A Diagonal Fault Attack on the Advanced Encryption Standard

Dhiman Saha * Debdeep Mukhopadhyay ** Dipanwita RoyChowdhury ***

Keywords: Fault Attack, AES-Rijndael, Clock Glitching

Abstract. *The present paper develops an attack on the AES algorithm, exploiting multiple byte faults in the state matrix. The work shows that inducing a random fault anywhere in one of the four diagonals of the state matrix at the input of the eighth round of the cipher leads to the deduction of the entire AES key. We also propose a more generalized fault attack which works if the fault induction does not stay confined to one diagonal. To the best of our knowledge, we present for the first time actual chip results for a fault attack on an iterative AES hardware running on a Xilinx FPGA platform. We show that when the fault stays within a diagonal, the AES key can be deduced with a brute force complexity of approximately* 2^{32}*, which was successfully performed in about* 400 *seconds on an Intel Xeon Server with* 8 *cores. We show further that even if the fault induction corrupts two or three diagonals,* 2 *and* 4 *faulty ciphertexts are necessary to uniquely identify the correct key.*

1 Introduction

Fault tolerance in cryptography is nowadays a widely researched topic. The advent of mobile handsets, smart cards, personal digital assistants (PDAs) with cryptographic hardware requires protection against accidental or intentional faults. The first use of faults to attack crypto-hardware dates back to 1996 by Boneh, DeMillo and Lipton [1, 2] from Bellcore. Since then fault based attacks have been extended successfully to both asymmetric and symmetric ciphers. The concept of Differential Fault Analysis (DFA) was introduced by Biham et. al. [3] on the Data Encryption Standard (DES). The Advanced Encryption Standard (AES), being the global standard for sensitive data encryption, has been a popular target for fault attacks. With the work on optical fault induction reported in [4], 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. Several differential fault attacks on AES have been reported in literature. While most attacks exploit the properties of the encryption function, recent reported attacks have also targeted the key scheduling. In [5], authors mount DFA on AES by inducing faults at *byte* level to the input of ninth round of AES using 250 faulty ciphertexts. The

* Dhiman Saha is a PhD Student in the Department of Computer Sc. and Engg, IIT Kharagpur, India. E-mail: dhimans@cse.iitkgp.ernet.in

** Debdeep Mukhopadhyay is an Assistant Professor in the Department of Computer Sc. and Engg, IIT Kharagpur. E-mail: debdeep@cse.iitkgp.ernet.in.

*** Dipanwita RoyChowdhury is a Professor in the Department of Computer Sc. and Engg, IIT Kharagpur. E-mail: drc@cse.iitkgp.ernet.in

attack reported in [6] recovers the key with around 128 to 256 faulty ciphertexts. In [7], Dusart et. al. show that using a *byte* level fault induction anywhere between the eighth round MixColumn and ninth round MixColumn, the attacker is able to break the key with 40 faulty ciphertexts. The authors of [8] mounted an attack on AES with just single *byte* faults using two faulty outputs. The point of induction was at the input of the eighth or ninth round. In [9], the authors present a fault attack on AES when the fault is induced in a 32 bit word of AES in the ninth round. The authors propose two models for fault occurrence. In the first model, they assume that at least one of the bytes among the four targetted bytes are uncorrupted. While in the second model they assume that all the four bytes are corrupted. The former fault requires 6 faulty ciphertexts, while the later requires around 1500 faulty ciphertexts to discover the key. We observe that when the assumptions are on the value of a byte (either it being faulty or uncorrupted) the number of faulty pairs is quite small. We claim that attacks based on multiple byte faults are more practical as opposed to those based on single byte faults. Also as induction of faults requires high precision instruments (more the precision, more the cost!) and are harder to guarantee, an attack which requires large number of faulty pairs is impractical. Among the attacks on key expansion, it has been suggested in [10, 11] that with *byte* level fault, a minimum of two faulty ciphertexts and a brute force search of 48 and 40 bits respectively reveal the AES key. In [12] a fault attack against AES was proposed, which revealed the key using a single *byte* fault induction at the input of the eighth round. The attack exploited the inter-relations between the fault values in the state matrix after the ninth round MixColumn operation. Through simulations it was shown that the attack reduced the AES-128 key space to 2^{32} using a single faulty ciphertext.

However in spite of several research works, there is surprisingly no reported work on actual fault attacks on real life hardware. Most of the attacks proposed, model the fault as a non-zero disturbance of a single byte of the AES state matrix. In this paper we introduce a new fault attack on AES, based on the fault induction in the *diagonals* of the AES state matrix at the input of the eighth round. Our attack named as the *Diagonal Fault Attack* is thus based on a multi-byte level fault modeling as opposed to a single byte level fault model. We have verified the entire attack on an iterative architecture of AES on a Xilinx FPGA platform with real-time fault injection using clock glitching via less sophisticated and less costly instruments. We show that when a non-zero fault is induced in one of the four diagonals of the AES state matrix at the input of the eighth round, a single faulty ciphertext is needed to reduce the key space of AES-128 to 2^{32}. The brute force search requires around 400 seconds on an eight core Intel Xeon server running Linux operating system. Further, we present attack methods to retrieve the key when the faults are not confined to one diagonal. We show that when two diagonals are corrupted the attacker needs two faulty ciphertexts, whereas if three diagonals are disturbed four ciphertexts are required to ascertain the correct key. Our experimental findings show that when the eighth round of AES is clocked at slightly higher than the maximum permissible frequency, in all the faulty cases at most three of the four diagonals are disturbed, and hence can be cryptanalyzed by the proposed method.

