# Algebraic Cryptanalysis of Reduced AES

Ameneh FARHADIAN, M.R. AREF

*Abstract*—**Algebraic attack on AES was proposed, recently. It was called XSL attack. The suggested computational complexity of this attack on AES is not dependent on the number of cipher rounds, roughly. So the implementation of attack is not practical even for the low number of AES round. In addition, the estimate given for the number of linearly independent equations generated by XSL technique has not been proved exactly and it can not be checked even for reduced AES, because the mentioned reason. But there is a question: weather it may be possible to have another equation system expressing the AES that enable the algebraic attack on reduced AES with computational complexity proportional to the number of rounds. In this paper a new equation system for AES is proposed that results in new algebraic attack on reduced AES up to 5 rounds. Solving this equation system even by worst method like guessing the unknowns is faster than the brute force attack.**

***Keywords*** **— Algebraic cryptanalysis, AES, Closed form representation, Ground idea.**

## I. Introduction

Algebraic attack on ciphers has gained more and more attention in cryptography[1]-[6]. The idea of algebraic attack is to express a cipher by a system of equations whose solution revels the secret key. So the differences in the way of generating the equation system or in the methods of solving the equation may result in different algebraic attacks. The algebraic attack on AES [7] was proposed by Courtois and Pieprzyk in [1], it was called XSL attack. The computational complexity of this attack has not been proved, perfectly. Because the estimates given for the number of linearly independent equations generated by the XSL technique, appears to be inaccurate. Murphy and Robshaw have improved this attack by rewriting the equations in $GF(2^8)$ instead of GF(2)[2]. They argue that it should be possible to break 128-bit AES with an effort equivalent to $2^{100}$ AES encryption, of course if the estimates for XSL technique are valid.

In [8], a simple closed form of AES was presented. But solving this equation or using it in cryptanalysis has remained as an open problem, until now. In this paper we use solving such closed form of equation by some difference to cryptanalysis reduced AES. The obtained equation system could be solved faster than brute force attack even by guessing the unknowns.

The equation system is established in the next section. The Section 3 deals with solving discussion of the obtained equation. An improved attack is presented in Section 4. Finally we summarized the paper in the Section 5.

Manuscript received November 9, 2008. This work was supported in part by ITRC under Grant T500/5990.

A.Farhadian received the B.S., and M.S. degrees, in Electrical Engineering, from Sharif University of Technology of Iran. (Phone: +98-21-44990823; e-mail: afarhadian@ yahoo.com).

M.R. AREFis with the Electrical Engineering Department, Sharif University of Technology. (phone: +98-21-66165935; e-mail: aref@sharif.ir).

## II. Generating the Equation System

In algebraic cryptanalysis of block cipher, usually a big equation system with large number of equations and unknowns is established. But here, only one equation is written to describe the cipher and cryptanalysis is done by solving this equation for some chosen plaintexts.

In AES encryption, all the operations can be expressed in $GF(2^8)$ field. So every intermediate byte or output byte can be expressed in closed form of input bytes, easily. In [4], Ferguson et al. derived a closed formula for AES (rijndael) that can be seen as a generalization of continued fractions. We use such closed form equation by some modifications. Before generating the equation, let define the notation that will be used. We refer to FIPS 197 [7] for a full description of the cipher.

$p_{i,j}$ One byte of plaintext which is placed in i-th row and j-th column of block.

$c_{i,j}$ One byte of ciphertext which is placed in i-th row and j-th column of block.

$S(x)$ S-box function by input byte x.

$p_{i,j}^{(r)}$ The input byte of r-th round which is placed in i-th row and j-th column of block.

$\alpha_i, \beta_i, \gamma_i$ Coefficients of the mix column operation.

$\lambda_i$ Coefficients of the mix column inversion operation.

$k_{i,j}^{(r)}$ One byte of subkey in r-th round, which is placed in i-th row and j-th column of block.

