Transient-Steady Effect Attack on Block Ciphers

Yanting Ren[1,2], An Wang*[1,2], and Liji Wu*[1,2]

[1] Tsinghua National Laboratory for Information Science and Technology (TNList),
Beijing, China
[2] Institute of Microelectronics, Tsinghua University, Beijing, China
ryt10@mails.tsinghua.edu.cn
{wangan1,lijiwu}@mail.tsinghua.edu.cn

Abstract. A new Transient-Steady Effect attack on block ciphers called TSE attack is presented in this paper. The concept of transient-steady effect denotes the phenomenon that the output of a combinational circuit keeps a temporal value for a while before it finally switches to the correct value. Unlike most existing fault attacks, our attack does not need a large amount of encryptions to build a statistical model. By injecting a clock glitch to capture the temporal value caused by transient-steady effect, attackers can obtain the information of key from faulty outputs directly. This work shows that AES implementations, which have transient-steady property, are vulnerable to our attack. Experiments are successfully conducted on two kinds of unmasked S-boxes and one kind of masked S-box implemented in serial with FPGA board. After a moderate pre-computation, we need only 1 encryption to recover a key byte of the unmasked S-boxes, and 20 encryptions to recover a key byte of the masked S-box. Furthermore, we investigate the key recover method for parallel unmasked implementation, and discuss a possible attack scenario which may deem WDDL-AES insecure.

1 Introduction

Side-channel attacks have drawn much attention since being proposed by Kocher et al. [1]. Up to now, many attack methods have been introduced to analyze side-channel information leaked by cryptographic devices, such as correlation power analysis [2, 3], template [4], collision [5, 6], mutual information [7] and fault attack. Differential Fault Analysis (DFA) [8] is one of the most well-known *fault attacks*. In DFA attack, the ciphertext with fault injected during executing is called a faulty output. The key is recovered from correct outputs and corresponding faulty outputs based on a fault model.

In CHES 2010, Li et al. [9] proposed the Fault Sensitivity Analysis (FSA) based on the fact that the critical paths of some Advanced Encryption Standard (AES) S-box combinational circuits are data dependent. However, a large number of encryptions is needed in an FSA attack. The adversary has to encrypt every plaintext for many times and shorten the glitch cycle gradually, in order to

* Corresponding authors.

obtain the critical frequency (the fault sensitivity) at which faulty outputs begin to appear. In 2011, Moradi et al. extended FSA to masked AES implementation by combining it with collision attack [10]. Their attack is carried out at a fixed glitch frequency, but it still requires a lot of encryptions to extract the distribution of faulty ciphertexts. In addition, as correlation-based methods, both of these attacks need to enumerate the values of the plaintext, which increase the total number of encryptions. In 2012, Li et al. presented the Clockwise Collision Fault Sensitivity Analysis (CC-FSA) attack on unmasked AES [11]. They pointed out: in an iterative AES implementation, if inputs of two consecutive cycles are identical, the setup time of the second cycle is extremely short, because there are almost no toggles in the combinational circuit. Soon after that, Wang et al. proposed an improved clockwise collision attack called Fault Rate Analysis (FRA) and broke a masked serial AES S-box implementation [12]. The two methods are carried out at fixed glitch frequency, but they both suffer from the inefficiency of detecting clockwise collisions, and need a large number of encryptions.

Our contribution. In this paper, we propose a new fault attack based on Transient-Steady Effect (TSE attack). Transient-steady effect denotes the phenomenon that the output of a gate turns to a temporal value and keeps steady for a while before it switches to the final steady state. We analyze the circuits of several AES S-box implementations and find out that the path of the key is usually much shorter than other signals. Therefore, soon after the rising edge of the clock, the output turns to a value that is computed from the key in current clock cycle and other data in the last cycle. By injecting a clock glitch, we can capture the temporal value as a faulty output to recover the key. We propose several fault models based on the transient-steady effect and verify TSE attack on both unmasked and masked AES S-boxes. Our attack has the following features:

- In comparison to the existing works, TSE attack needs less encryptions in the attack stage. We only need to sweep the frequency of clock glitch for one time in the pre-computation stage. Then the attack stage can be conducted at a fixed frequency. Furthermore, the key can be recovered directly from faulty outputs, so we do not need a large amount of encryptions to build a statistical model.
- TSE attack is verified to be effective to a masked implementation of AES based on tower field. Other masking techniques with obvious transient-steady effect may also be insecure under this attack.
- TSE attack can break the protection strategy that changes the plaintext for every encryption, because correct outputs are not necessary.

Organization. We organize the rest part of this paper as follows. Related preliminaries are introduced in Sect. 2. The basic idea and attack scenarios are detailed in Sect. 3. We present experimental results and efficiency comparison in Sect. 4. Then we discuss about the application of our attack on parallel AES implementation and WDDL-AES in Sect. 5. Conclusions are given in Sect. 6.

2 Preliminaries

2.1 AES S-box and Masking

AES is a widely used symmetric cryptographic algorithm, which is composed of 10 rounds, and each round includes 16 S-boxes. When the area or the power consumption is limited, serial implementation of the algorithm is preferred. For example, a circuit with 4 S-boxes can accomplish one round in 4 cycles, and each S-box is reused for 4 times [13, 14]. Many low-power and low-area S-boxes have been proposed. For example, Morioka et al. gave a low-power approach [15], and Canright proposed a low-area approach based on tower-field [16].

Masking is a regular countermeasure against power analysis. Mask values randomize sensitive intermediate values and minimize the dependency between data and power consumption. S-box is the only nonlinear operation in AES algorithm, and many masking schemes have been proposed for it, such as the approach based on tower-field [17].

As shown in Fig. 1, a standard masked S-box has one masked output and three inputs: the masked value x_m, the input mask m and the output mask w. We do not show the output mask as an output of S-box in Fig. 1, but it is also recorded for the next round.