The paper is organised as follows: *section 2* describes the AES-Rijndael algorithm. The fault model is described in *section 3* and the attack environment is stated in *section 4*. *Section 5, 6* and *7*, propose the attacks using three different

fault models. *Section 8* reports the experimental results, and the comparisions are furnished in *section 9*. The working principle of the Finally, the work is concluded in *section 10*.

2 The Description of AES-Rijndael Algorithm

The description of the AES-Rijndael algorithm may be found in [13]. 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 by the variable, b_{ij}, where $0 \leq i, j \leq 3$ and i, j refers to the row and column positions.

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

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[13].
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 [13]. 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.

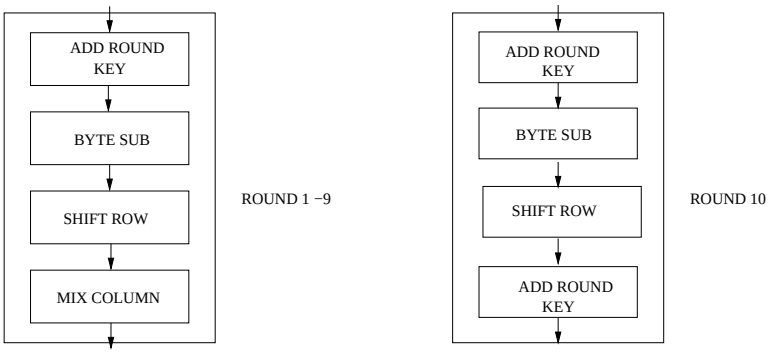

Fig. 2. The Round Transforms of AES-Rijndael

The proposed fault attack is based on a fault model which assumes fault induction in the diagonals of the AES state matrix. Hence before proceeding to the attack we give a formal definition of the diagonal of the state matrix.

Definition 1. *Diagonal: A diagonal is a set of four bytes of the state matrix, where the* i^{th} *diagonal is defined as follows:*

$$D_i = \{b_{j,(j+i)mod4} \quad ; \quad 0 \leq j < 4\} \tag{1}$$

According to the above definition and with reference to the state matrix of AES (refer figure 2*) we obtain the following four diagonals.*

$$D_0 = (b_{00}, b_{11}, b_{22}, b_{33})$$
$$D_1 = (b_{01}, b_{12}, b_{23}, b_{30})$$
$$D_2 = (b_{02}, b_{13}, b_{20}, b_{31})$$
$$D_3 = (b_{03}, b_{10}, b_{21}, b_{32})$$

In the next section, we present the fault model used in the attack.

3 Fault Model of the Attack

The Fault model determines the assumptions that we make about the nature of the fault that are induced in the attack. The implementation of AES we target is an iterative one. Research shows that unrolled or pipelined design of AES are unpopular because they do not allow to operate the cipher in Output Feedback Mode (OFB) or Cipher Block Chaining (CBC) mode, which are more secured modes of operation.

The attacker aims to inject a fault at the input of the eighth round. An iterative design helps in this regard, as the attacker is able to control the timing of fault induction by simply counting the number of clock edges from the start of an encryption. We classify the induced faults into four models:

Model 0 (M0): A random non-zero fault is induced in one of the diagonals, D_0, D_1, D_2 and D_3.

Model 1 (M1): A random non-zero fault occurs at most across two diagonals.

Model 2 (M2): A random non-zero fault occurs at most across three diagonals.

Model 3 (M3): A random non-zero fault occurs at most across all four diagonals.

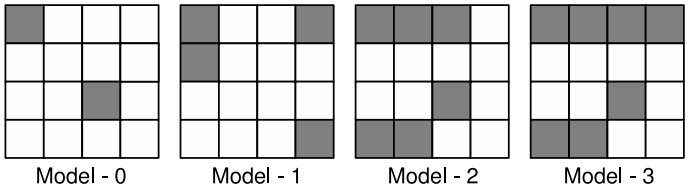

Fig. 3. Fault Models

Figure 3 shows example faulty states according to the above models. It may be noted in the figure, that in the first state matrix only diagonal D_0 is affected. In the second state matrix both D_0 and D_3 is corrupted. The diagonals D_0, D_2 and D_3 are disturbed in the third state matrix, whereas all the four diagonals are affected in the fourth state matrix. Thus the faults in the four state matrices are respectively in accordance with models, 0, 1, 2 and 3.

