An Improved Fault Based Attack of the Advanced Encryption Standard

Debdeep Mukhopadhyay

Computer Sc. and Engg, IIT Kharagpur, India
debdeep@cse.iitkgp.ernet.in

Abstract. In the present paper a new fault based attack has been proposed against AES-Rijndael. The paper shows that inducing a single random byte fault at the input of the eighth round of the AES algorithm the block cipher key can be deduced. Simulations show that when two faulty ciphertext pairs are generated, the key can be exactly deduced without any brute-force search. Further results show that with one single faulty ciphertext pair, the AES key can be ascertained with a brute-force search of 2^{32}.

1 Introduction

In order to satisfy the security requirements of various information disciplines e.g. networking, telecommunications, data base systems and mobile applications, applied cryptography has gained immense importance now-a-days. To satisfy the high throughput requirements of such applications, the complex cryptographic systems are implemented by means of either VLSI devices (crypto-accelerators) or highly optimized software routines (crypto-libraries). The high complexity of such implementations raises concerns regarding their reliability. Hence in this scenario it is imperative that the crypto-algorithms should not only prevent conventional cryptanalysis but also should prevent the deduction of the keys from accidental faults or intentional intrusions. Such attacks are known as fault attacks and were first conceived in September 1996 by Boneh, DeMillo and Lipton [1,2] from Bellcore. The fault attack was applicable to public key cryptosystems and was extended to various secret key ciphers like DES, the technique being known as Differential Fault Analysis (DFA)[3]. On 2^{nd} October 2000, the US National Institute of Standards and Technology (NIST) selected Rijndael [4] as the Advanced Encryption Standard (AES) and thus replaced DES as a world-wide standard for symmetric key encryption. Thus smart cards and secure micro-controllers are designed using AES to protect both the confidentiality and the integrity of sensitive information. With the work on optical fault induction reported in [5], research in the field of fault-based side channel cryptanalysis of AES has gained considerable attention. Less costly methods for fault injection include variation of supply voltages, clock frequency, clock glitches or temperature variations. DFA on AES was reported in [6] by inducing faults at byte level to the input of 9^{th} round of AES using 250 faulty ciphertexts. In the fault based

B. Preneel (Ed.): AFRICACRYPT 2009, LNCS 5580, pp. 421–434, 2009.

attack on AES reported in [7] around 128 to 256 faulty ciphertexts are required to discover the key. Dusart et. al. [8] performs a Differential Fault Analysis on AES and shows that using a byte level fault induction anywhere between the eighth round and ninth round the attacker is able to break the key with 40 faulty ciphertexts. Finally, [9] shows that using byte level faults at the input of the eighth round or the input of the ninth round of a ten round AES-128 algorithm, an attacker can retrieve the whole AES-128 key with two faulty ciphertexts. Recently fault attacks on AES exploiting the key-scheduling algorithms [10,11] have been developed which compute the value of the AES key with a minimum of two faulty ciphertexts at the cost of a brute force search of 48 and 40 bits respectively.

In the present work we present a fault based side-channel attack on AES using a single byte level fault. In the proposed attack the attacker induces a random non-zero byte level fault at the input of the 8^{th} round by affecting a single byte of the data. Extensive experimentations have been performed on a PC and it has been found that the key can be obtained using only two faulty ciphertexts. The attack has a workload of around 2^{16} and does not require any brute-force to obtain the final key. It takes a few seconds on a PC for completion. The idea of using algebraic equations have also been adopted in [8]. But the equations proposed in the present paper lead to much simpler analysis and reduce the number of faulty ciphertexts required from 40 to 2. Unlike [10,11] the present work does not require any brute force search. The present paper shows that although the present attack, like [9] requires two faulty ciphertexts, it requires a fault induction at a single byte location, as opposed to two required in [9]. Further we also suggest in the paper a modification to the basic attack, so that with one faulty ciphertext, the present attack can obtain the key with a brute force search of 2^{32}, which can be performed in less than 15 minutes using a PC. In short the present paper demonstrates the fact that AES can be broken using a single byte fault for one single instance. This increases the probability of making a fault attack practical using simple and less costly methods, like clock glitches and voltage fluctuations.

The paper is organised as follows: *section 2* describes the AES-Rijndael algorithm. The fault model and the attack environment is stated in *section 3*. The working principle of the attack is described in *section 4*, while the proposed attack is described in *section 5*. Finally the results of the work are compared to existing research in *section 6*. The work is concluded in *section 7*.

2 The Description of AES-Rijndael Algorithm

The description of the AES-Rijndael algorithm may be found in [4]. The typical round is described in the current subsection. The 128 bit message and key sizes have been considered, but the discussion can be extended to other specifications of the Rijndael block cipher.