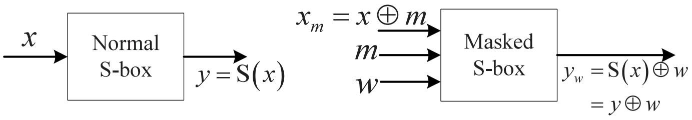

Fig. 1. Unmasked S-box (left) and masked S-box (right)

2.2 Fault-Based Clockwise Collision Analysis

CC-FSA attack was presented by Li et al. in 2012 [11]. The attack is based on the fact that if the inputs of a circuit do not change in two consecutive clock cycles, there will be almost no toggles in the second cycle. It is called a clockwise collision, and the setup time of the second clock cycle will be extremely short. They let the target circuit work normally in the first cycle, and insert a clock glitch to create a very short second cycle. If the output is correct, a clockwise collision will be detected.

3 Transient-Steady Effect Attack

In most standard logic designs, the lengths of data paths in combinational circuits are usually different. If we focus on a gate, we can see that after the rising edge of the clock, the inputs of the gate do not necessarily arrive simultaneously.

For example, we can assume the path delay of signal a is shorter than that of signal b. Hence, after the switch of a and before the arrival of b, the output turns to a transient illegal value. When the difference of the propagation delay between the two signals is large enough, the output stays at the illegal value for a while before all the propagations are done correctly. This is called the transient-steady effect. Related works have proved that transient-steady effect can lead to a data-dependent power consumption and leak the secret information *indirectly* [18–22]. However, in this paper we show that the temporal value caused by transient-steady effect can be captured and used to retrieve secret information *directly.*

Based on the transient-steady effect, we propose the TSE attack: We let the target circuit compute normally in the first cycle, and inject a clock glitch to create a very short second cycle. The normal output of the first clock cycle is computed from the short-path data and the long-path data in the first cycle. The faulty output of the second cycle is computed from the short-path data in the second cycle and the long-path data in the *first* cycle. By combining the outputs of the two consecutive clock cycles, we can recover information of the short-path data.

3.1 Basic Idea

Without loss of generality, we first look at a combinational circuit which computes the output with two inputs, e.g. X and Y. Their propagation delays are denoted as t_X and t_Y. The output, denoted as $Z = f(X, Y)$, is captured by a register. As shown in Fig. 2, we assume the propagation delays of the two inputs are different, for example, $t_Y \gg t_X$. Focusing on two specific clock cycles, we denote the inputs in the first cycle as X_1 and Y_1, and the inputs in the second cycle as X_2 and Y_2. After the rising edge of the second clock, the effects of X_2 and Y_2 begin to propagate along the two data paths, like two ripples with different speeds. After a period of time t ($t_Y > t > t_X$), X_2 has impacted all the gates in the circuit, but the ripple of Y_2 has not arrived at the output, so the output Z turns to a value of $f(X_2, Y_1)$. We assume the difference of path delays, denoted as $d = t_Y - t_X$, is large enough. Hence, the temporal value $f(X_2, Y_1)$ keeps steady at the output for a while. As presented in Fig. 3, if a glitch is injected to make the length of the second cycle within the range from t_X to t_Y, the temporal value can be stored in RegZ.

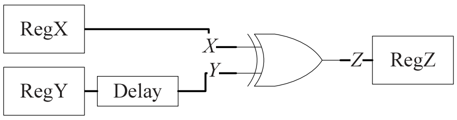

Fig. 2. An example of circuit with different propagation delays

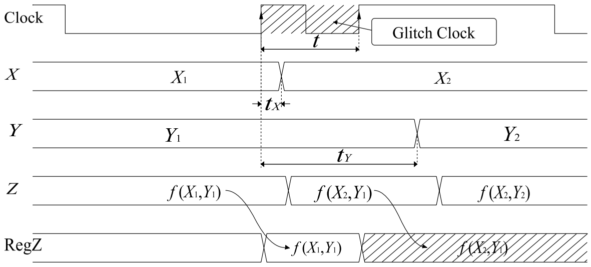

Fig. 3. The sequence diagram with clock glitch

3.2 Attack Scenario on Unmasked S-Box

First, we analyze the unmasked S-box. As mentioned in Sect. 2.1, we consider the serial implementation, where the inputs of different S-boxes are fed to the same combinational circuit in consecutive clock cycles. We assume the circuit executes the S-box operations of the final AES round consecutively. As in Fig. 4, there are two data paths in the circuit: The longer data path is marked with dashed red arrow, and its delay is denoted as t_y. The shorter one is marked with solid green arrow, and its delay is denoted as t_k.

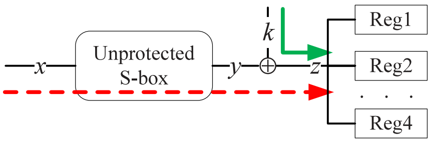

Fig. 4. The data path of unmasked S-box in the final AES round

As shown in Fig. 5, the output of first cycle $z_1 = \mathrm{S}(x_1) \oplus k_1$ is stored in register Reg1 at the rising edge of the second cycle. After the duration time of t_k, k_2 propagates through the exclusive-or gate and the output switches to a temporal value $\tilde{z}_2 = \mathrm{S}(x_1) \oplus k_2$. The temporal value stays for the duration time of $t_y - t_k$. If we inject a clock glitch after the first clock cycle, and make sure the length of the glitch cycle satisfies $t_y > t_g > t_k$, $\tilde{z}_2$ can be stored in Reg2. With z_1 and $\tilde{z}_2$, we can compute

$$\begin{aligned} z_1 \oplus \tilde{z}_2 &= \mathrm{S}(x_1) \oplus k_1 \oplus \mathrm{S}(x_1) \oplus k_2 \\ &= k_1 \oplus k_2 \\ &= \mathit{\Delta} k_{1,2} \ . \end{aligned} \tag{1}$$

Since the data paths' delays are unknown to us, we conduct the TSE attack practically in the following steps:

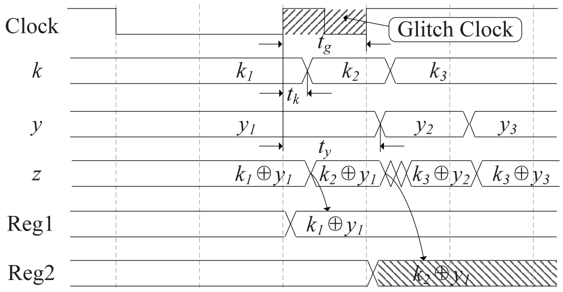

Fig. 5. Sequence diagram of unmasked S-box with clock glitch

- **Step 1**: Sweep the frequency of clock glitch, i.e. change the length of the clock glitch cycle gradually. At each frequency point, do encryptions with fixed x_1 and random x_2 for N_{pre} times and make a record for the faulty outputs.
- **Step 2**: Find out the range of glitch frequency in which the faulty outputs keep stable. According to the analysis above, with fixed x_1, the faulty output $\tilde{z}_2 = \mathrm{S}(x_1) \oplus k_2$ should be a constant value independent of x_2.
- **Step 3**: Choose a proper glitch frequency in the range detected in Step 2.
- **Step 4**: Do encryptions for N_{attack} times at the chosen glitch frequency, record z_1, $\tilde{z}_2$, and compute the attack result $z_1 \oplus \tilde{z}_2$ for every encryption.
- **Step 5**: Among all the attack results $z_1 \oplus \tilde{z}_2$, choose the value which has the highest occurrence rate as the value of $\Delta k_{1,2}$.
- **Step 6**: Repeat Step 4 to 5 for other clock cycles to recover $\Delta k_{2,3}$, $\Delta k_{3,4}$, etc.

We call Step 1 to Step 3 as the pre-computation stage, which only needs to be done one time for a target circuit. Step 4 to 6, called the attack stage, can be done at a fixed frequency.

Note that there is no specific requirement on the unmasked S-box's structure, as long as the shortest data path delay of the S-box is sufficiently long.

3.3 Attack Scenario on Masked S-Box

Masked S-box has three inputs: the masked value $x_m = x \oplus m$, the input mask m and the output mask w. The output is masked with w: $y_w = y \oplus w = \mathrm{S}(x) \oplus w$. Since w is used to mask the output of S-box, its data path is usually shorter than x_m and m [12]. Here we focus on the masked S-box based on tower field [17]. As shown in Fig. 6, the data path of x_m and m, which is marked with dashed red arrow, is much longer than those of others. Similar to the unmasked S-boxes, the normal output of the first clock is captured in Reg1:

$$\begin{aligned} z_1 &= y_{w_1} \oplus k_1 \oplus w_1 \\ &= \mathrm{S}(x_1) \oplus w_1 \oplus k_1 \oplus w_1 \\ &= \mathrm{S}(x_1) \oplus k_1 \ . \end{aligned}$$

Here y_{w_1} represents the masked S-box output of the first clock cycle. We inject a glitch after the first clock cycle. If the length of the glitch cycle is shorter than the delay of x_m and m, and longer than that of w and k, the temporal output $\tilde{z}_2$ can be captured in Reg2:

$$\begin{aligned} \tilde{z}_2 &= \tilde{y}_{w_2} \oplus k_2 \oplus w_2 \\ &= \mathrm{S}(x_1) \oplus w_2 \oplus k_2 \oplus w_2 \\ &= \mathrm{S}(x_1) \oplus k_2 \ . \end{aligned}$$

By combining z_1 and $\tilde{z}_2$, we have the result similar to unmasked S-box:

$$\begin{aligned} z_1 \oplus \tilde{z}_2 &= \mathrm{S}(x_1) \oplus k_1 \oplus \mathrm{S}(x_1) \oplus k_2 \\ &= k_1 \oplus k_2 \\ &= \Delta k_{1,2} \ . \end{aligned} \tag{2}$$

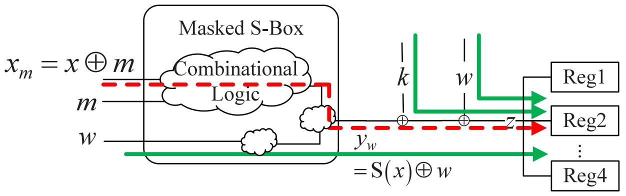

Fig. 6. Data path of masked S-box in the final AES round

Note that the attack described in this section is only applicable if the final unmasking is done within the same clock cycle as the final key addition.

4 Experiments and Efficiency

We verify the proposed TSE attack on two unmasked S-boxes [15, 16] and one masked S-box [17] which are implemented on DE2-115 FPGA board with Altera Cyclone IV EP4CE115. We use a RIGOL DG4102 function generator as the input clock. The circuit diagram of attack on masked S-box is shown in Fig. 7, and the setup for unmasked S-box is similar. A PLL is employed in the glitch generator to create clock for the control module and the circuit under attack. The PLL outputs two clock signals. The low frequency signal is used as the normal clock, and high frequency signal is used as the clock glitch. A clock multiplexer is used to switch between the normal and clock glitch. The outputs of two consecutive clock cycles, z_1 and $\tilde{z}_2$, are stored in the registers Reg1 and Reg2 respectively, and the attack result $z_1 \oplus \tilde{z}_2$ is stored in RAM. As presented in Sect. 3, if no fault is injected, the attack result should be $\mathrm{S}(x_1) \oplus k_1 \oplus \mathrm{S}(x_2) \oplus k_2$. If the attack succeeds, it should be $k_1 \oplus k_2$.