Using the defined notation and description of the cipher, the $p_{i,j}^{(3)}$, one input byte of the third round can be written as

$$\begin{aligned} p_{m,n}^{(3)} = \; & \alpha_1 S\Big(\sum_e \beta_{1,e} S(p_{i_{1,e},j_{1,e}} + k_{i_{1,e},j_{1,e}}) + k_{l_1,k_1}^{(1)}\Big) \\ & + \alpha_2 S\Big(\sum_e \beta_{2,e} S(p_{i_{2,e},j_{2,e}} + k_{i_{2,e},j_{2,e}}) + k_{l_2,k_2}^{(1)}\Big) \\ & + \alpha_3 S\Big(\sum_e \beta_{3,e} S(p_{i_{3,e},j_{3,e}} + k_{i_{3,e},j_{3,e}}) + k_{l_3,k_3}^{(1)}\Big) \\ & + \alpha_4 S\Big(\sum_e \beta_{4,e} S(p_{i_{4,e},j_{4,e}} + k_{i_{4,e},j_{4,e}}) + k_{k_4,l4}^{(1)}\Big) + k_{p,q}^{(2)} \end{aligned} \quad (1)$$

Now we want to simplify this equation. For this reason, we choose only plaintexts which are identical in all bytes except one, say $p_{i_{1,1},j_{1,1}}$. We call it "active byte". Choosing plaintexts in this way enables us to summarize the equation. Now, we have some constant components in the equation for this set of chosen plaintexts. We summarize these constant parts in equation by assigning a new simple constant parameter. One constant part in the (1) within such chosen plaintexts are shown by underling. Using this idea, equation (1) is rewritten to

$$p^{(3)}_{m,n} = \alpha_1 S(\beta_1 S(p_{i_{1,1},j_{1,1}} + k_{i_{1,1},j_{1,1}}) + const_1) + const_2 \quad (2)$$

In (2), $const_2$ is equal to the part that is underlined in (1). As we see, the number of unknowns are reduced in (2), notably. If we consider the reduced 4-round AES, the $p^{(3)}_{i,j}$ byte can be expressed by ciphertext bytes, too. The expression of $p^{(3)}_{i,j}$ according to ciphertext bytes is

$$S(p^{(3)}_{m,n}) = k^{/(3)}_{m,n} + \sum_{1\le y\le 4} \lambda_y S^{-1}(c_{k_y,l_y} + k^{(4)}_{k_y,l_y}) \quad (3)$$

Please note that $k^{/(3)}_{m,n}$ is the subkey that is obtained after transposition of two linear operations in 3-th round. "Add round key" operation and "Mix column" operation are transported together. Since both of them are linear, this transposition doesn't have any effect on cipher. Just, we should notice that replacing these two operations requires changing the corresponding subkey. (By performing the mix column inversion operation on corresponding subkey)
Combining (2) and (3), we have

$$S(\alpha_1 S(\beta_1 S(p_{i_{1,1},j_{1,1}} + k_{i_{1,1},j_{1,1}}) + const_1) + const_2) = k^{/(3)}_{m,n} + \sum_{1\le y\le 4} \lambda_y S^{-1}(c_{k_y,l_y} + k^{(4)}_{k_y,l_y}) \quad (4)$$

Now, we could find a relation between plaintext bytes and ciphertext bytes for 4-round AES. Let develop this equation to 5-round AES. We know that each output byte of mix column operation in 3-th round is a linear combination of $S(p^{(3)}_{i,j})$ s. thus

$$b^{(3)}_{m,n} = \gamma_1 S(p^{(3)}_{i,j}) + \gamma_2 S(p^{(3)}_{i,j}) + \gamma_3 S(p^{(3)}_{i,j}) + \gamma_4 S(p^{(3)}_{i,j}) \quad (5)$$

And Substituting (5) into (4), it gives

$$\begin{aligned} b^{(3)}_{m,n} = {} & \gamma_1 S(\alpha_1 S(\beta_1 S(p_{i_1,j_1} + k_{i_1,j_1}) + const_{1,1}) + const_{2,1}) \\ & + \gamma_2 S(\alpha_2 S(\beta_2 S(p_{i_1,j_1} + k_{i_1,j_1}) + const_{1,2}) + const_{2,2}) \\ & + \gamma_3 S(\alpha_3 S(\beta_3 S(p_{i_1,j_1} + k_{i_1,j_1}) + const_{1,3}) + const_{2,3}) \\ & + \gamma_4 S(\alpha_4 S(\beta_4 S(p_{i_1,j_1} + k_{i_1,j_1}) + const_{1,4}) + const_{2,4}) \end{aligned} \quad (6)$$

