Snowball: Another View on Side-Channel Key Recovery Tools*

Jiangshan Long[1], Changhai Ou[1], Zhu Wang[2], Shihui Zheng[3], Fei Yan[1], Fan Zhang[4], and Siew-Kei Lam[5]

[1] Wuhan University, Wuhan, China
longjiangshan@whu.edu.cn, ouchanghai@whu.edu.cn, yanfei@whu.edu.cn
[2] Chinese Academy of Sciences, Beijing, China
wangzhu@iie.ac.cn
[3] Beijing University of Posts and Telecommunications, Beijing, China
shihuizh@bupt.edu.cn
[4] Zhejiang University, Beijing, China
fanzhang@zju.edu.cn
[5] Nanyang Technological University, Singapore
assklam@ntu.edu.sg

Abstract

The performance of Side-Channel Attacks (SCAs) decays rapidly when considering more sub-keys, making the full-key recovery a very challenging problem. Limited to independent collision information utilization, collision attacks establish the relationship among sub-keys but do not significantly slow down this trend. To solve it, we first exploit the samples from the previously attacked S-boxes to assist attacks on the targeted S-box under an assumption that similar leakage occurs in program loop or code reuse scenarios. The later considered S-boxes are easier to be recovered since more samples participate in this assist attack, which results in the "snowball" effect. We name this scheme as Snowball, which significantly slows down the attenuation rate of attack performance. We further introduce confusion coefficient into the collision attack to construct collision confusion coefficient, and deduce its relationship with correlation coefficient. Based on this relationship, we give two optimizations on our Snowball exploiting the "values" information and "rankings" information of collision correlation coefficients named Least Deviation from Pearson correlation coefficient (PLD) and Least Deviation from confusion coefficient (CLD). Experiments show that the above optimizations significantly improve the performance of our Snowball.

Contents

*Corresponding author: Changhai Ou.

1 Introduction

Secret information will unintentionally leak through side-channels such as execution time [13], power consumption [16,26], electromagnetic radiation [8] and cache patterns [17–19] when cryptographic algorithms are executed on devices. By collecting these side-channel informations and performing SCAs, many cryptographic systems in real world have been successfully conquered, which makes SCAs attract wide attentions. Power side-channel attacks are the most popular one of them in these years, and they can be launched in two models: divide-and-conquer (e.g., Correlation Power Analysis (CPA) [4] and Template Attack (TA) [6]) and analytical (e.g., collision attacks [20]). For the former, they divide the full key into small blocks (e.g., sub-keys in AES-128) and conquer them one by one, then exploit key enumeration tools [11,15,22] to enumerate the full-key candidates from the most possible one to the least possible one. However, they are still limited by the computing power of the attacker, and can only be exploited in the scenarios that cryptographic implementations are "practically insecure" (for which the leakage allows for key enumeration).

In this paper, we aim at side-channel full-key recovery and we are interested in the analytical collision attacks [14] considering all sub-keys simultaneously by solving a system of equations, since they exploit more leaky information and are more efficient than divide-and-conquer attacks. However, they are also more complex and the probability of successful attack (i.e., Success Rate [25]) will be significantly pulled down when attacking more sub-keys simultaneously. Related works will be introduced in the next subsection before introducing our contributions on solving the above problems.

1.1 Related Works

Benefiting from the repeated operations (e.g., program loop and code reuse) in the cryptographic implementations, the hardware generates very similar side-channel leakages. Unlike the divide-and-conquer attacks exploiting direct leakages, side-channel collision attacks exploit these collision leakages and achieve a higher performance. This advantage was firstly taken in [24] on DES and then taken in [23] on AES to identify the same inputs of S-boxes in the

same encryption or decryption by comparing the similarity of leakage samples. Therefore, side-channel collision attacks circumvent the modeling issue, i.e., they are non-profiled attacks. To exploit more leakage information, Bogdanov et al. extended this to different encryptions or decryptions in [1, 2].