It may be noted that all the models are multi-byte fault models and thus attacks based on these models are more feasible than those based on single byte faults as found in the literature. Among the models, $M0 \subseteq M1 \subseteq M2 \subseteq M3$. Thus in the series, $M0, M1, M2, M3$, the higher fault models are more relaxed and thus attacks based on them are more realistic. For example, most ideal fault model would have been $M3$ which captures all possible faults. We show in our proposed attack methods, that faults based on $M0$, $M1$ and $M2$ lead to the successful retrieval of the key. However, the same attack is not possible for faults according to $M3$ but not belonging to $M2$. It may be noted from the above facts that the proposed attack can capture all faults upto three bytes, as three bytes can never lie across all the four diagonals.

In the next section we present the implementation of AES targeted for the proposed fault attack.

4 The Attack Environment

In this section we outline the target design of AES which has been attacked by the proposed fault attack. Subsequently, we present the fault injection technique adopted for experimentally carrying out the proposed attack.

4.1 Target Implementation of AES

We have implemented an iterative architecture of AES using Verilog HDL. The implementation is exhibited in *figure 4*. The key and plaintext get loaded when the 'Load Key' and 'Load Plaintext' signals are asserted. The module first generates each round-key and stores it in the *Key Memory*. Once key generation is over the *Key Scheduler* asserts the *Kdone* signal. This happens once for each new key loaded. Then the round function module iterates *10* times before generating the ciphertext which is indicated by the *Output valid* signal. The controller consists of a counter which keeps track of the number of rounds and also selects the specific round key from the *Key Memory*. The design is downloaded on a Xilinx Spartan-3E XC3S500E platform and operates at a maximum frequency of 36 MHz.

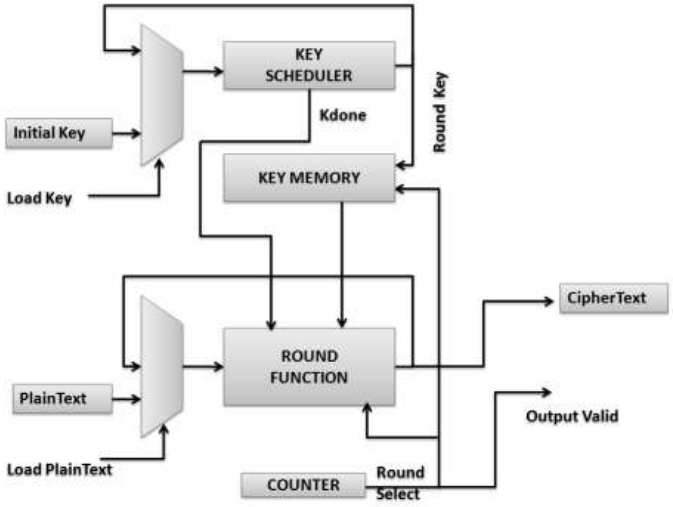

Fig. 4. Implementation of AES

4.2 Fault Injection Set-up

The effectiveness of fault attacks depend on the ability to induce faults inside the hardware. Methods proposed in [4] use optical beams to induce faults inside a circuit. However for our experimentation we use *clock glitching* as a means to induce internal faults. The advantage of such a method is in the lesser cost of the fault injection technique. The attacker thus manipulates only the clock signal. The set-up for such an injection is given in *figure 5*. The basic idea is to use a multiplexer to switch from the low frequency clock to a higher frequency clock at

the beginning of the eight round. The clocks are generated from an Agilent 33250A 80Mhz Function/Arbitrary Waveform Generator. So only the eighth round will run on the faster clock. The frequency of the faster clock has to be carefully adjusted from a signal generator. It should be slightly higher than the maximum frequency that the AES design supports. Our experiments show that there is a relation between the number of faults induced and the frequency of the faster clock. We have verified using Xilinx ChipScope Pro embedded logic analyzer that the number of faulty diagonals is directly proportional to the clock frequency.

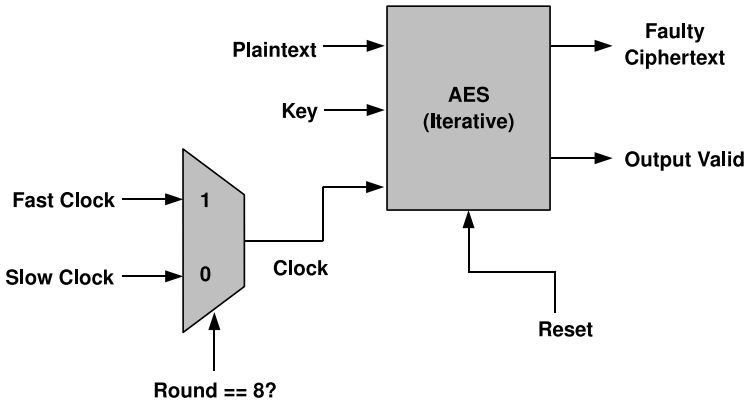

Fig. 5. Fault Injection Set-up

In the next section, we present fault attacks on AES according to the fault model $M0$. Hence we have non-zero faults in one of the four diagonals of the AES state matrix at the input of the eighth round.

5 The Proposed Attack according to Fault Model $M0$

We first show that faults which are confined to one diagonal are equivalent and can be used to retrieve the key using the same method.