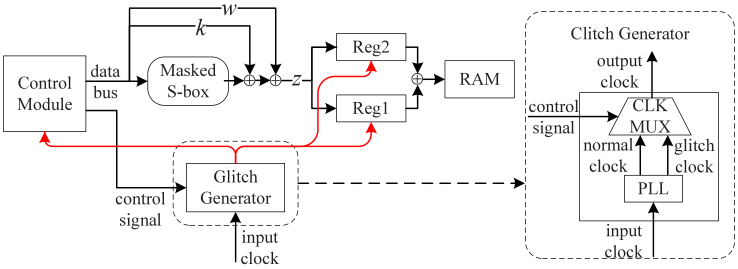

Fig. 7. Experimental circuit diagram of masked S-box

4.1 Experiment on Unmasked S-Box A

The S-box we attack in this section is presented in [15], We set the key bytes as k_1 = 0xE2 and k_2 = 0x19. If the attack succeeds, the result stored in RAM should be $\Delta k_{1,2}$ = 0xFB.

Following the steps of TSE attack detailed in Sect. 3.2, we first do the precomputation stage for S-box A: We choose 80 frequency points from 64MHz to 480MHz to sweep the glitch frequency. At every frequency point, the experiment is conducted as follows: We fix the value of x_1 as 0x31, and enumerate the value of x_2. For each x_2, we encrypt it for 256 times. Therefore, at each frequency point, 65536 attack results are stored. As shown in Fig. 8, we count the occurrence rates for all the possible values of $\Delta k_{1,2}$ at every frequency point. Within the range from 360MHz to 430MHz, the occurrence rate of the correct value of $\Delta k_{1,2}$ rises up to nearly 100%. Obviously, the range is suitable for the TSE attack.

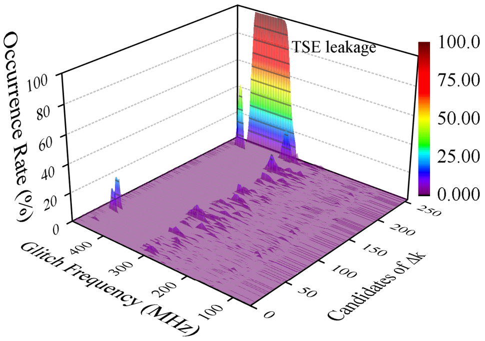

Fig. 8. Results of sweeping glitch frequency for S-box A

We also verify that the proper frequency range is valid for all the possible inputs of the S-box. The 256 occurrence rate curves of correct $\Delta k_{1,2}$ corresponding to all the 256 values of x_2 are plotted in Fig. 9. Even though the critical timing delay of the S-box depends on the Hamming weight of the inputs [9], we can conclude from Fig. 9 that there is a proper frequency range, i.e. from 360MHz to 430MHz, for all the possible inputs.

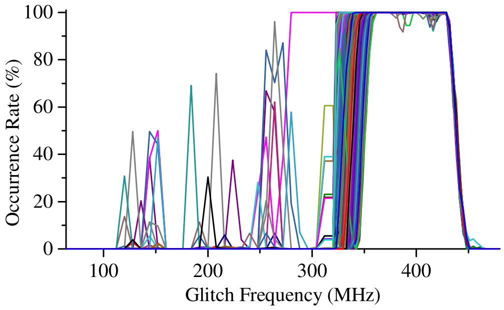

Fig. 9. The 256 occurrence rate curves of correct $\varDelta k_{1,2}$ corresponding to 256 values of x_2 for S-box A.

At the attack stage, TSE attack can be done at any glitch frequency within the range from 360MHz to 430MHz. However, to illustrate the result more clearly, we conduct attacks on all the frequency points, and the success rate of attack is in Fig. 10. Increasing the number of encryptions used for each attack, i.e. N_{attack}, can slightly widen the range of proper frequency. Even with only 1 encryption, our attack can achieve a success rate of nearly 100%.

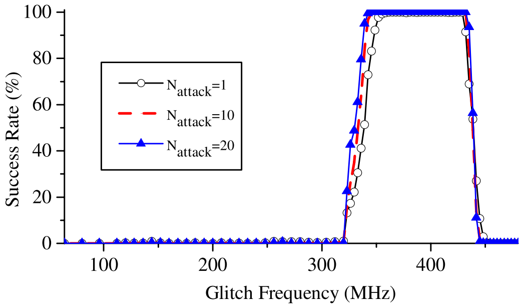

Fig. 10. Success rate *vs.* frequency with different N_{attack} for S-box A

4.2 Experiment on Unmasked S-Box B

We carry out experiments on a very compact unmasked S-box [16] in the same way of Sect. 4.1. The results are shown in Fig. 11, 12, 13.

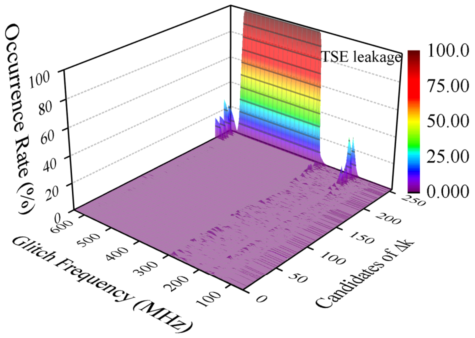

Fig. 11. Results of sweeping glitch frequency for S-box B

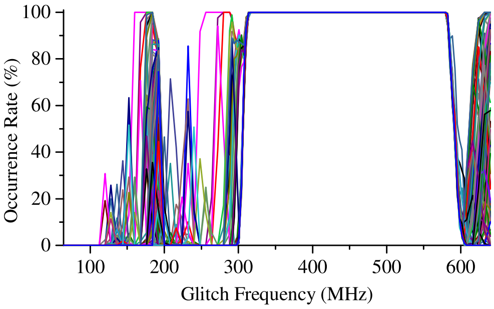

Fig. 12. The 256 occurrence rate curves of correct $\Delta k_{1,2}$ corresponding to 256 values of x_2 for S-box B.