The 128 bit input block to AES is arranged as a 4×4 array of bytes, known as the state matrix, refer to *figure 1*. The elements of the matrix are represented

b_{00}	b_{01}	b_{02}	b_{03}
b_{10}	b_{11}	b_{12}	b_{13}
b_{20}	b_{21}	b_{22}	b_{23}
b_{30}	b_{31}	b_{32}	b_{33}

Fig. 1. The State Matrix of AES-Rijndael

by the variable, b_{ij}, where $0 \leq i, j \leq 3$ and i, j refers to the row and column positions.

The algorithm has ten rounds and the keys of each round are generated by a key scheduling algorithm. The design of the key scheduling algorithm of AES is such that the knowledge regarding any round key reveals the original input key (named as the master key) from which the round keys are derived. The input state matrix (plaintext) is transformed by the various round transforms. The state matrix evolves as it passes through the various steps of the cipher and finally emerges in the form of ciphertext.

The rounds of AES use the following steps (*figure 2*):

1. The Byte Sub Step: The Byte Sub is the only non-linear step of the cipher. It is a bricklayer permutation consisting of an S-box applied to the bytes of the state. Each byte of the state matrix is replaced by its multiplicative inverse, followed by an affine mapping. Thus the input byte x is related to the output y of the S-Box by the relation, $y = A.x^{-1} + B$, where A and B are constant matrices[4].
2. The ShiftRows Step: Each row of the state matrix is rotated by a certain number of byte positions. This is a byte transposition step.
3. The MixColumn Step: The MixColumn is a bricklayer permutation operating on the state column by column. Each column of the state matrix is considered as a 4-dimensional vector where each element belongs to $GF(2^8)$. A 4×4 matrix M whose elements are also in $GF(2^8)$ is used to map this column into a new vector. This operation is applied on all the 4 columns of the state matrix [4]. Here M is defined as follows:

$$M = \begin{pmatrix} 2 & 3 & 1 & 1 \\ 1 & 2 & 3 & 1 \\ 1 & 1 & 2 & 3 \\ 3 & 1 & 1 & 2 \end{pmatrix}$$

4. AddRoundKey: Each byte of the array is exclusive-ored with a byte from a corresponding array of round subkeys.

The first 9 rounds of AES-Rijndael are identical - only the last round is not because the MixColumn step does not exist.

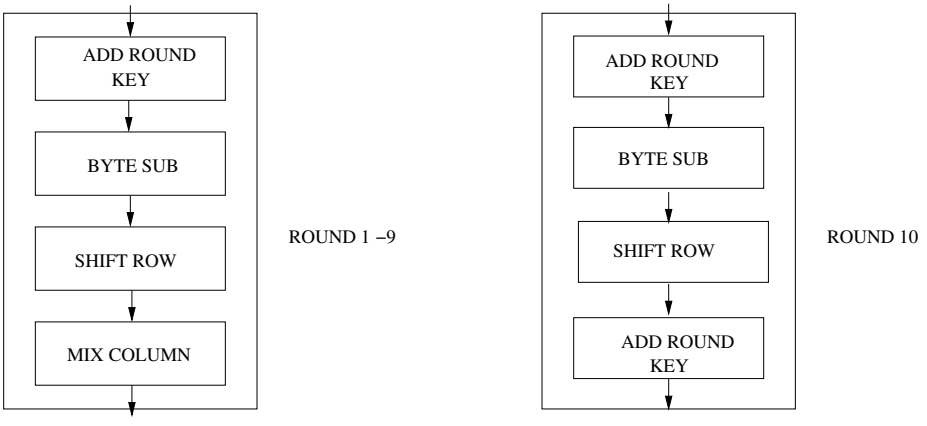

Fig. 2. The Round Transforms of AES-Rijndael

3 Fault Model Used and the Attack Environment

In this work, the fault assumed is a single byte fault. Single byte fault means the fault f_{ij} is injected in one particular byte b_{ij}, where i refers to the row position and j refers to the column position in the state matrix ($0 \leq i, j \leq 3$). The number of bits in the byte which are affected by the fault is indicated by $w(f_{ij})$, where $1 \leq w(f_{ij}) \leq 8$.

The attacker injects fault at the input of the eighth round in a single byte. The fault value can be arbitrary but non-zero. Before stating the fault attack techniques we outline two practical scenarios where the attack may be carried out. The scenarios show that depending upon the implementation of cryptographic hardware there are two different requirements on the attacker in order to inject the fault at any precise round location.