5.1 Equivalence of faults in the same diagonal

Lets us first observe the propagation of a fault injected in diagonal D_0 through the round transformations from the input of the eighth round to the output of the ninth round. *Figure 6* shows the diffusion of a byte fault induced at the input of the eighth round. The difference of the state matrices of the two ciphers are depicted in the figure.

From *figure 6* we can see that fault induced spreads to an entire column at the end of the eighth round. By the end of the ninth round the fault spreads to the entire state matrix. However, the bytes of each the four columns of the the state matrix have some inter-relations among them. We exploit these relations to filter out keys and reduce the key space. We now show that inter-relations remain invariant for any type of faults that occur within a diagonal. For instance, if multiple bytes of the diagonal D_0 are faulty, the relations among the bytes after

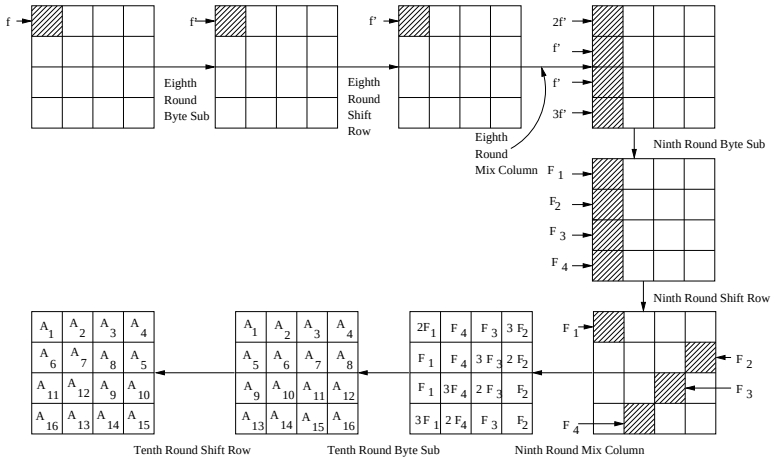

Fig. 6. Propagation of Fault Induced in the input of eighth round of AES

the ninth round MixColumn (as depicted in *figure 6*) will still hold. *Figure 7* shows some cases of fault induction in diagonal D_0. The faults vary in the number of bytes that are faulty in D_0 at the input of the eight round. We emphasize on the fact irrespective of the number or positions of bytes that are faulty in D_0, the fault is confined to the first column C_0 of the state matrix at the end of the eighth round. So the fault propagation in the ninth round for all these cases is similar and leads to the same byte inter-relations at the end of the ninth round.

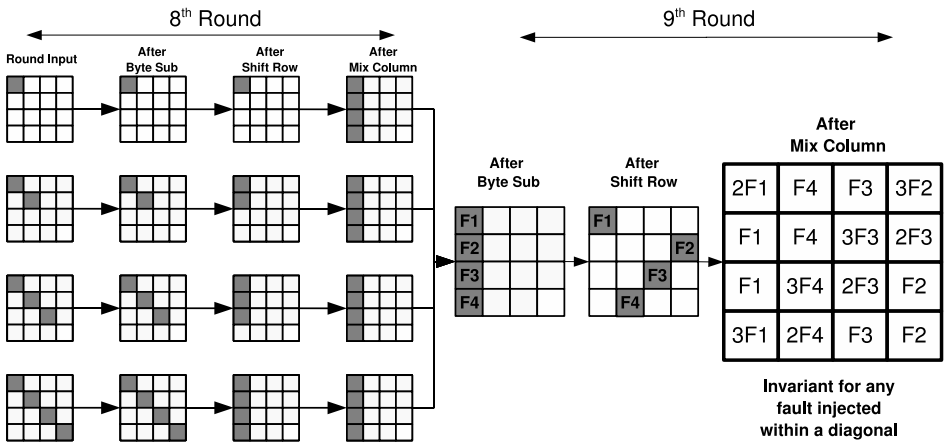

Fig. 7. Equivalence of different kinds of faults induced in diagonal D_0 at the input of eighth round of AES

In the last paragraph, we saw that any fault in D_0 results in C_0 getting affected. The happens due to the property of the ShiftRow operation. In general any fault at

the input of the eighth round in the i^{th} diagonal, $0 \leq i \leq 3$, leads to the i^{th} column being affected at end of the round. There are four diagonals and faults in each diagonal maps to four different byte inter-relations at the end of the ninth round. These relations are depicted in *figure 8*. These relations will remain unchanged for any combination of faults injected within a particular diagonal. Each of the four sets of relations in *figure 8* will be used to form key dependent equations. The next subsection explains the generations of the equations from the byte inter-relations.