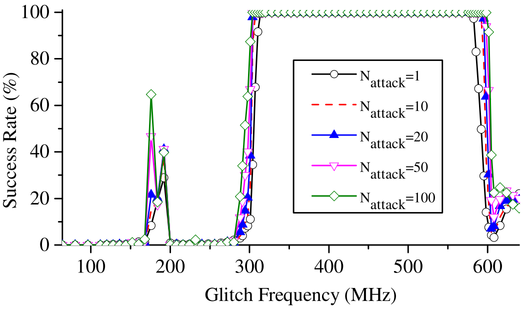

Fig. 13. Success rate *vs.* frequency with different N_{attack} for S-box B

As shown in Fig. 11, the range of proper glitch frequency is from 320MHz to 580MHz. From Fig. 12, we can see that inputs of the S-box have little effect on the frequency range.

As shown in Fig. 13, a small peak appears at about 176MHz when 50 or more encryptions are used. The frequency of the peak is much lower than the range from 320MHz to 580MHz, so it may be easier to inject glitch at this frequency. However, the peak is very narrow, so the width of the glitch has to be very accurate to mount a success TSE attack. Moreover, as shown in Fig. 11, there are several peaks at 176MHz corresponding to different attack results, for example, 0xFB (correct value of $\Delta k_{1,2}$) and 0xF3. Since we always choose the value which has the largest occurrence rate as $\Delta k_{1,2}$, the probability of choosing 0xF3 is also very high. Hence, by injecting glitch at 176MHz, we may not be able to recover $\Delta k_{1,2}$ directly, but the key space can still be reduced significantly.

4.3 Experiment on Masked S-Box C

S-box C is the masked version of S-box B [17]. We set the inputs of S-box as $x_1 = $ 0x9D, $x_2 = $ 0xE6, and the key bytes as $k_1 = $ 0x3F and $k_2 = $ 0x58. With no fault injected, the attack result should be $\mathrm{S}(x_1) \oplus k_1 \oplus \mathrm{S}(x_2) \oplus k_2 = $ 0xB7. If the attack succeeds, the result should be $\Delta k_{1,2} = $ 0x67.

We choose 72 frequency points between 50MHz to 200MHz for frequency sweeping. At each frequency point, we encrypt the plaintext with random masks m and w for 65536 times. As shown in Fig. 14, when the glitch frequency is lower than 75MHz, the occurrence rate of value 0xB7 is 100%, namely no fault occurs. When the glitch frequency gets higher, there is only one peak higher than 60%, which corresponds to the value of $\Delta k_{1,2} = $ 0x67. The feasible frequency range for this S-box is from 145MHz to 150MHz.

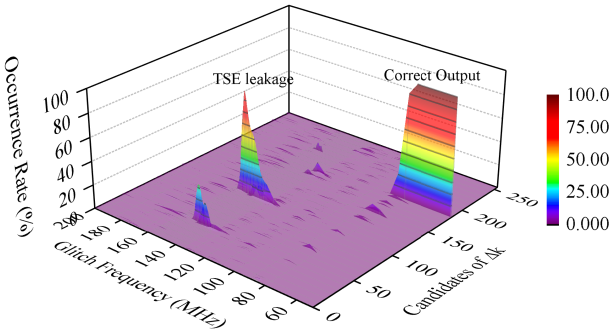

Fig. 14. Results of sweeping glitch frequency for S-box C

The results of the attack stage are shown in Fig. 15. With only one encryption for each attack, the success rate of TSE attack reaches to 90% at the frequency of

150MHz. With more encryptions, the range of proper glitch frequency is widened obviously. With more than 20 encryptions, our attack can have a success rate higher than 90% within the frequency range from 142MHz to 152MHz.

It is worth noting that the proper glitch frequency for attacking the masked S-box is much lower than unmasked S-box. That is because, as a countermeasure against side-channel attacks, masking usually results in a longer data path delay for x_m and m, which turns out to be vulnerable to our attack.

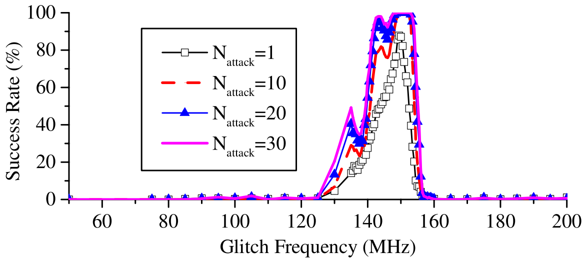

Fig. 15. Success rate *vs.* frequency with different N_{attack} for S-box C

4.4 Efficiency Comparison

We compare TSE attack with related fault based attacks on AES S-box in Table 1. The comparison is based on the effort to disclose 8-bit information of the key. Our attack has obvious advantages in the memory space, the offline complexity and the number of encryptions needed to recover a key byte. Here C_{ρ_n} means the complexity of calculating the correlation coefficient of two n-sample vectors. Previous works [9, 10, 12] need many encryptions to obtain the statistical data or to build models of the target circuit's behavior in each attack. However, our attack puts most workload into the pre-computation stage, i.e. sweeps the glitch frequency for only one time to find a proper frequency range. Then, in the attack stage, it is feasible and efficient to obtain key-related information from the faulty output directly.

The last row in Table 1 denotes the number of encryptions needed in pre-computation stage, the data in this row is estimated. Experienced attackers usually do not need so many encryptions.

5 Further Discussion

5.1 Key Recovery for Parallel AES Implementation

In some AES implementations, 16 S-boxes are implemented in parallel to achieve high throughput [23]. In such implementations, the transient-steady effect still exists. However, the temporal value turns out to be related with two adjacent

Table 1. Comparison with three fault based attacks

Method	FSA [9]	CTC [10]	FRA [12]	TSE Attack	TSE Attack
Target S-box	Unmasked	Masked	Masked	Unmasked	Masked
Num of Enc	840	10^6	8×10^4	1	20
Space (bytes)	120	2048	80	1	20
Offline Complexity	$256C_{\rho_7}$	$256C_{\rho_{256}}$	$1C_{div}$	≈ 0	≈ 0
Num of Pre-Enc	0	0	0	4×10^4	4×10^4

rounds, rather than two S-boxes in one round. To apply our attack to parallel AES implementations, we focus on a standard structure of unmasked AES shown in Fig. 16. Here we use A to D to denote four 128-bit intermediate states in different stages, and use K to denote the 128-bit key. The index of rounds is denoted by the superscript, and the byte number is denoted by the subscript. For example, K_4^{10} means the 4th byte of the 10th round key.