- Scenario 1: Certain implementations of AES-Rijndael requires pipelining at all stages (unrolled rounds), due to the requirement of throughput. Thus each key requires access to a key memory which cannot be shared among the rounds. Hence, the entire key has to be stored in a key register or memory. In such a case the attacker wishes to cause faults in the value that is being read from the memory while leaving the value stored in the memory unaffected. This does not hamper the normal functionality of the device and is thus undetectable. Further, in such an implementation large number of faulty ciphertexts can be obtained, since the key stored in the memory is unaffected. Thus the requirement on the attacker in such a case is *Control on Fault Location*.
- Scenario 2: The other way in which block ciphers like AES-Rijndael are implemented is through iterative structures (rolled rounds) or a combination of unrolled and rolled rounds. In such a case the key is not stored in the memory and thus the requirement on the attacker is *Control on Fault Timing*.

Imprecise control over fault location or fault timing hinders the attacker to be able to inject a fault at the intended round. In our present attack, we assume that the attacker intends to inject fault in a byte at the input of the 8^{th} round. In the following sections we present strategies through which if the attacker induces only a single byte fault for one time he is able to discover the key. First we explain the principle of the attack.

4 Working Principle of the Proposed Attack in the Ninth Round

The proposed attack is based on the induction of a single byte fault at the input of the eighth round. In order to explain the attack, first let us consider the scenario when there is a single byte disturbance at the input of the ninth round.

The 9^{th} round has a diffusion step, so a disturbance in one byte affects 4 bytes at the output. The last round does not have a diffusion step and so the disturbances remain in 4 bytes of the state matrix. If one traces the disturbance in the state matrix through the last two rounds the following properties can be identified. These properties can be utilized to develop an attack against the block cipher.

4.1 Property of the State Matrix

Figure 3 shows the propagation of the fault, when it is induced in a byte at the input of the ninth round Byte Sub. In the figure we consider the case when there is an arbitrary but non-zero disturbance at the 00^{th} byte of the state matrix. The disturbance or fault value is say f, and after the Byte Sub, then it is transformed to a value f'. The byte fault is propagated to four byte positions. As the figure suggests, the faults at the input of the tenth round Byte Sub have values of

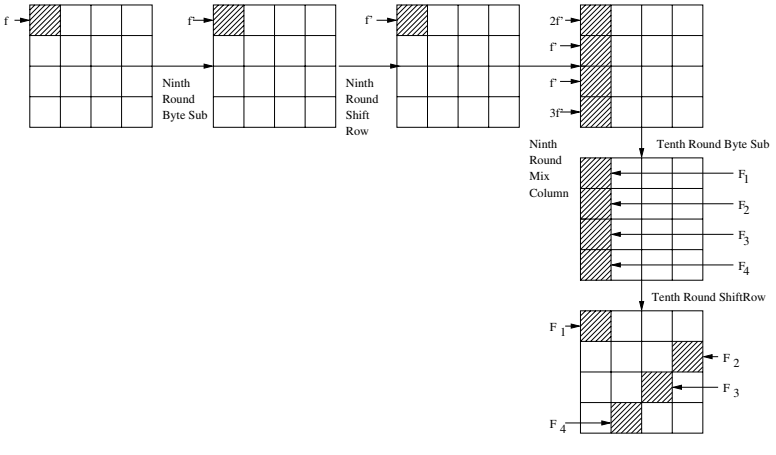

Fig. 3. Propagation of Fault Induced in the input of the ninth round of AES

$2f'$, f', f' and $3f'$. The faulty values after the tenth round Byte Sub gets transformed into F_1, F_2, F_3 and F_4. The attacker obtains a pair of ciphertext, (CT, CT'). The ciphertext CT is a fault free ciphertext and the ciphertext CT' is the ciphertext, when a fault is induced in a byte. When a bytewise fault is induced at the input of the ninth round Byte Sub, the difference of the ciphertexts CT and CT' has the pattern as shown in *figure 3* after the tenth round ShiftRow.

The fault pattern as shown in *figure 3* depicts the difference between the fault free ciphertext CT and the faulty ciphertext CT'. Let the values of the bytes shaded in *figure 3* after the tenth round Shift Row in the ciphertext CT be denoted by x_1, x_2, x_3 and x_4. Then the corresponding values for the faulty ciphertext CT' is denoted by $x_1 + F_1$, $x_2 + F_2$, $x_3 + F_3$ and $x_4 + F_4$. Here the sign $+$ stands for the bit-wise exclusive-or operation of two bytes. The corresponding key bytes are K_1, K_2, K_3 and K_4.

