: Rijndael,” in *1st AES Conference*, California, USA, 1998.

[14] J. Daemen and V. Rijmen, *The Design of Rijndael – AES – The Advanced Encryption Standard*, Springer-Verlag, 2002.

[15] N. Ferguson, J. Kelsey, S. Lucks, B. Schneier, M. Stay, D. Wagner, and D. Whiting, “Improved cryptanalysis of Rijndael,” in *7th Fast Software Encryption Workshop*, LNCS 1978, pp. 213-230, Springer-Verlag, 2000.

[16] H. Gilbert and M. Minier, “A collision attack on seven rounds of Rijndael,” in *3rd AES Conference*, New York, USA, 2000.

[17] P. Junod and S. Vaudenay, “Perfect diffusion primitives for block ciphes – Building efficient MDS matrices,” in *Selected Areas in Cryptology (SAC 2004)*, LNCS 3357, pp. 84-99, Springer-Verlag, 2004.

[18] D. Kahn, *The Codebreakers: the Story of Secret Writing*, MacMillan Publishing Company Inc., 1967.

[19] J. Kelsey, B. Schneier, and D. Wagner, “Mod n cryptanalysis, with applications against RC5P and M6,” in *6th Fast Software Encryption Workshop*, LNCS 1636, pp. 139-155, Springer-Verlag, 1999.

[20] M. Matsui, “Linear cryptanalysis method for DES cipher,” in *Advances in Cryptology (Eurocrypt’93)*, LNCS 765, pp. 386-397, Springer-Verlag, 1994.

[21] R. J. McEliece, *A Public-Key Cryptosystem Based on Algebraic Coding Theory*, pp. 42-44, DSN progress report, Jet Propulsion Laboratory, Pasadena, 1978.

[22] F. J. McWilliams and N. J. A. Sloane, *The Theory of Error Correcting Codes*, Amsterdam, The Netherlands, North Holland, 1977.

[23] A. J. Menezes, P. C. v. Oorschot, and S. Vanstone, *Handbook of Applied Cryptography*, CRC Press, 1997.

[24] NIST, *Advanced Encryption Standard AES*, FIPS PUB 197 Federal Information Processing Standard Publication 197, U.S. Department of Commerce, Nov. 2001.

[25] NIST, *Skipjack and KEA Specification, version 2.0*, May 1998.

[26] R. C. W. Phan, “Classes of impossible differentials of advanced encryption standard,” *IEE Electronics Letters*, vol. 38, no. 11, pp. 508-510, May 2002.

[27] R. C. W. Phan and M. U. Siddiqi, “Generalized impossible differentials of advanced encryption standard,” *IEE Electronics Letters*, vol. 37, no. 14, pp. 896-898, July 2001.

[28] V. Rijmen, J. Daemen, B. Preneel, A. Bosselaers, and E. D. Win, “The cipher SHARK,” in *3rd Fast Software Encryption Workshop*, LNCS 1039, pp. 99-112, Springer-Verlag, 1996.

[29] *The Block Cipher Rijndael*, http://www.iaik.tugraz.at/research/krypto/AES/.

[30] S. Vaudenay, “On the need for multipermutations: Cryptanalysis of MD4 and SAFER,” in *2nd Fast Software Encryption Workshop*, LNCS 1008, pp. 286-297, Springer-Verlag, 1995.

[31] A. M. Youssef, S. Mister, and S. E. Tavares, “On the design of linear transformation for substitution permutation encryption networks,” in *Workshop on Selected Areas in Cryptography (SAC’97)*, pp. 40-48, 1997.