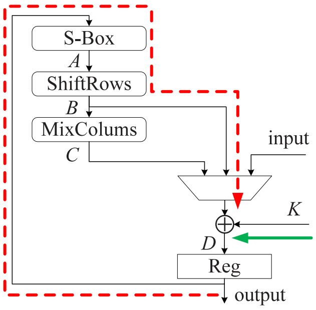

Fig. 16. Standard structure of parallel AES implementation

As in Fig. 16, at the beginning of the 10th round, there are two data paths: the delay of the key is shorter than that of the red dashed path. Consequently, if we shorten the 10th clock cycle to a proper length, we can capture the temporal value $B^9 \oplus K^{10}$ as the faulty ciphertext $\tilde{D}^{10}$, before the intermediate value B^9 is contaminated. Without fault, the output of the circuit is the correct ciphertext $D^{10} = B^{10} \oplus K^{10}$. It is worth noting that both the correct output and faulty output are needed to attack a parallel implementation with TSE attack, which is different from the situation in serial implementation.

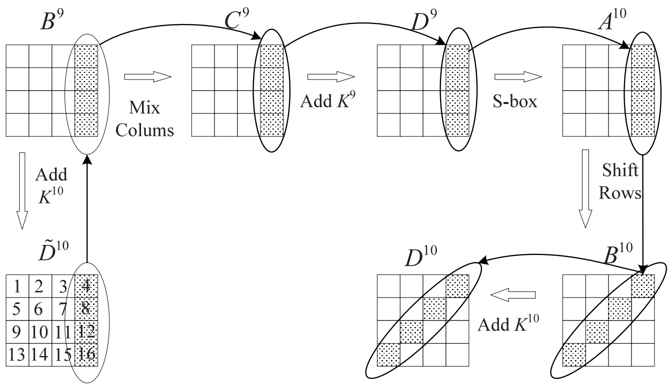

Fig. 17. Key recovery of parallel AES implementation

As shown in Fig. 17, we can deduce equations of K^9 and K^{10} as follows once we have D^{10} and $\tilde{D}^{10}$:

$$\begin{cases} \tilde{D}^{10} \oplus K^{10} = B^9 \\ \text{MixCol}\left(B^9\right) = C^9 \\ C^9 \oplus K^9 = D^9 \\ \text{Sbox}\left(D^9\right) = A^{10} = B^{10} \\ B^{10} \oplus K^{10} = D^{10} \end{cases} \tag{3}$$
$$\Rightarrow \text{Sbox}\left(\text{MixCol}\left(\tilde{D}^{10} \oplus K^{10}\right) \oplus K^9\right) \oplus K^{10} = D^{10} \ .$$

According to AES key schedule, K^9 can be expressed by K^{10}, so K^{10} is the only variable in (3). Solving the equation system in (3) is similar to breaking one round AES by algebraic attack, which can be solved by MiniSAT tool.

However, TSE attack on parallel implementation is feasible only if the round-key is precomputed and stored in registers. Otherwise, the data path of key schedule is comparable to that of S-box.

5.2 Attack Scenario for WDDL-AES

Wave Dynamic Differential Logic (WDDL) is a kind of dual-rail precharge logic. Every signal of WDDL has two complementary wires. Every clock cycle consists of two phases: in precharge phase, both of the wires are precharged to a fixed value, for example, (0, 0); in the evaluation phase, the values of two wires are either (1, 0) or (0, 1). WDDL is believed to be secure against setup violation faults [24]. Because the precharge phase inserts an all-zeros state in every clock, shortening a clock cycle will lead to an all-zeros faulty ciphertext. However, if the delays of different data paths have significant difference, the circuit may not be perfectly secure any more.

Considering a WDDL-AES implementation with the same structure in Fig. 4, we assume that after the rising edge of the clock, the all-zeros state propagates slow enough along the data path of y, so k_2 arrives before the value of y_1 is cleared. As shown in Fig. 18, by shortening the length of the clock, the attacker can store the temporal value and recover the key, which is similar to the analysis in Sect. 3.2.

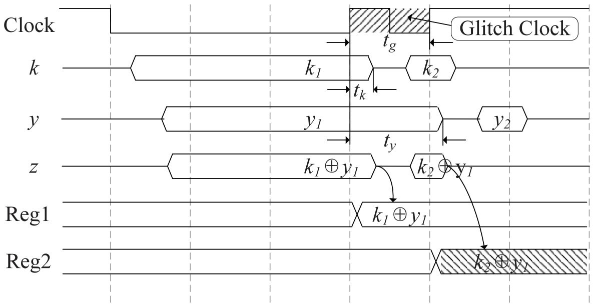

Fig. 18. Sequence diagram of WDDL-AES with clock glitch. The all-zeros state is denoted by Z (high impedance) state.

TSE attack on WDDL-AES is more difficult than non-WDDL AES implementations, because the all-zeros state usually propagates faster than other states [25, 26]. However, it is noteworthy that unbalanced data path remains a potential vulnerability to our attack.

5.3 Glitch Injection

In this section, we discuss about the feasibility of injecting clock glitch externally. The clock glitch required in TSE attack is very short. For example, the width of clock glitch should be no more than 2.8ns for unmasked S-box A. Such a short glitch may be filtered out when injected externally, even though it is reported in many literatures that the glitch width can be smaller than 3ns [27, 28].

A straightforward way to bypass the obstacle is to do a semi-invasive attack: cut the clock line and connect it to a external glitch signal. Another option is to slow down the target circuit, so that TSE attack can be carried out with wider glitch.