The fault pattern gives the following set of equations:

$$
\begin{aligned}
ISB(x_1 + K_1) + ISB(x_1 + F_1 + K_1) &= 2[ISB(x_2 + K_2) + ISB(x_2 + F_2 + K_2)] \\
ISB(x_2 + K_2) + ISB(x_2 + F_2 + K_2) &= \ ISB(x_3 + K_3) \ + ISB(x_3 + F_3 + K_3) \\
ISB(x_4 + K_4) + ISB(x_4 + F_4 + K_4) &= 3[ISB(x_2 + K_2) + ISB(x_2 + F_2 + K_2)]
\end{aligned}
$$

In the above set of equations ISB stands for the Inverse Byte Sub operation, which is defined as the inverse of the Byte Sub step. The values of (x_1, x_2, x_3, x_4) and $(x_1 + F_1, x_2 + F_2, x_3 + F_3, x_4 + F_4)$ are known to the adversary. The attacker intends to compute the values of K_1, K_2, K_3 and K_4 from the equations. The attacker evaluates the keys as follows: The attacker guesses the bytes K_1 and K_2 and checks whether they satisfy the first equation. The solution sets for the second and third equations are searched in parallel. Thus the time complexity of the key conjuring is 2^{16}. Finally since the variable K_2 is in all the three equations, the solution set of K_2 from each equation is intersected to arrive at a reduced solution space for K_2. The reduced space of K_2 is then used to find a reduced set of K_1, K_3 and K_4 from the three equations. It may be noted that the above attack does not depend upon the value of the induced fault, which may be random but non-zero. This completes a single pass of the algorithm with one (CT, CT') pair. In order to ascertain the key bytes, further passes of the algorithm are run with other (CT, CT') pairs. We have experimentally verified that the number of passes of the algorithm is two with a probability of around 0.99, thus revealing the key within two faulty encryptions. In the other few cases, a third faulty ciphertext is required to uniquely ascertain the key.

It may be observed that the attack proposed in [8] also use the properties of the MixColumn matrix, similar to the proposed attack. However one of the key differences between the present attack and the one proposed by [8] lies in the mechanism of solving the equations. [8] requires around $10 - 40$ faulty pairs to determine 4 key bytes of the 10^{th} round, while the proposed attack requires only 2. The difference comes from the fact that we use the Inverse SubByte, compared to SubByte used in [8]. This helps to obtain a better filter for the wrong key bytes.

4.2 A Working Example

In the present section we outline the working of the attack through an example.

Let the plaintext be:

$$\mathbf{PT_1} = \begin{pmatrix} 00000001 & 11111110 & 10000001 & 11111100 \\ 10100110 & 01101011 & 11100001 & 10100011 \\ 11110100 & 10100010 & 11110111 & 01111000 \\ 01000000 & 10100101 & 10001110 & 11110000 \end{pmatrix}$$

The plaintext is encrypted using AES-Rijndael encryption algorithm. The key matrix is:

$$\mathbf{K0} = \begin{pmatrix} 11100111 & 00101000 & 10010101 & 01100001 \\ 01110110 & 10101110 & 11110111 & 11001111 \\ 00010101 & 11011010 & 00110101 & 01011111 \\ 00111110 & 10000010 & 10100100 & 01001100 \end{pmatrix}$$

and the tenth round key is:

$$\mathbf{K10} = \begin{pmatrix} \mathbf{00101101} & 00010100 & 00011101 & 11000101 \\ 11110000 & 11100010 & 11010111 & \mathbf{01000001} \\ 11100010 & 00000111 & \mathbf{10100010} & 10110010 \\ 01010101 & \mathbf{11101010} & 01110000 & 00111100 \end{pmatrix}$$

The corresponding ciphertext is:

$$\mathbf{CT_1} = \begin{pmatrix} 11101110 & 01111111 & 11110100 & 01100101 \\ 01011000 & 01001101 & 10110101 & 10110101 \\ 11111001 & 00101001 & 11010010 & 11100010 \\ 10000101 & 00111011 & 11111100 & 11110111 \end{pmatrix}$$

The faulty ciphertext for a random fault induced in the 00^{th} position at the input of the ninth round leads to the following faulty ciphertexts:

$$\mathbf{CT'_1} = \begin{pmatrix} \mathbf{00101111} & 01111111 & 11110100 & 01100101 \\ 01011000 & 01001101 & 10110101 & \mathbf{11111111} \\ 11111001 & 00101001 & \mathbf{01111000} & 11100010 \\ 10000101 & \mathbf{10010101} & 11111100 & 11110111 \end{pmatrix}$$

The bytes in the faulty ciphertexts which are bolded show how the fault has propagated. The equations developed in the previous attack are applied to obtain the key bytes. From one faulty ciphertext the key space is reduced to around 32 possibilities per key byte.