Appendix

Compared with the expression $S[t] = 63_x + 8f_x \cdot t^{127} + b5_x \cdot t^{191} + 01_x \cdot t^{223} + f4_x \cdot t^{239} + 25_x \cdot t^{247} + f9_x \cdot t^{251} + 09_x \cdot t^{253} + 05_x \cdot t^{254}$ of the AES S-box, the algebraic expression of Skipjack’s S’-box in GF(2^8)=GF(2)[x]/(m(x)) has a more involved representation. For $t \in GF(2^8)$:

$S'[t] = a3_x + 10_x.t + b1_x.t^2 + 7a_x.t^3 + ec_x.t^4 + a5_x.t^5 + 8b_x.t^6 + 67_x.t^7 + 11_x.t^8 + a1_x.t^9 + 6e_x.t^{10} + af_x.t^{11} + 0f_x.t^{12} + 3c_x.t^{13} + d6_x.t^{14} + b9_x.t^{15} + 4f_x.t^{16} + 27_x.t^{17} + 5c_x.t^{18} + 6a_x.t^{19} + 6c_x.t^{20} + 9a_x.t^{21} + 1e_x.t^{22} + cf_x.t^{23} + 65_x.t^{24} + 77_x.t^{25} + 86_x.t^{26} + f5_x.t^{27} + 93_x.t^{28} + c8_x.t^{29} + 43_x.t^{30} + 43_x.t^{31} + 39_x.t^{32} + db_x.t^{33} + 85_x.t^{34} + 05_x.t^{35} + 36_x.t^{36} + 98_x.t^{37} + d9_x.t^{38} + 3b_x.t^{39} + 8a_x.t^{40} + c4_x.t^{41} + f9_x.t^{42} + 68_x.t^{43} + 3d_x.t^{44} + b0_x.t^{45} + ee_x.t^{46} + 0a_x.t^{47} + 74_x.t^{48} + 51_x.t^{49} + c1_x.t^{50} + d0_x.t^{51} + 76_x.t^{52} + 67_x.t^{53} + 88_x.t^{54} + d1_x.t^{55} + 38_x.t^{56} + 13_x.t^{57} + 06_x.t^{58} + d0_x.t^{59} + e2_x.t^{60} + 4b_x.t^{61} + 65_x.t^{6}2 + ea_x.t^{63} + 1d_x.t^{64} + 27_x.t^{65} + d9_x.t^{66} + 5d_x.t^{67} + 39_x.t^{68} + fb_x.t^{69} + c9_x.t^{70} + 13_x.t^{71} + 7c_x.t^{72} + 43_x.t^{73} + a6_x.t^{74} + 5f_x.t^{75} + dd_x.t^{76} + d9_x.t^{77} + 41_x.t^{78} + 99_x.t^{79} + 67_x.t^{80} + ee_x.t^{81} + 07_x.t^{82} + 90_x.t^{83} + 9d_x.t^{85} + af_x.t^{86} + 89_x.t^{87} + cf_x.t^{8}8 + c7_x.t^{89} + df_x.t^{90} + f5_x.t^{91} + ff_x.t^{92} + 1f_x.t^{93} + 78_x.t^{94} + da_x.t^{95} + 73_x.t^{96} + 1d_x.t^{97} + 8b_x.t^{9}8 + 08_x.t^{100} + e9_x.t^{101} + 84_x.t^{102} + 71_x.t^{103} + 16_x.t^{104} + 0b_x.t^{105} + 6b_x.t^{106} + 07_x.t^{107} + 92_x.t^{108} + f4_x.t^{109} + 05_x.t^{110} + 4e_x.t^{111} + d5_x.t^{112} + 1f_x.t^{113} + 29_x.t^{114} + 29_x.t^{115} + 08_x.t^{116} + 36_x.t^{117} + db_x.t^{118} + 2e_x.t^{119} + a2_x.t^{120} + 5d_x.t^{121} + 3d_x.t^{122} + 72_x.t^{123} + 36_x.t^{124} + a5_x.t^{125} + 60_x.t^{126} + da_x.t^{127} + 3c_x.t^{128} + 28_x.t^{129} + 55_x.t^{130} + a0_x.t^{131} + 36_x.t^{132} + 1a_x.t^{133} + 81_x.t^{134} + 60_x.t^{135} + 5b_x.t^{136} + bf_x.t^{137} + 0f_x.t^{138} + 40_x.t^{139} + 0a_x.t^{140} + 86_x.t^{141} + cf_x.t^{142} + 7f_x.t^{143} + 0a_x.t^{144} + e5_x.t^{145} + 5b_x.t^{146} + ed_x.t^{147} + a7_x.t^{148} + e3_x.t^{149} + a5_x.t^{150} + 11_x.t^{151} + da_x.t^{152} + 6b_x.t^{153} + 10_x.t^{154} + 92_x.t^{155} + d9_x.t^{156} + 6e_x.t^{157} + 7a_x.t^{158} + dc_x.t^{159} + 17_x.t^{160} + 84_x.t^{161} + e7_x.t^{162} + 62_x.t^{163} + 9f_x.t^{164} + d3_x.t^{165} + 0e_x.t^{166} + 71_x.t^{167} + 80_x.t^{168} + 13_x.t^{169} + f6_x.t^{170} + f3_x.t^{171} + 0d_x.t^{172} + 77_x.t^{173} + 37_x.t^{174} + f6_x.t^{175} + a7_x.t^{176} + 82_x.t^{177} + 61_x.t^{178} + 78_x.t^{179} + 39_x.t^{180} + 51_x.t^{181} + 3a_x.t^{182} + 3f_x.t^{183} + a4_x.t^{184} + e3_x.t^{185} + 38_x.t^{186} + 25_x.t^{187} + 95_x.t^{188} + 0e_x.t^{189} + 71_x.t^{190} + b1_x.t^{191} + 44_x.t^{192} + ce_x.t^{193} + 21_x.t^{194} + c6_x.t^{195} + 96_x.t^{196} + 13_x.t^{197} + d3_x.t^{198} + 0c_x.t^{199} + 13_x.t^{200} + e9_x.t^{201} + 19_x.t^{202} + af_x.t^{203} + 15_x.t^{204} + fa_x.t^{205} + 15_x.t^{206} + 4c_x.t^{207} + e6_x.t^{208} + 19_x.t^{209} + 98_x.t^{210} + 09_x.t^{211} + cd_x.t^{212} + ee_x.t^{213} + 10_x.t^{214} + 59_x.t^{215} + 1d_x.t^{216} + 5b_x.t^{217} + 6a_x.t^{218} + 8f_x.t^{219} + d5_x.t^{220} + d4_x.t^{221} + ed_x.t^{222} + ca_x.t^{223} +$