In other hand, $b^{(3)}_{m,n}$ can be represented by ciphertext bytes as follows

$$b^{(3)}_{m,n} = S(k^{/(4)}_{m,n} + \sum_{1\le y\le 4} S^{-1}(c_{k_y,l_y} + k^{(5)}_{k_y,l_y})) + k^{(3)}_{m,n} \quad (7)$$

Combining [6] and [7] results in

$$\begin{aligned} & S^{-1}(k^{/(4)}_{m,n} + \sum_{1\le y\le 4} \lambda_y S^{-1}(c_{k_y,l_y} + k^{(5)}_{k_y,l_y})) + k^{(3)}_{m,n} = \\ & \gamma_1 S(\alpha_1 S(\beta_1 S(p_{i_1,j_1} + k_{i_1,j_1}) + const_{1,1}) + const_{2,1}) \\ & + \gamma_2 S(\alpha_2 S(\beta_2 S(p_{i_1,j_1} + k_{i_1,j_1}) + const_{1,2}) + const_{2,2}) \\ & + \gamma_3 S(\alpha_3 S(\beta_3 S(p_{i_1,j_1} + k_{i_1,j_1}) + const_{1,3}) + const_{2,3}) \\ & + \gamma_4 S(\alpha_4 S(\beta_4 S(p_{i_1,j_1} + k_{i_1,j_1}) + const_{1,4}) + const_{2,4}) \end{aligned} \quad (8)$$

The resulted equation is very interesting. We could find an equation that relates the plaintext bytes and cipher text bytes together, with low number of unknown variables. The number of unknown variables in this equation is only 15.

After generating the equation for 4-round and 5-round AES, let's investigate the solving strategy. In the next section we deal with the strategy of solving.

## III. SOLVING THE EQUATIONS

Since the number of unknowns in the equations is low, the first way that appears to solve the equation is to guess the unknowns. The equation corresponding to 4-round AES has 8 unknowns. And other equation corresponding to 5-round AES has 15 unknowns.

The second way is to substitute the S(x) relation in the equation. From [7], we know that

$$\begin{aligned} S(x) = {} & '63' + '05' x^{254} + '09' x^{253} + 'f9' x^{251} + '25' x^{247} \\ & + 'f4' x^{239} + '01' x^{223} + 'b5' x^{191} + '8f' x^{127} \end{aligned} \quad (9)$$

Substituting S(x) relation in original Equation results in a multivariate polynomial over GF($2^8$) with maximum possible degree. Solving the resulted multivariate polynomial seems to be very difficult. We don't have any proper tool to solve such equation.

Another way is to use $xy = 1$ relation for nonlinear part of s-box. This idea was used in establishing the equation system for XSL attack on AES. According to [1], it is possible to have 23 implicit quadratic equations with 80 distinct terms for each s-box. We define new variables and expand the equation to be possible to use such above-mentioned implicit equation system for s-box. Each s-box should be assigned to a new defined variable like as follows for 4-round AES equation

$$\underbrace{S(\alpha_1 S(\beta_1 \underbrace{S(p_{i_{1,1},j_{1,1}} + k_{i_{1,1},j_{1,1}})}_{y_1} + const_1) + const_2)}_{y_2} = k^{/(3)}_{m,n} + \sum_{1\le j\le 4} \lambda_j \underbrace{S^{-1}(c_{k_j,l_j} + k^{(4)}_{k_j,l_j})}_{z_j} \quad (10)$$

Thus

$$\begin{cases} y_1 = S(p_{i_{1,1},j_{1,1}} + k_{i_{1,1},j_{1,1}}) \\ y_2 = S(v_1), \quad v_1 = \beta_1 y_1 + const_1 \\ y_3 = S(v_2), \quad v_2 = \alpha_1 y_2 + const_2 \\ z_1 = S^{-1}(c_{k_1,l_1} + k^{(4)}_{k_1,l_1}) \\ z_2 = S^{-1}(c_{k_2,l_2} + k^{(4)}_{k_2,l_2}) \\ z_3 = S^{-1}(c_{k_3,l_3} + k^{(4)}_{k_3,l_3}) \\ z_4 = S^{-1}(c_{k_4,l_4} + k^{(4)}_{k_4,l_4}) \\ y_3 = k^{/(3)}_{m,n} + \lambda_1 z_1 + \lambda_2 z_2 + \lambda_3 z_3 + \lambda_4 z_4 \end{cases} \quad (11)$$