Although taking advantage of non-profiled attacks and circumventing the modeling issue, the above mentioned side-channel collision attacks cannot be exploited in the protected implementations like masking, since the intermediate values are masked and their correlation with the leakages is hidden in this case. However, for some flawed masking implementation like the flawed first-order Rotated S-boxes Masking (RSM) [21] scheme implemented by the DPA *contest* $v4.1$[1], first-order leakage still exists. Moradi et al. extracted the leakage samples for each possible plaintext byte value, then averaged them to estimate their mean moments to de-noise in [20]. They then performed Correlation-Enhanced Collision Attack (abbreviated as CECA here) using Pearson correlation coefficient on the XORed value of each two sub-keys, and successfully detected the leakage. Although the original target of CECA is to detect flawed masking implementations, its performance is still limited by its inefficient use of leakage information and could be further improved.

Bruneau et al. combined the flavours of stochastic and collision attacks and provided the stochastic collision attack in [5], which exploited scalar product scores to measure the similarity of leakages and extended this to multi-collision cases. Cezary et al. proposed a strategy named optimal collision attack based on the maximum likelihood principle [10], and combined an additional key searching algorithm. Wiemers et al. also exploited a key searching algorithm after the classic CECA to enhance the full-key recovery in [27]. They extracted a part of the best collision candidates and discarded the remaining most combinations unsatisfying the given collision conditions, thus making the key recovery easier. However, if we only consider the performance of key recovery from distinguishers (i.e., without consideration on the combined key searching algorithms), the above two collision attacks achieve a close performance but lower than stochastic collision attack.

It's obviously that the above works have made great efforts and gained remarkable progress on side-channel collision attacks. However, their performance may still be insufficient when facing with very huge candidate space in full-key recovery. More efficient collision-based full-key recovery schemes are worth studying.

1.2 Our Contributions

Most of the existing side-channel full-key recovery schemes ignore the information reusability of samples corresponding to different S-boxes due to the repeated operations (e.g., program loop and code reuse) in the cryptographic implementations, which causes the attack performance to decay rapidly when considering more sub-keys and attracts wide attentions. We focus on CECA in this paper and our main contributions on this issue are as follows:

(i) Unlike exploiting leakage samples only once in the previous attacks, we exploit the samples from the previously attacked S-boxes to assist attacks on the targeted S-box under an assumption that similar leakage occurs in the program loop or code reuse scenarios. The later an sub-key is considered, the more gentle the attenuation trend of attack performance on it, and the "snowball" effect happens in this case. We name this full-key recovery scheme as Snowball.

[1]DPA Contest. http://www.dpacontest.org/home/

(ii) We introduce the confusion coefficient into side-channel collision attacks to analyse their mathematical properties, and build the collision confusion coefficient suitable for collision attacks.

(iii) We deduce the relationship between collision confusion coefficient and collision correlation coefficient, and propose two optimizations to improve our Snowball. Our first optimization named PLD exploits all collision correlation coefficients to identify their "values" deviation caused by noise rather than only one or several best candidates of them in the previous attacks, and restores their "false" descending order in the noise scenarios to the one without noise. Our second optimization named CLD exploits the "rankings" deviation of collision correlation coefficients rather than "values" deviation to analyze the similarity of the correlation coefficient with or without noise.

Our Snowball is very simple, and it brings us a new road for efficient full-key recovery. Our experimental results fully illustrate its superiority.

1.3 Organization

The rest of this paper is organized as follows: side-channel leakage, collision attack, CECA and optimal collision attack are introduced in Section 2. Our Snowball including its collision model, principle and algorithm description are detailed in Section 3. Collision confusion coefficient and its relationship with collision (Pearson) correlation coefficient are given in Section 4. Based on this, our two optimizations PLD and CLD exploiting the "values" information and "rankings" information of collision correlation coefficients are further detailed in Section 5. Experiments on simulated samples and an ATmega328p micro-controller are presented in Section 6 to illustrate the superiority of our Snowball and its optimizations. Finally, we conclude this paper in Section 7.