The actual key can be ascertained if we consider another faulty encryption. Let another plaintext be:

$$\mathbf{PT_2} = \begin{pmatrix} 10101001 & 01010110 & 10001101 & 11001100 \\ 11110110 & 01001011 & 10100011 & 10000011 \\ 11110100 & 00000010 & 10100110 & 11110000 \\ 11110000 & 11100101 & 00111100 & 11110001 \end{pmatrix}$$

The corresponding ciphertext is:

$$\mathbf{CT_1} = \begin{pmatrix} 00001101 & 11111101 & 00111011 & 10001101 \\ 11000110 & 00100011 & 11110101 & 01110001 \\ 11001111 & 00101110 & 10100101 & 11011010 \\ 01110011 & 00001111 & 10101101 & 11000100 \end{pmatrix}$$

and the faulty ciphertext is

$$\mathbf{CT'_1} = \begin{pmatrix} \mathbf{01011100} & 11111101 & 00111011 & 10001101 \\ 11000110 & 00100011 & 11110101 & \mathbf{10100001} \\ 11001111 & 00101110 & \mathbf{11111011} & 11011010 \\ 01110011 & \mathbf{00011011} & 10101101 & 11000100 \end{pmatrix}$$

Intersection of the two solution sets leaves only one element, which is the correct solution. In this example we arrive at the key bytes: K_1=00101101, K_2=01000001, K_3=10100010 and K_4=11101010. The bold elements in the matrix of $K10$ show that the guesses are correct.

The above discussion shows that one byte fault reveals four bytes of the key. Thus for all the 16 bytes of AES it is necessary to induce faults at four bytes. However, often it may not be possible to induce faults at four byte positions. In the next section, we outline a modification to perform the fault based cryptanalysis of AES with one fault induction.

5 The Proposed Attack Strategy in the Eighth Round

In this attack we assume that the adversary has induced fault in a byte of the input to the eighth round. If the fault is induced in a byte of the state matrix, which is input to the eighth round, the disturbance spreads to the entire state matrix when it emerges out after the tenth round. In this case, a single byte fault creates four byte faults at the input of the ninth round. An attack similar to the previous attack can thus be used to compute the AES key.

5.1 Property of the State Matrix

Figure 4 shows the diffusion of a byte fault induced at the input of the eighth round. Similar to the previous section the various round operations transform the initial value of the fault f. The attacker observes, like in the previous cases two ciphers - one fault free and the other faulty. The difference of the state matrices of the two ciphers are depicted in *figure 4*.

The attacker knows the value of CT and CT' from the two ciphertexts that he obtains. Let, the two ciphertexts be represented by:

$$\mathbf{CT} = \begin{pmatrix} x_1 & x_2 & x_3 & x_4 \\ x_5 & x_6 & x_7 & x_8 \\ x_9 & x_{10} & x_{11} & x_{12} \\ x_{13} & x_{14} & x_{15} & x_{16} \end{pmatrix}$$

and

$$\mathbf{CT'} = \begin{pmatrix} x_1 + A_1 & x_2 + A_2 & x_3 + A_3 & x_4 + A_4 \\ x_5 + A_6 & x_6 + A_7 & x_7 + A_8 & x_8 + A_5 \\ x_9 + A_{11} & x_{10} + A_{12} & x_{11} + A_9 & x_{12} + A_{10} \\ x_{13} + A_{16} & x_{14} + A_{13} & x_{15} + A_{14} & x_{16} + A_{15} \end{pmatrix}$$

Here x_i and A_i $(1 \leq i \leq 16)$ are each one byte.
The corresponding key matrix for the tenth round is:

$$\mathbf{K_{10}} = \begin{pmatrix} K_{00} & K_{01} & K_{02} & K_{03} \\ K_{10} & K_{11} & K_{12} & K_{13} \\ K_{20} & K_{21} & K_{22} & K_{23} \\ K_{30} & K_{31} & K_{32} & K_{33} \end{pmatrix},$$

where each term k_{ij} $(0 \leq i, j \leq 3)$ is also a byte value.

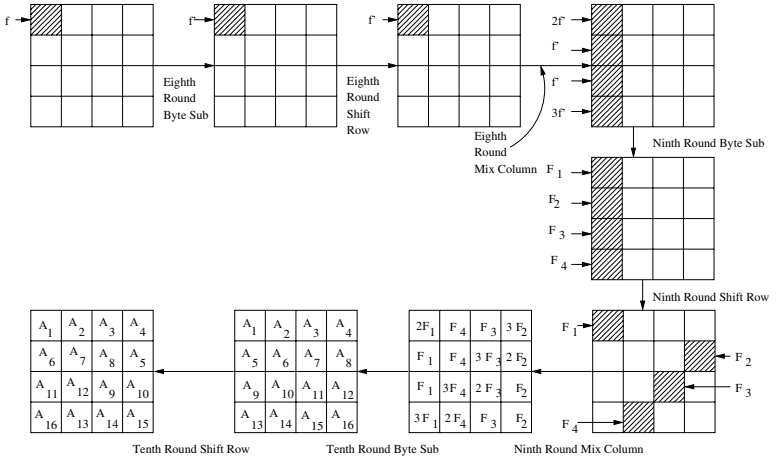

Fig. 4. Propagation of Fault Induced in the input of eighth round of AES

We note the state of the differences after the ninth round shift row from *figure 4*. Combining the above facts we obtain the following set of equations to evaluate the values of the key bytes K_{00}, K_{13}, K_{22} and K_{31}, thus revealing 32 bits of the AES key.