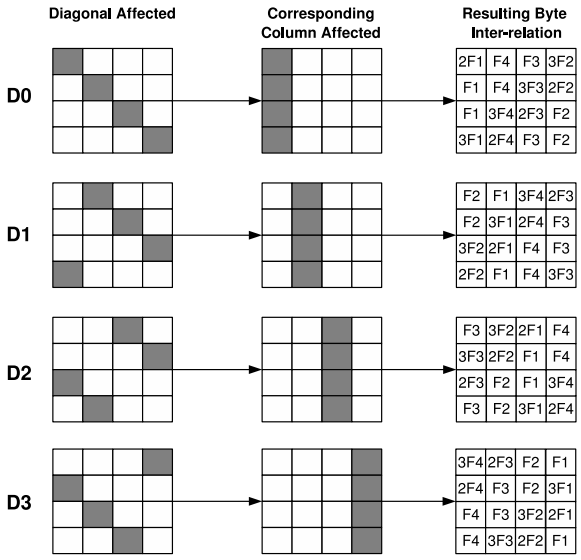

Fig. 8. Byte inter-relations at the end of ninth round corresponding to different diagonals being faulty

5.2 Equation generation from byte inter-relations

Our attack requires an attacker to obtain a single faulty ciphertext and one fault-free ciphertext. Let the fault-free ciphertext be denoted by CT while the faulty one be denoted by CT' 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.

Let us assume that the fault was injected somewhere in diagonal D_0. So the inter-relations after the ninth round MixColumn corresponding to D_0 given in *figure 8* are to be considered. 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.

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

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 obtains reduced solution spaces for the bytes K_{00}, K_{13}, K_{22} and K_{31} from the three equations. At first he guesses all values of K_{00} and K_{13} in the first equation and filters out all such key pairs that do not satisfy it. He now has a reduced key space for K_{13} which he uses for the second equation and for each such guess he guesses all values for K_{22} and proceeds similarly. The worst case complexity for one pass of the algorithm is 2^{16} and is independent of the value of the fault induced. The above system of equations is used to reduce the possibilities of 32 bits of the key. The average size of the reduced key space per 32 bits is roughly 2^8.

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:

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

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

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

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

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

Using all the equations above we get a total key space of 2^{32}. We get different sets of equations corresponding to a different diagonal being affected. So we will have four equation-sets each covering all possible faults within a specific diagonal. In the analysis above we assumed that the faulty diagonal was known to the attacker. This can be easily extended to a scenario where the fault location is not known to the attacker. In such a case the attacker guesses the faulty diagonal and repeats the attack four times. So the size of the key space will be $2^{32} \times 4 = 2^{34}$ which can be brute forced feasibly with present day computational power.

In the previous section, we have considered the scenario when the fault is restricted to one of the diagonals of the AES state matrix. In the next two sections, we describe attack methods when the fault spreads among the diagonals. First, we present the attack technique when at most two diagonals are affected by the fault induction.

6 The Proposed Attack according to Fault Model $M1$

In this section the fault induction takes place in the AES state matrix during the beginning of the eighth round according to fault model $M1$. Thus, at most two diagonals may get affected by the fault. It may be noted that since the fault model $M0$ is a special case of $M1$, the attack strategy for $M1$ also works if the fault is induced according to $M0$, that is if only one diagonal gets affected. Hence the attack based on model $M1$ is more general than that based on model $M0$.

6.1 Propagation of Faults when two diagonals are affected

In *figure 9*, we observe the propagation of faults when the diagonals, D_0 and D_1 are affected.

The fault is induced at the input of the eighth round. We note that for all possible faults in these two diagonals, the nature of faults in the state matrix at the input of the ninth round MixColumn is an invariant. This property is exploited to develop equations which are used to retrieve the correct key.

6.2 The equations based on the fault propagation

From *figure 9*, we note that for all possible faults corrupting the diagonals D_0 and D_1, the nature of the faults at the input of the ninth round MixColumn is an invariant and as depicted in the figure. Hence, the fault nature at the output of the ninth round MixColumn is also an invariant. We denote the fault values in the first column of the output of the ninth round MixColumn by a_0, a_1, a_2, a_3, where each a_i is a byte $0 \leq i \leq 3$.

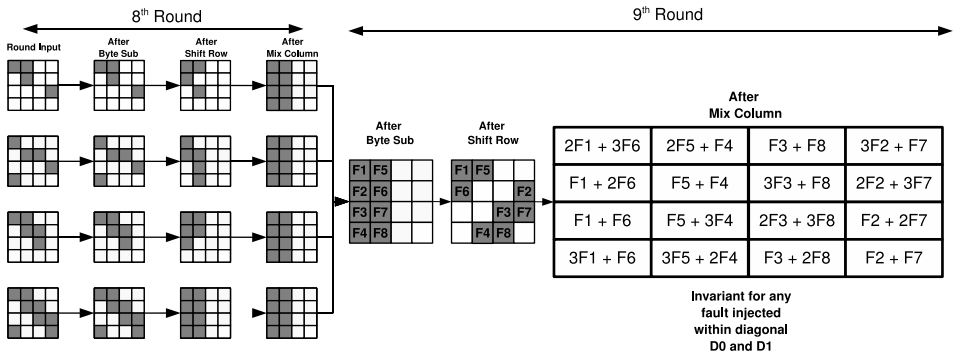

Fig. 9. Fault Propagation if diagonals D_0 are D_1 are affected

The following equations thus hold (refer *figure 9*):

$$
\begin{aligned}
a_0 &= 2F_1 + 3F_6 \\
a_1 &= F_1 + 2F_6 \\
a_2 &= F_1 + F_6 \\
a_3 &= 3F_1 + F_6
\end{aligned}
$$