Under some conditions, the attacker can increase the delay of the target circuits. For example, by reducing the supply voltage, the propagation delay can be increased [29]. We reduce the supply voltage of FPGA chip from 1.5V to 1.08V and rerun the experiments in Sect. 4.1. As shown in Fig. 19, we cut the power supply of Cyclone IV EP4CE115 and connect it to a DC power supply. The attack results are shown in Fig. 20 and Fig. 21. By reducing the voltage to 1.08V, the feasible glitch frequencies go down to the range from 125MHz to 136MHz, which is about 1/3 of the frequency range with normal voltage.

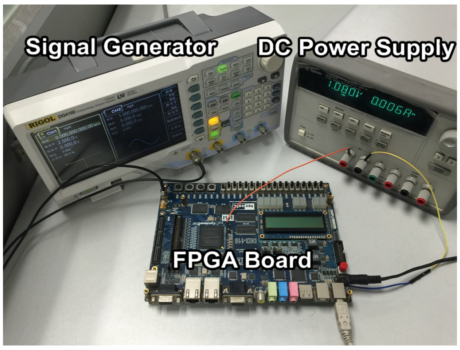

Fig. 19. Glitch injection experiment with reduced supply voltage

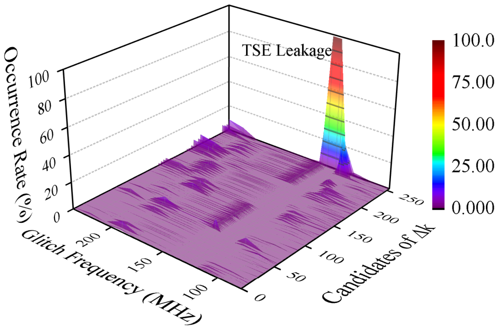

Fig. 20. Results of sweeping glitch frequency for S-box A with reduced supply voltage

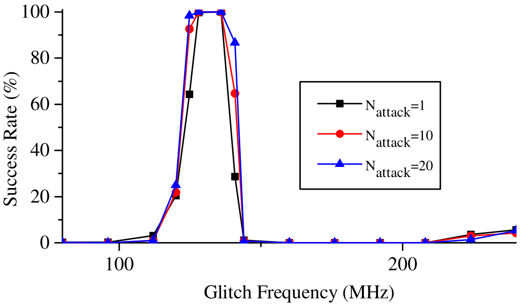

Fig. 21. Success rate *vs.* frequency with different N_{attack} for S-box A with reduced supply voltage

6 Conclusions

In this paper, we propose a new TSE attack based on the transient-steady effect. By injecting glitch in clock signal, the transient-steady value can be captured to recover the key of AES. We conduct experiments on two kinds of unmasked S-boxes and one kind of masked S-box, and all the S-boxes are implemented in serial with an FPGA board. Experimental results show that TSE attack can recover a key byte of the unmasked S-boxes with 1 encryption, and recover a key byte of the masked S-box with less than 20 encryptions. The attack scenarios on parallel AES implementation and WDDL-AES are also discussed.

The foundation of TSE attack is that the path of key is obviously shorter than other data, i.e. the inputs of S-box. Hence, against TSE attack, we recommend the architectures in which the key's path is sufficiently long, for example, the roundkey is generated simultaneously with the encryption. Countermeasures such as inserting dummy operations into the key's path are also feasible options, but the throughput may be impacted.

Acknowledgments. This work is supported by the Foundation of Science and Technology on Information Assurance Laboratory (Nos. KJ-13-101, KJ-14-006), the Major Program Core of Electronic Devices, High-End General Chips, and Basis of Software Products of the Ministry of Industry and Information Technology of China (Nos. 2014ZX01032205, 2014ZX01032401-001), the National Natural Science Foundation of China (No. 61402252), and the National 12th Five-Year Plan Development Foundation for Cryptological Research (No. MMJJ201401009).

References

1. Kocher, P. C.: Timing Attacks on Implementations of Diffie-Hellman, RSA, DSS, and Other Systems. In: Koblitz, N. (ed.) CRYPTO 1996. LNCS, vol. 1109, pp. 104–113. Springer, Heidelberg (1996)
2. Brier, E., Clavier, C., Olivier, F.: Correlation Power Analysis with a Leakage Model. In: Quisquater, J. (ed.) CHES 2004. LNCS, vol. 3156, pp. 16–29. Springer, Heidelberg (2004)
3. Oswald, E., Mangard, S., Herbst, C., Tillich, S.: Practical Second-Order DPA Attacks for Masked Smart Card Implementations of Block Ciphers. In: Pointcheval, D. (ed.) CT-RSA 2006. LNCS, vol. 3860, pp. 192–207. Springer, Heidelberg (2006)
4. Chair, S., Rao, J. R., Rohatgi, P.: Template Attacks. In: Kaliski, B., Koç, C., Paar, C. (eds.) CHES 2002. LNCS, vol. 2523, pp. 13–28. Springer, Heidelberg (2003)
5. Schramm, K., Wollinger, T. J., Paar, C.: A New Class of Collision Attacks and Its Application to DES. In: Johansson, T. (ed.) FSE 2003. LNCS, vol. 2887, pp. 206–222. Springer, Heidelberg (2003)
6. Clavier, C., Feix, B., Gagnerot, G., Roussellet, M., Verneuil, V.: Improved Collision-Correlation Power Analysis on First Order Protected AES. In: Preneel, B., Takagi, T. (eds.) CHES 2011. LNCS, vol. 6917, pp. 49–62. Springer, Heidelberg (2011)