$$ISB(x_1 + K_{00}) + ISB(x_1 + A_1 + K_{00}) = 2[ISB(x_8 + K_{13}) + ISB(x_8 + A_5 + K_{13})]$$
$$ISB(x_8 + K_{13}) + ISB(x_8 + A_5 + K_{13}) = ISB(x_{11} + K_{22}) + ISB(x_{11} + A_9 + K_{22})$$
$$ISB(x_{14} + K_{31}) + ISB(x_{14} + A_{13} + K_{31}) = 3[ISB(x_8 + K_{13}) + ISB(x_8 + A_5 + K_{13})]$$

The unknowns in the above set of equations is the value of the key bytes K_{00}, K_{13}, K_{22} and K_{31}. The attacker similar to the previous strategy obtains reduced solution spaces for the bytes K_{00}, K_{13}, K_{22} and K_{31} from the three equations. The worst case complexity for one pass of the algorithm is 2^{16} and is again independent of the value of the fault induced. Another solution set for the

key bytes is obtained with another CT and CT' pair. The two solution sets are intersected to arrive at the correct key bytes, as the intersection set leaves only one element.

The above system of equation is used to reduce the possibilities of 32 bits of the key. In order to obtain the remaining three 32 bits of the AES key the attacker uses three more similar systems of equations. We briefly state the three other system of equations as follows:

In order to obtain $(K_{01}, K_{10}, K_{23}, K_{32})$ the attacker uses the following equations:

$$ISB(x_{15} + K_{32}) + ISB(x_{15} + A_{14} + K_{32}) = 2[ISB(x_2 + K_{01}) + ISB(x_2 + A_2 + K_{01})]$$
$$ISB(x_2 + K_{01}) + ISB(x_2 + A_2 + K_{01}) = [ISB(x_5 + K_{10}) + ISB(x_5 + A_6 + K_{10})]$$
$$ISB(x_{12} + K_{23}) + ISB(x_{12} + A_{10} + K_{23}) = 3[ISB(x_2 + K_{01}) + ISB(x_2 + A_2 + K_{01})]$$

In order to obtain $(K_{02}, K_{11}, K_{20}, K_{33})$ the attacker uses the following equations:

$$ISB(x_9 + K_{20}) + ISB(x_9 + A_9 + K_{20}) = 2[ISB(x_3 + K_{02}) + ISB(x_3 + A_3 + K_{02})]$$
$$ISB(x_3 + K_{02}) + ISB(x_3 + A_3 + K_{02}) = [ISB(x_{16} + K_{33}) + ISB(x_{16} + A_{15} + K_{33})]$$
$$ISB(x_6 + K_{11}) + ISB(x_6 + A_7 + K_{11}) = 3[ISB(x_3 + K_{02}) + ISB(x_3 + A_3 + K_{02})]$$

In order to obtain $(K_{03}, K_{12}, K_{21}, K_{30})$ the attacker uses the following equations:

$$ISB(x_7 + K_{12}) + ISB(x_7 + A_8 + K_{12}) = 2[ISB(x_{10} + K_{21}) + ISB(x_{10} + A_{12} + K_{21})]$$
$$ISB(x_{10} + K_{21}) + ISB(x_{10} + A_{12} + K_{21}) = [ISB(x_{13} + K_{30}) + ISB(x_{13} + A_{16} + K_{30})]$$
$$ISB(x_4 + K_{03}) + ISB(x_4 + A_4 + K_{03}) = 3[ISB(x_{10} + K_{21}) + ISB(x_{10} + A_{12} + K_{21})]$$

It may be noted that the equations are identical to that obtained in the previous section and thus the solutions are of similar nature. If only one faulty pair is used for the guessing of the key, the number of possible key values for each of the bytes of 32 bits of the key is around 32. Thus, one pass of the algorithm leaves on the average around $32^4 \approx 2^{20}$ possible candidate 32 bits of the key. Thus after one pass of the attack, number of possible 128 bit 10^{th} round AES keys is $(2^{20})^4 = 2^{80}$, as there are four 32 bit keys. Hence, brute force is not possible. However using another faulty ciphertext identifies the key uniquely. In the next section, we present an improvement to the above attack to reduce the brute force complexity to find the complete 128 bits AES key with one faulty ciphertext to only 2^{32}.