Here it may be noted that the additions and multiplications are in $GF(2^8)$, with the reduction polynomial same as that in AES. Eliminating, F_1 and F_6 we obtain:

$$
\begin{aligned}
a_1 + a_3 &= a_0 \\
2a_1 + 3a_3 &= 7a_2
\end{aligned}
$$

We can express a_0, a_1, a_2, a_3 in terms of the fault free ciphertext (CT), faulty ciphertext (CT') and the tenth round key ($\mathbf{K_{10}}$) defined in *section* 5.2, as follows:

$$
\begin{aligned}
a_0 &= ISB(x_1 + K_{00}) + ISB(x_1 + A_1 + K_{00}) \\
a_1 &= ISB(x_8 + K_{13}) + ISB(x_8 + A_5 + K_{13}) \\
a_2 &= ISB(x_{11} + K_{22}) + ISB(x_{11} + A_9 + K_{22}) \\
a_3 &= ISB(x_{14} + K_{31}) + ISB(x_{14} + A_{13} + K_{31})
\end{aligned}
$$

Thus if we use the inter-relationships among the bytes a_i's $(0 \leq i \leq 3)$, and the above equations, we are able to reduce the possible key space of $K_{00}, K_{13}, K_{22}, K_{31}$. Similar to the solving mentioned in *section 5.2*, we know all the variables on the right hand side of the above equation from CT and CT'. We guess the bytes $K_{00}, K_{13}, K_{22}, K_{31}$, and maintain a list of probable keys which satisfy the inter-relationships among the bytes a_0, a_1, a_2 and a_3. We observe that using one faulty ciphertext, the attacker reduces the key space of $K_{00}, K_{13}, K_{22}, K_{31}$ from 2^{32} to 2^{16}. However with two faulty ciphertexts, an intersection of the solutions obtained from

the two pairs leaves an unique key with a probability of 0.99. For few other cases, using two faulty ciphertexts leave two or three keys. If a third faulty ciphertext is obtained, then the exact key is identified.

Proceeding similarly equations are derived for all other columns at the output of the ninth round MixColumn. For example the second column involves the key bytes $K_{01}, K_{10}, K_{23}, K_{32}$. The equations corresponding to the third column yields the key bytes $K_{02}, K_{11}, K_{20}, K_{33}$. Similarly, the fourth column leads to the key bytes $K_{03}, K_{12}, K_{21}, K_{30}$.

Depending on the combination of two diagonals affected out of the four diagonals, there are six such sets of equations. Hence if the attacker does not know which two diagonals are affected, he guesses the faulty diagonals and applies the corresponding set of equations to ascertain the key. Thus with two faulty ciphertexts the attacker reduces the AES key space to six possible keys, which he can easily brute force.

In the next section, we present an attack strategy if the fault gets spread to atmost three diagonals. This fault model, $M2$ thus covers the first two models of fault.

7 The Proposed Attack according to Fault Model $M2$

In *figure 10*, we observe the propagation of faults when the diagonals, D_0, D_1 and D_2 are affected.

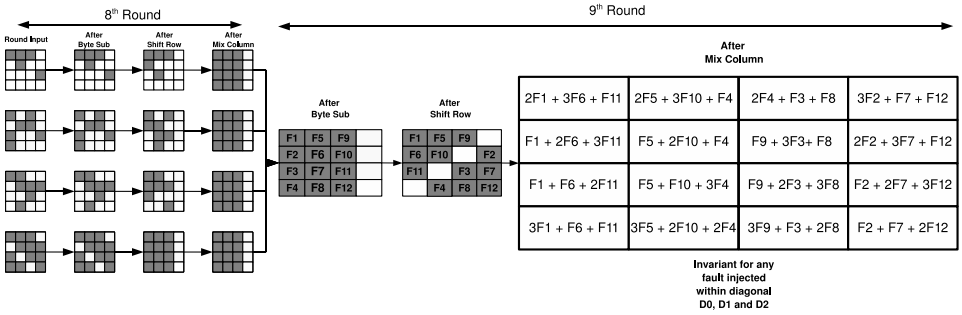

Fig. 10. Fault Propagation if diagonals D_0, D_1 and D_2 are affected

From *figure 10*, we note that for all possible faults corrupting the diagonals D_0, D_1 and D_2, the nature of the faults at the input of the ninth round MixColumn is an invariant. The fault nature at the output of the ninth round MixColumn is as seen in the figure, also an invariant. We denote the fault values in the first column of the output of the ninth round MixColumn by a_0, a_1, a_2, a_3, where each a_i is a byte $0 \leq i \leq 3$.