$02_x.t^{224} + fe_x.t^{225} + f7_x.t^{226} + be_x.t^{227} + a3_x.t^{228} + fa_x.t^{229} + 17_x.t^{230} + d3_x.t^{231} + 81_x.t^{232} + a0_x.t^{233} + fc_x.t^{234} + 78_x.t^{235} + 6c_x.t^{236} + bb_x.t^{237} + 9c_x.t^{238} + ab_x.t^{239} + 5e_x.t^{240} + 08_x.t^{241} + a2_x.t^{242} + 1b_x.t^{243} + 14_x.t^{244} + b8_x.t^{245} + 78_x.t^{246} + a4_x.t^{247} + 34_x.t^{248} + 4d_x.t^{249} + 65_x.t^{250} + da_x.t^{251} + d7_x.t^{252} + 6c_x.t^{253} + bc_x.t^{254}$.

Jorge Nakahara Jr obtained his BSc and MSc degrees in Computer Science from the Institute of Mathematics and Statistics (IME) of the University of São Paulo, in São Paulo, Brazil, in 1989 and 1996. He further obtained a MSc and PhD degrees in Electrical Engineering from the Katholieke Universiteit Leuven, in Leuven, Belgium, in 1998 and 2003. He is currently a member of the Distributed Systems group at the Univ. Católica de Santos, in Santos, Brazil.

Élcio Abrahão graduated as an Agronomic Engineer from Paulista State University, São Paulo, Brazil in 1993. He further specialized in Informatics at Federal University of Lavras, Minas Gerais, Brazil in 1998. He is a teacher at Faculdade de Informática Administração Paulista (FIAP) since 2002, as well as a software engineering consultant at Grass Roots Bittime America, Miami, USA. He is currently pursuing an MSc degree in Computer Science at Univ. Católica de Santos, in Santos, Brazil.