In above equation system, there are 10 equations over $GF(2^8)$. Each equation over $GF(2^8)$ is equal to 8 equation over GF(2). We have 7 s-box relations that each of them can generate 23 implicit quadratic equations. Therefore this equation system is equal to an equation system over $GF(2)$ with 185 ($7\times 23$ implicit quadratic $+3\times 8$ linear) equations and 136 unknown bits and nearly 591 distinct terms. The number of equations is more than the number of unknowns by 49, so the XL method can be successful to solve the system. But it seems that the work load will be more than to be acceptable. In future, may be, one can solve this equation system better.

Let come back to first way, guessing the unknowns. In the equation corresponding to 4-round AES, there are 8 unknown bytes, so the search space has $(2^8)^8$ elements. Each element should be tested by computing the equation and check the equality for it. To overcome the answers by chance, we should iterate the test at least 8 times for 8 different chosen plaintexts. Because the probability that equality holds by chance is $2^{-8}$. The computation of equation takes $1/8$ of work load taken for one 4-round AES encryption. Therefore the work load of this attack is $2^{64}$.

There are 15 unknown bytes in 5-round expressing equation. So, 15 bytes should be guessed. To overcome the answers by chance, we should iterate the test at least 15 times for 15 different chosen plaintexts. Finally, the work load will be $2^{121.5}$. This is lower than exhaustive search by factor of $2^{6.5}$.

In the next section we will improve this attack.

## IV. THE ATTACK IMPROVEMENT

As we saw in the previous section, the resulted equation can be solved by guessing the unknowns with computational complexity lower than exhaustive key search attack. Here, we want to improve the prior attack.

Equation (4) can be rewritten as follows

$$k^{/(3)}_{m,n} = S(\alpha_1 S(\beta_1 S(p_{i_{1,e},j_{1,e}} + k_{i_{1,e},j_{1,e}}) + const_1) + const_2) + \sum_{1\le y\le 4} \lambda_y S^{-1}(c_{k_y,l_y} + k^{(4)}_{k_y,l_y}) \quad (12)$$

We see that the left hand side of the above equation is a byte of subkey and is constant for every plaintext. Therefore it doesn't need to guess the $k^{/(3)}_{m,n}$. It is sufficient to compute the right hand side of the equation and to check if the computed values for right hand side are identical for all chosen plaintexts. If they are the same, the guessed values for unknowns can be correct. So it is sufficient to guess only 7 bytes and to compute the right hand side. By means of this idea, the number of unknown is reduced. The number of chosen plaintexts that the right hand side is computed for them should be enough to overcome the answers by chance. The probability that n bytes take the same value by chance is $2^8/(2^8)^n$. And the search space for 7 unknown bytes has $(2^8)^7$ elements. Thus n should be at least 8 to overcome the wrong solutions. In this case, the computational complexity would be $2^{56}$. It shows that the attack is improved by factor of $2^8$ to prior attack.

We define the $k^{/(3)}_{m,n}$ term as "Ground" of the equation [9]. Since it is constant within all chosen plaintexts and can be considered as a ground for our computations.

Now we want to use the "Ground" idea to improve the 5-round AES cryptanalysis, too. In (8), $k^{(3)}_{m,n}$ can be taken as a "Ground". Thus

$$\begin{aligned} Ground = k^{(3)}_{m,n} = \\ S^{-1}(k^{/(4)}_{m,n} + \sum_{1\le y\le 4} \lambda_y S^{-1}(c_{k_y,l_y} + k^{(5)}_{k_y,l_y})) \\ + \gamma_1 S(\alpha_1 S(\beta_1 S(p_{i_1,j_1} + k_{i_1,j_1}) + const_{1,1}) + const_{2,1}) \\ + \gamma_2 S(\alpha_2 S(\beta_2 S(p_{i_1,j_1} + k_{i_1,j_1}) + const_{1,2}) + const_{2,2}) \\ + \gamma_3 S(\alpha_3 S(\beta_3 S(p_{i_1,j_1} + k_{i_1,j_1}) + const_{1,3}) + const_{2,3}) \\ + \gamma_4 S(\alpha_4 S(\beta_4 S(p_{i_1,j_1} + k_{i_1,j_1}) + const_{1,4}) + const_{2,4}) \end{aligned} \quad (13)$$