The following equations thus hold (refer *figure 9*):

$$a_0 = 2F_1 + 3F_6 + F_{11}$$
$$a_1 = F_1 + 2F_6 + 3F_{11}$$

$$a_2 = F_1 + F_6 + 2F_{11}$$
$$a_3 = 3F_1 + F_6 + F_{11}$$

Here also the additions and multiplications are in $GF(2^8)$, with the reduction polynomial same as that in AES. Eliminating, F_1, F_6 and F_{11} we obtain:

$$11a_0 + 13a_1 = 9a_2 + 14a_3$$

As before in case of faults modeled by $M1$, we can express a_0, a_1, a_2, a_3 in terms of the fault free ciphertext (CT), faulty ciphertext (CT') and the tenth round key ($\mathbf{K_{10}}$) defined in *section* 5.2, as follows:

$$a_0 = ISB(x_1 + K_{00}) + ISB(x_1 + A_1 + K_{00})$$
$$a_1 = ISB(x_8 + K_{13}) + ISB(x_8 + A_5 + K_{13})$$
$$a_2 = ISB(x_{11} + K_{22}) + ISB(x_{11} + A_9 + K_{22})$$
$$a_3 = ISB(x_{14} + K_{31}) + ISB(x_{14} + A_{13} + K_{31})$$

Thus if we use the inter-relationship among the bytes a_i's $(0 \leq i \leq 3)$, and the above equations, we are able to reduce the possible key space of $K_{00}, K_{13}, K_{22}, K_{31}$. Similar to the solving mentioned in *section 5.2*, we know all the variables on the right hand side of the above equation from CT and CT'. We guess the bytes $K_{00}, K_{13}, K_{22}, K_{31}$, and maintain a list of probable keys which satisfy the inter-relationship among the bytes a_0, a_1, a_2 and a_3. We observe that using one faulty ciphertext, the attacker reduces the key space of $K_{00}, K_{13}, K_{22}, K_{31}$ from 2^{32} to 2^{24}. However with four faulty ciphertexts, an intersection of the solutions obtained from the two pairs leaves an unique key.

It may be noted that when the faults occur according to the model $M3$, that is all the four diagonals are affected, the proposed method fails to obtain the key. However, as we show in the next section, when we clock the design at a slightly higher frequency than the maximum permissible frequency the faults occur according to the models $M0, M1$ and $M2$ with a much larger probability.

8 Experimental Results

In this section, we furnish experimental results on the proposed fault attacks. The attacks have been performed on a hardware implementation of AES on a Xilinx Spartan-3E platform. The design operates at a maximum frequency of 36 MHz, when the design operates as expected. The attack strategy is to switch the clock and pass the faster clock to drive the circuit at the time when the 8^{th} round encryption commences. In Table 1, we summarize the nature of the induced faults with respect to the frequency of the faster clock. It may be observed that when the faster clock has a frequency of 36 MHz then there is no faulty state. Hence we obtain the original fault free ciphertext. Gradually we increase the frequency of the faster clock in steps of 0.1 MHz and use the ChipScope 7.1 analyzer to observe the faulty bytes in the state matrix of the AES hardware running on the FPGA,

when the eighth round starts. The number of attempts to inject a fault at each step is 512. This is the maximum possible number of samples that Chipscope 7.1 can handle.

Table 1. Fault classification with clock frequency

Clock Frequency (MHz)	No Fault	Model 0 (M0)	Model 1 (M1)	Model 2 (M2)	Model 3 (M3)
36.0	512	0	0	0	0
36.1	512	0	0	0	0
36.2	512	0	0	0	0
36.3	510	2	0	0	0
36.4	511	1	0	0	0
36.5	508	4	0	0	0
36.6	504	8	0	0	0
36.7	507	5	0	0	0
36.8	490	22	0	0	0
36.9	489	23	0	0	0
37.0	419	79	14	0	0
37.1	448	60	4	0	0
37.2	434	64	13	1	0
37.3	403	94	15	0	0
37.4	408	99	5	0	0
37.5	248	226	38	0	0
37.6	214	205	84	9	0
37.7	128	205	122	57	0
37.8	76	180	133	123	0
37.9	20	122	145	225	0
38.0	158	191	129	34	0
38.1	27	116	185	185	0
38.2	40	127	198	147	0
38.3	26	69	155	257	5
38.4	17	62	137	254	42
38.5	0	20	68	361	63
38.6	0	0	16	319	177
38.7	0	2	20	293	197
38.8	0	1	8	290	213
38.9	0	11	42	368	91
39.0	15	59	107	308	23
39.1	0	2	12	197	301
39.2	0	5	26	339	142
39.3	0	3	11	285	213
39.4	0	0	0	134	378
39.5	0	0	6	138	368
39.6	0	0	0	150	362
39.7	0	0	0	21	491
39.8	0	0	0	18	494
39.9	0	0	0	14	498
40.0	0	0	0	0	512

We note that at 36.3 MHz faults start to appear for the first time. Till 37.5 MHz faults under model $M0$ dominate. From 38.1 to 38.4 MHz the faults under the model $M0, M1$ and $M2$ occur simultaneously with similar frequency. $M2$ starts to dominate when the clock frequency reaches 38.5 MHz and continues to do so till 39.4 MHz; during this period $M0$ and $M1$ faults get reduced. After this from 39.5 MHz $M3$ faults start to dominate, while $M2$ still appears but $M0$ and $M1$ faults vanish.