6 The Proposed Fault Attack with One Faulty Ciphertext

As discussed in the last section, the proposed attack computes four bytes of the AES key by using a faulty ciphertext and a fault free ciphertext and solving a system of equations. Each system of equation reduces the possible values of 32 bits of the AES key. There are four system of equations, each giving possible candidates for 32 bits of the key and the solutions are independent. Since, the

system of equations are similar in nature, their pruning capability of the key values are also similar. As we have discussed that the number of values for each of the bytes is 32, the total number of AES key values is as large as 2^{80}. In the following we propose an improvement which reduces the possibility of AES key values to as low as 2^{32} without using any other faulty ciphertext.

The improvement is based on the following observation: The fault induced at the input of the eighth round gets spread to the entire column after the MixColumn operation. Due to the next round shift row, this disturbance gets spread to each column. More specifically, each column has one byte of disturbance. The other three bytes in each column are undisturbed. We exploit this property for the further pruning of the key space.

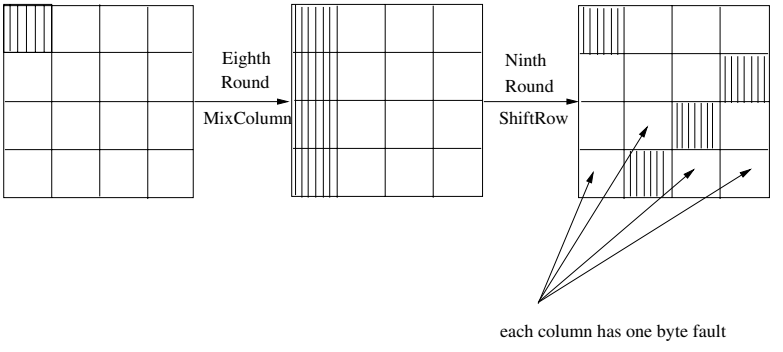

Fig. 5. Improvement to perform the fault attack with one faulty ciphertext

Assuming that the fault has been induced at the 00^{th} byte position of the input state matrix of the eighth round, the propagation of the fault through the MixColumn and Shift Row is as depicted in *figure 5*. It may be noted that each column at the output of the ninth round ShiftRow has a one byte disturbance. After the attack proposed in section 5, there are four reduced set of 32 bit keys. To further reduce the key size, the keys are used to check the fault pattern at the output of the ninth round ShiftRow. It is inspected whether only one specific byte position (depending on the initial fault location) is disturbed. For example, let us assume that after one pass of the previous algorithm we have a reduced set for 32 bits of the key, denoted by $(K_{00}, K_{13}, K_{22}, K_{31})$. We denote the fault free ciphertext by CT and faulty ciphertext by CT'. First we take the four ciphertext bytes, $(x_1, x_8, x_{11}, x_{14})$ and decrypt with the 32 bits of the key $(K_{00}, K_{13}, K_{22}, K_{31})$. Then we perform inverse Byte Sub. The same procedure is repeated over the faulty ciphertext, CT'. This reveals to us the fault pattern after AddRoundKey at the input of the tenth round. Since the fault pattern is not disturbed by the AddRoundKey after the ninth round, we have the same at the output of the ninth round MixColumn. Then we perform Inverse MixColumn to find out the fault pattern of the first column after the ninth round ShiftRow. We check whether the left byte is non-zero, while the rest are zero. We eliminate all key bytes which do not satisfy the above property. Using exhaustive

experimentations, we observe that after the above pruning there are around 240 possible key values for the four key bytes.

In order to obtain the entire key, the same pruning is applied on the reduced set of the other three 32 bits of the key. Hence, the total number of keys remaining after this additional pruning is $240^4 \approx 2^{32}$. This may be explored with todays computation power using a brute force search.

It may be noted that the proposed attack assumed that the fault was an arbitrary non-zero byte value at a known byte position (say the 00^{th} byte position of the state matrix). However the attack is easily extended to an attack which does not require the knowledge of the fault location, that is which byte position. This is based on the observation that depending on the fault location (the byte value where the fault is induced) we have four sets of equations. This cluster of equations varies if the byte position where the fault is located changes. Since there can be 16 byte positions, there are 16 clusters of such equations. It may be easily noted that the cluster of equations are identical and have the same pruning power. Thus, if the attacker induces a byte fault of arbitrary but non-zero value at the input of the eighth round, but is not sure about the exact byte location, he assumes the byte location and runs the above attack. Thus for each possible byte location, the proposed attack gives 2^{32} possible AES keys. Now varying the fault position, the number of keys is reduced to $16 \times 2^{32} = 2^{36}$, which is also within practical limits.

Next we present comparisons of our work with existing research in this area.

7 Comparison with Existing Works and Experimental Results

There have been considerable number of works on the subject. In this section we compare the existing fault based attack on AES with the help of *table 1*. The comparisions show that the current fault attack requires the minimum of faulty encryptions in order to derive the key like [9]. Recently, there have been some related fault attacks on AES, using the properties of the key scheduling algorithm. We compare our result with these attacks in *table 2*.