Now, there are 14 unknowns, which should be guessed, to compute the "Ground" value. Since the search space has $(2^8)^{14}$ elements, we should compute the Ground parameter for at least 14 chosen plaintexts to escape from accidental wrong outcomes. If the guessed variables are correct, the computed Ground for all chosen plaintexts will be the same. In this case, the computational complexity is $2^{112}$ that is improved by factor of $2^8$ rather prior attack. It is better than brute force attack by factor of $2^{16}$.

## V. CONCLUSION

In this paper, a new algebraic attack on reduced AES is proposed. It was shown that for the reduced AES up to 5 rounds, it is possible to write an equation that relates the plaintext and ciphertext bytes together. Solving this equation even by guessing the unknown is faster than brute force attack. The attack was improved by the Ground idea. The proposed algebraic Ground attack could break 4-round and 5-round AES by respectively $2^{56}$ and $2^{112}$ computational complexity. The number of required chosen plaintexts for cryptanalysis the 4-round and 5-round AES is 8 and 15, respectively.

Although, some more efficient structural attacks such as square attack [10], impossible differential attack [11] and

collision attack [12] were proposed on reduced AES before, but the new proposed attack in this paper is the first successful algebraic attack on reduced AES. It seems to be hard to extend the attack to more than 5 rounds. But may be, one can do it some years ago.

## REFERENCES

[1] N. Courtois, J. Pieprzyk, "Cryptanalysis of Block Ciphers with Overdefined Systems of Equations." In*: Proceedings of Asiacrypt 2002*, LNCS 2501, Springer-Verlag 2002, pp. 267-287.

[2] S. Murphy, J.B. Robshaw,"Essential Algebraic Structure Within the AES"*, Proceedings of CRYPTO 2002*, LNCS 2442, Springer-Verlag 2002, pp.17-38.

[3] A. Biryukov ,and C. De Canniere, "Block ciphers and systems of quadratic equations", in *Fast Software Encryption, FSE 2003*, pp. 274-289.

[4] C. Carlos, S. Murphy, M. Robshaw, *Algebraic Aspects of the Advanced Encryption Standard ,* 2006, Approx. 155 p., Hardcover, ISBN:0-387-4363-1.

[5] C. Cid, S. Murphy ,and M. Robshaw*,* "Computational and Algebraic Aspects of the Advanced Encryption Standard", *Seventh International Workshop on Computer Algebra in Scientific Computing*, *CASC'2004*, Petersburg, Russia, 2004, pp. 93-103.

[6] I. Toli ,and A, Zanoni, "An Algebraic Interpretation of AES−128", *AES 4th International Conference*, AES 2004 Bonn, Germany, pp. 84-97.

[7] National Institute of Standards and Technology. "*Advanced Encryption Standard", FIPS 197,* 26 November 2001.

[8] N. Ferguson, R. Schroeppel, D. Whiting, A Simple algebraic representation of Rijndael, *Proceeding of the Eighth International Workshop on selected areas in cryptography (SAC'2001),* LNCS 2259, Springer-Verlag, 2001, pp. 103-111.

[9] A. Farhadian, "Algebraic Cryptanalysis of AES" M.S. thesis, Dept. Electrical. Eng., Sharif University of Technology, Tehran, Iran, 2006.

[10] J. Daemen, L.R. Knudsen, and V.Rijmen,"The Block Cipher Square", *Fast Software Encryption*, LNCS 1267, Springer-Verlag, 1997, pp. 149-165.

[11] E. Biham ,and N. Keller, "Cryptanalysis of Reduced Variants of Rijndael", *3rd AES Conference*, New York, USA, 2000.

[12] H. Gilbert and M. Minier, "A Collision Attack on 7 Rounds of Rijndael", *Proceeding of the Third Advanced Encryption Standard Candidate Conference,* NIST, 2000, pp. 230-241.