So, we observe that if the clock changes are confined between 36 to 37.4 MHz, then $M0$ faults occur primarily and in few cases $M1$ faults. Since as we have previously shown, the fault attacks based on the model $M0$ is the strongest and most powerful, we can indeed break the AES key using a single fault injection in 400 seconds (refer *section 5*).

9 Comparison with Existing Works and Experimental Results

There have been considerable number of works on the subject. To the best of our knowledge, no reported works present fault attacks on real life implementations. In our paper, we have not only shown our attack on an FPGA implementation of AES but also demonstrated that fault injection via clock glitching can be a dangerous tool to the adversary.

In our approach, we have developed our attack based on multi-byte fault models, compared to single byte fault models existing in literature except [9]. In [9], the authors use an attack based on four byte faults where he needs about 1500 faulty ciphertexts. However they do not outline any practical method through which such a large number of faulty ciphertexts can be obtained.

Finally *table 2* compares the existing fault based attacks on AES with our work. The results show that though our work is based on multi-byte fault modeling, still it is able to attack AES with least number of faulty ciphertexts.

10 Conclusions

A new fault attack on AES based on multiple byte faults has been presented. Three attack strategies have been developed exploiting the disturbance in the diagonals of the AES state matrix at the input of the eighth round. The fault has been injected by the simple mechanism of clock glitching. For the first time in existing literature, the fault attack has been shown on actual hardware implementation. The experiments confirm that the fault attack outperforms existing works with respect to practical fault modeling and number of faulty ciphertexts needed.

References

1. D. Boneh, R.A. DeMillo and R.J. Lipton, "On the Importance of checking cryptographic Protocols for Faults," in *Eurocrypt 1997*. 1997, LNCS 1233.
2. D. Boneh, R.A. DeMillo and R.J. Lipton, "On the Importance of Eliminating Errors in Cryptographic Computations," *Journal of Cryptology*, pp. 101–120, 2001.
3. E. Biham and A. Shamir, "Differential Fault Analysis of Secret Key Cryptosystems," in *Advances in Cryptology, Crypto 1997*. 1997, LNCS 1294.

Table 2. Comparison of Existing Fault Attacks on AES exploiting properties of the encryption function

Reference	Fault Model	Fault Location	No. of Faulty Encryptions
[6]	Force 1 bit to 0	Chosen	128
[6]	Implementation Dependent	Chosen	256
[10, 11]	Disturb 1 byte	Key Scheduling	7
[5]	i) Switch 1 bit ii) Disturb 1 byte	i) Any bit of chosen bytes ii) Anywhere among 4 bytes	i) ≈ 50 ii) ≈ 250
[7]	Disturb 1 byte	Anywhere between last two MixColumns	≈ 40
[8]	Disturb 1 byte	Anywhere between 7^{th} round and 8^{th} round MixColumn	2
[12]	Disturb 1 byte	Anywhere between 7^{th} round MixColumn and 8^{th} round MixColumn	2
[9]	4 bytes: i)One byte undisturbed ii)All 4 bytes disturbed	Input of ninth Round	i) 6 ii) 1500
This paper	Multibytes: i) M0 ii) M1 iii) M2	Anywhere between 7^{th} round MixColumn and 8^{th} round MixColumn	i) 1 ii) 2 iii) 4

4. S. Skorobogatov and R. Anderson, "Optical Fault Induction Attacks," in *CHES 2002*. 2002, LNCS 2523.
5. C. Giraud, "DFA on AES," Cryptology ePrint Archive, Report 2003/008, 2003.
6. J. Blomer and J. P. Seifert, "Fault Based Cryptanalysis of the Advanced Encryption Standard (AES)," in *FC 2003*. 2003, pp. 162–181, LNCS 2742.
7. P. Dusart, G. Letourneux and O. Vivolo, "Differential Fault Analysis on A.E.S.," http://eprint.iacr.org/2003/010.
8. G. Piret and J. J. Quisquater, "A Differential Fault Attack Technique against SPN Structures, with Application to the AES and Khazad," in *CHES 2003*. 2003, pp. 77–88, LNCS 2779.
9. M. Salmasizadeh A. Moradi, T. M. Shalmani, "A generalized method of differential fault attack against aes cryptosystem.," in *CHES*, 2006, pp. 91–100.
10. J.Takahashi, T.Fukunaga and K.Yamakoshi, "DFA mechanism on the AES schedule," in *Proceedings of* 4^{th} *International Workshop on Fault Detection and Tolerance in Cryptography, FDTC*, 2007, pp. 62–72.
11. J.Takahashi, T.Fukunaga, "Differential Fault Analysis on the AES Key Schedule," http://eprint.iacr.org/2007/480, 2007.
12. Debdeep Mukhopadhyay, "An improved fault based attack of the advanced encryption standard," in *AFRICACRYPT*, 2009, pp. 421–434.
13. J. Daemen and V. Rijmen, *The Design of Rijndael*, Springer-Verlag, 2002.