2 Preliminaries

2.1 Side-Channel Leakage

Let n denote the input size of the S-box (e.g. $n = 8$ for AES-128), L denote the number of S-boxes in each round (e.g. $L = 16$ for AES-128), $k^{*(l)}$ denote the l-th sub-key, $k^{(l)}$ denote the corresponding guessing value ($l = 1, 2, \ldots, L$), Q denote the number of plaintexts totally encrypted, $t_q^{(l)}$ denote the l-th block of the q-th encrypted plaintext and $x_q^{(l)}$ denote the corresponding leakage ($q = 1, 2, \ldots, Q$). Here an identical leakage model can be expressed as:

$$x_q^{(l)} = \psi\left(t_q^{(l)} \oplus k^{*(l)}\right) + \mathbb{N}_q^{(l)}, \tag{1}$$

and abbreviated to:

$$x_q^{(l)} = \psi_{t_q^{(l)}, k^{*(l)}} + \mathbb{N}_q^{(l)}. \tag{2}$$

Here $\mathbb{N}_q^{(l)}$ is the additive and independent noise component on the q-th trace with zero mean $\left(\mathbb{E}\left\{N_q^{(l)}\right\} = 0\right)$ but not necessarily follows Gaussian distribution. ψ is a deterministic unknown leakage function corresponding to the look-up table operation of S-boxes. $\boldsymbol{x}^{(\cdot)}$ is the matrix with the q-th row corresponding to the L-variate leakage $x_q^{(1)}, x_q^{(2)}, \ldots, x_q^{(L)}$. Both $\mathbb{N}_q^{(l)}$ and ψ are specific to physical characteristics of the underlying implementation of S-boxes. Therefore, we do not make any particular assumption here, thus making our attack scenario general.

2.2 Collision Attack

S-boxes are designed to be bijective and identical in the most cases (e.g. AES-128), and the same input always generates the corresponding same output in this case. As a result, a collision happens if two identical S-boxes encounter the same input:

$$t_{q_1}^{(l_1)} \oplus k^{*(l_1)} = t_{q_2}^{(l_2)} \oplus k^{*(l_2)}, \tag{3}$$

and we obtain the collision value:

$$\begin{aligned} \delta^{*(l_1,l_2)} &= k^{*(l_1)} \oplus k^{*(l_2)} \\ &= t_{q_1}^{(l_1)} \oplus t_{q_2}^{(l_2)} \end{aligned} \tag{4}$$

in this case (as shown in Figure 1). This collision causes very similar effect (e.g. power consumption, electromagnetic radiation).

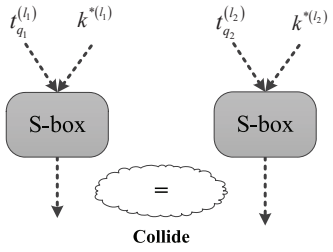

Figure 1: A collision happens if two S-boxes accept the same input and generate the same output.

Collision attack judges collisions by comparing the similarity of leakages [3]. The classic side-channel collision attack only exploits the collisions happening in the same encryption as described in [5] (see Equation (11) given in [5]), which can be expressed as:

$$\mathcal{D}_{coll} = \underset{\delta^{(l_1,l_2)} \in \mathbb{F}_2^8}{\arg\min} \frac{\sum_{q/t_q^{(l_1)} \oplus t_q^{(l_2)} = \delta^{(l_1,l_2)}} \left(x_q^{(l_1)} - x_q^{(l_2)} \right)^2}{\sum_{q/t_q^{(l_1)} \oplus t_q^{(l_2)} = \delta^{(l_1,l_2)}} 1} \tag{5}$$

under a hypothesis $\delta^{(l_1,l_2)} = t_q^{(l_1)} \oplus t_q^{(l_2)} = k^{(l_1)} \oplus k^{(l_2)}$ $(q = 1, 2, \ldots, Q)$.

2.3 Correlation-Enhanced Collision Attack

Correlation-Enhanced Collision Attack (CECA) divides the leakage $x_q^{(l)}$ $(q = 1 \ldots Q, l = 1 \ldots L)$ of each S-box according to plaintext byte values $t_q^{(l)}$, then averages them as:

$$\tau_u^{(l)} = \frac{\sum_{q/t_q^{(l)}=u} x_q^{(l)}}{\sum_{q/t_q^{(l)}=u} 1} \tag{6}$$

according to their input values u-s. $\boldsymbol{x}^{(\cdot)}$ becomes a matrix of real numbers of dimension $2^n \times L$ after performing average, where the q-th row corresponds to the leakage of the plaintext byte

value $q-1$. CECA then correlates the l_1-th S-box with the l_2-th S-box under a guessing collision value $\delta^{(l_1,l_2)}$ and obtains:

$$\mathcal{D}_{corr.coll} = \underset{\delta^{(l_1,l_2)} \in \mathbb{F}_2^8}{\arg\max}\, \rho\left\{\left(\tau^{(l_1)}_{u\in\mathbb{F}_2^8}, \tau^{(l_2)}_{u\oplus\delta^{(l_1,l_2)}}\right)\right\}. \tag{7}$$