If the work reported in [9] be compared with the present attack based on the result with one faulty encryption, then both the attacks reduce the number of possible 32 bits of the key. The present attack reduces the number of candidate keys to an average of about 240 compared to 1036 required in the previous attack. Often the second fault induction may not be possible, in such a case the present attack can lead to a brute force attack after reducing the key space by using one faulty ciphertext. Thus if only faulty ciphertext is used, the present attack requires a brute force search of 2^{32} if the byte position of the fault in the state matrix is known and 2^{36} if not known. In comparison the work of Piret et. al. [9] requires around $(2^{10})^4 = 2^{40}$, which is higher than that of the present attack.

Table 1. Comparison of Existing Fault Attacks on AES exploiting properties of the encryption function

Reference	Fault Model	Fault Location	No. of Faulty Encryptions
[7]	Force 1 bit to 0	Chosen	128
[7]	Implementation Dependent	Chosen	256
[6]	Switch 1 bit	Any bit of chosen bytes	≈ 50
[6]	Disturb 1 byte	Anywhere among 4 bytes	≈ 250
[8]	Disturb 1 byte	Anywhere between last two MixColumn	≈ 40
[9]	Disturb 1 byte	Anywhere between 7^{th} round and 8^{th} round MixColumn	2
This Paper	Disturb 1 byte	Anywhere between 7^{th} round MixColumn and 8^{th} round MixColumn	2

Table 2. Comparison with Existing Fault Attacks on AES exploiting key scheduling

Reference	No of fault Injection Points	No. of Faulty Encryptions	Brute-force Search
[10]	1	2	2^{48}
	2	4	2^{16}
	3	7	0
[11]	1	2	2^{40}
	3	7	0
Our Attack	1	2	0
	1	1	2^{32}

8 Conclusions

The paper proposes an improved fault based cryptanalysis of the AES algorithm. The work shows that using only one arbitrary but non-zero byte fault at the input of the eighth round MixColumn, the 128 bit AES key can be deduced. Results have been furnished to show that the proposed attack leads to the evaluation of the exact key without any requirement for brute force search if two faulty ciphertexts are available. The attack has been further improved to show that even if only one faulty ciphertext is available, the AES key can be ascertained with a brute force of only 2^{32}, thus significantly improving existing fault based cryptanalysis of AES.

References

1. Boneh, D., DeMillo, R.A., Lipton, R.J.: On the Importance of checking cryptographic Protocols for Faults. In: Fumy, W. (ed.) EUROCRYPT 1997. LNCS, vol. 1233, pp. 37–51. Springer, Heidelberg (1997)
2. Boneh, D., DeMillo, R.A., Lipton, R.J.: On the Importance of Eliminating Errors in Cryptographic Computations. Journal of Cryptology, 101–120 (2001)
3. Biham, E., Shamir, A.: Differential Fault Analysis of Secret Key Cryptosystems. In: Kaliski Jr., B.S. (ed.) CRYPTO 1997. LNCS, vol. 1294, pp. 513–525. Springer, Heidelberg (1997)
4. Daemen, J., Rijmen, V.: The Design of Rijndael. Springer, Heidelberg (2002)
5. Skorobogatov, S., Anderson, R.: Optical Fault Induction Attacks. In: Kaliski Jr., B.S., Koç, Ç.K., Paar, C. (eds.) CHES 2002. LNCS, vol. 2523, pp. 2–12. Springer, Heidelberg (2003)
6. Giraud, C.: DFA on AES. Cryptology ePrint Archive, Report 2003/008 (2003)
7. Blomer, J., Seifert, J.P.: Fault Based Cryptanalysis of the Advanced Encryption Standard (AES). In: Wright, R.N. (ed.) FC 2003. LNCS, vol. 2742, pp. 162–181. Springer, Heidelberg (2003)
8. Dusart, P., Letourneux, G., Vivolo, O.: Differential Fault Analysis on A.E.S. (2003), `http://eprint.iacr.org/2003/010`
9. Piret, G., Quisquater, J.J.: A Differential Fault Attack Technique against SPN Structures, with Application to the AES and Khazad. In: Walter, C.D., Koç, Ç.K., Paar, C. (eds.) CHES 2003. LNCS, vol. 2779, pp. 77–88. Springer, Heidelberg (2003)
10. Takahashi, J., Fukunaga, T., Yamakoshi, K.: DFA mechanism on the AES schedule. In: Proceedings of 4^{th} International Workshop on Fault Detection and Tolerance in Cryptography, FDTC, pp. 62–72 (2007)
11. Takahashi, J., Fukunaga, T.: Differential Fault Analysis on the AES Key Schedule (2007), `http://eprint.iacr.org/2007/480`