7. Gierlichs, B., Batina, L., Tuyls, P., Preneel, B.: Mutual Information Analysis. In: Oswald, E., Rohatgi, P. (eds.) CHES 2008. LNCS, vol. 5154, pp. 426–442. Springer, Heidelberg (2008)
8. Biham, E., Shamir, A.: Differential Fault Analysis of Secret Key Cryptosystems. In: Kaliski, B. (ed.) CRYPTO 1997. LNCS, vol. 1294, pp. 513–525. Springer, Heidelberg (1997)
9. Li, Y., Sakiyama, K., Gomisawa, S., Fukunaga, T., Takahashi, J., Ohta, K.: Fault Sensitivity Analysis. In: Mangard, S., Standaert, F. (eds.) CHES 2010. LNCS, vol. 6225, pp. 320–334. Heidelberg (2010)
10. Moradi, A., Mischke, O., Paar, C., Li, Y., Ohta, K., Sakiyama, K.: On the Power of Fault Sensitivity Analysis and Collision Side-Channel Attacks in a Combined Setting. In: Preneel, B., Takagi, T. (eds.) CHES 2011. LNCS, vol. 6917, pp. 292–311. Springer, Heidelberg (2011)
11. Li, Y., Ohta, K., Sakiyama, K.: An Extension of Fault Sensitivity Analysis Based on Clockwise Collision. In: Kutyłowski, M., Yung, M. (eds.) Inscrypt 2012. LNCS, vol. 7763, pp. 46–59. Springer, Heidelberg (2013)
12. Wang, A., Chen, M., Wang Z., Wang X.: Fault Rate Analysis: Breaking Masked AES Hardware Implementations Efficiently. In: IEEE Trans. Circuits and Systems-II, vol. 60, no. 8, pp. 517–521. (2013)
13. Mangard, S., Aigner, M., Dominikus, S.: A Highly Regular and Scalable AES Hardware Architecture. In: IEEE Trans. Computer, vol. 52, no. 4, pp. 483–491. IEEE (2003)
14. Feldhofer, M., Wolkerstorfer, J., Rijmen, V.: AES Implementation on a Grain of Sand. In: IEE Proceedings of Information Security, vol. 152, no. 1, pp. 13–20. (2005)
15. Morioka, S., Satoh, A.: An Optimized S-box Circuit Architecture for Low Power AES Design. In: Kaliski, B., Koç, C., Paar, C. (eds.) CHES 2002. LNCS, vol. 2523, pp. 172–186. Springer, Heidelberg (2003)
16. Canright, D.: A Very Compact S-Box for AES In: Rao, J.R., Sunar, B. (eds.) CHES 2005. LNCS, vol. 3659, pp. 441–455. Springer, Heidelberg (2005)
17. Canright, D., Batina, L.: A Very Compact Perfectly Masked S-Box for AES. In: Bellovin, S., Gennaro, R., Keromytis, A., Yung, M. (eds.) ACNS 2008. LNCS, vol. 5037, pp. 446–459. pringer, Heidelberg (2008)
18. Mangard, S., Popp, T., Gammel, B. M.: Side-Channel Leakage of Masked CMOS Gates. In: Menezes, A. (ed.) CT-RSA 2005. LNCS, vol. 3376, pp. 351–365. Springer, Heidelberg (2005)
19. Mangard, S., Pramstaller, N., Oswald, E.: Successfully Attacking Masked AES Hardware Implementations. In: Rao, J.R., Sunar, B. (eds.) CHES 2005. LNCS, vol. 3659, pp. 157–171. Springer, Heidelberg (2005)
20. Suzuki, D., Saeki, M., Ichikawa, T.: Random Switching Logic: A Countermeasure against DPA Based on Transition Probability. Cryptology ePrint Archive, Report 2004/346 (2004), `http://eprint.iacr.org/2004/346`
21. Suzuki, D., Saeki, M., Ichikawa, T.: DPA Leakage Models for CMOS Logic Circuits. In: Rao, J.R., Sunar, B. (eds.) CHES. LNCS, vol. 3659, pp. 366–382. Springer, Heidelberg (2005)
22. Mangard, S., Schramm, K.: Pinpointing the Side-Channel Leakage of Masked AES Hardware Implementations. In: Goubin, L., Matsui, M. (eds.) CHES 2006. LNCS, vol. 4249, pp. 76–90. Springer, Heidelberg (2006)
23. Satoh, A., Morioka, S., Takano, K., Munetoh, S.: A Compact Rijndael Hardware Architecture with S-Box Optimization. In: Boyd, C. (ed.) ASIACRYPT 2001. LNCS, vol. 2248, pp. 239–254. Springer, Heidelberg (2001)

24. Guilley, S., Graba, T., Selmane, N., Bhasin, S., Danger, J.-L.: WDDL is Protected Against Setup Time Violation Attacks. In: FDTC, pp. 73–83. IEEE Computer Society, Los Alamitos (2009)
25. Moradi, A., Immler, V.: Early Propagation and Imbalanced Routing, How to Diminish in FPGAs. In: Batina, L., Robshaw, M. (eds.) CHES 2014. LNCS, vol. 8731, pp. 598–615. Springer, Heidelberg (2014)
26. Tiri, K., Akmal, M., Verbauwhede, I.: A Dynamic and Differential CMOS Logic with Signal Independent Power Consumption to Withstand Differential Power Analysis on Smart Cards. In: ESSCIRC 2002, pp. 403–406. (2002)
27. Takahashi, J., Fukunaga, T., Gomisawa, S., Li, Y., Sakiyama, K., Ohta, K.: Fault Injection and Key Retrieval Experiments on an Evaluation Board. In: Joye, M., Tunstall, M. (eds) Fault Analysis in Cryptography, pp. 313–331. Springer, Heidelberg (2012)
28. Agoyan, M., Dutertre, J. M., Naccache, D., Robisson, B. Tria, A.: When Clocks Fail: On Critical Paths and Clock Faults. In: Gollmann, D., Lanet, J. L., Iguchi-Cartigny, J. (eds.) Smart Card Research and Advanced Application, LNCS, vol. 6035, pp. 182–193. Springer, Heidelberg (2010)
29. Guilley, S., Danger, J. L.: Global Faults on Cryptographic Circuits. In: Joye, M., Tunstall, M. (eds) Fault Analysis in Cryptography, pp. 295–311. Springer, Heidelberg (2012)