Here $\rho\{\cdot\}$ denotes the correlation coefficient computation. Wiemers et al. gave an ad-hoc evaluation function on CECA as:

$$\mathcal{D}_{corr.coll} = \underset{\delta \in (\mathbb{F}_2^8)^{L-1}}{\arg\max} \sum_{l_1<l_2} \rho\left\{\left(\tau^{(l_1)}_{u\in\mathbb{F}_2^8}, \tau^{(l_2)}_{u\oplus\delta^{(l_1,l_2)}}\right)\right\} \tag{8}$$

in [27] for the full-key recovery.

2.4 Optimal Collision Attack

Cezary et al. gave a new attack named optimal collision attack as:

$$\mathcal{D}_{opt.coll} = \underset{\delta\in\mathbb{F}_2^8}{\arg\max} \sum_{u\in\mathbb{F}_2^8} \tau^{(l_1)}_u \times \tau^{(l_2)}_{u\oplus\delta} \tag{9}$$

with strict theory proof in [10], which follows the maximum likelihood principle. Actually, $\sum_{u\in\mathbb{F}_2^8} \tau^{(l_1)}_u \times \tau^{(l_2)}_{u\oplus\delta}$ is the cross multiplication term of two columns of the averaged power consumptions according to plaintext byte values in CECA, which fully illustrates the relationship between the optimal collision attack and CECA.

3 Our Snowball

3.1 Targeted Collision Sequence

There are $\frac{16*15}{2} = 120$ different $\delta^{*(l_1,l_2)}$ $(1 \le l_1, l_2 \le 16)$ for AES-128, but 15 collision values such as $\delta^{*(1,2)}$, $\delta^{*(2,3)}$, ..., $\delta^{*(15,16)}$ can provide us with all remaining δ^*-s. Since any $\delta^{*(l_1,l_3)}$ can be deduced from $\delta^{*(l_1,l_2)} \oplus \delta^{*(l_2,l_3)}$. We only need to enumerate a sub-key after determining such a collision sequence, and the key entropy eventually reduces from 128 bits to 8 bits, which can easily be enumerated and verified. Here the targeted collision sequence in our Snowball can be $\delta^{*(1,2)}$, $\delta^{*(1,3)}$, ..., $\delta^{*(1,16)}$, and the same analysis can be directly applied to any other possible targeted sequences (e.g., $\delta^{*(1,2)}$, $\delta^{*(2,3)}$, ..., $\delta^{*(15,16)}$) in a straightforward way.

3.2 Core Idea

In previous works, collision attacks targeting at different $\delta^{*(l_1,l_2)}$ are conducted independently. In other words, the targeted collision sequence can be determined in an arbitrary order. For simplicity, we suggest that collision attacks should be conducted on the targeted collision sequence sequentially. The candidates of collision values $\delta^{*(1,2)}$, $\delta^{*(1,3)}$, ..., $\delta^{*(1,l-1)}$ may be known in attack since the previous attacks have already finished. Unfortunately, the samples corresponding to these candidates of $\delta^{*(1,2)}$, $\delta^{*(1,3)}$, ..., $\delta^{*(1,l-1)}$ are just left aside, and they make no contribution to the current attack on the l-th S-box. On the other hand, if more than one candidate are considered after each collision attack, how to combine them to recover the full-key becomes another difficulty.

S-boxes in AES-128 are designed and implemented to be identical and even share the same implementation (e.g., code reuse) in some cases. Samples derived from different S-boxes follow the same (or very similar) distribution in this case. Base on this assumption, we exploit the samples from the previously attacked S-boxes to assist attacks on the targeted S-box, which virtually increases the samples of the targeted S-box currently under attack, thus significantly improving the attack performance. Moreover, the later an S-box is considered, the more obvious this advantage is, since the errors happening on a very small part of the previous collision values will not have a significant impact on its attack performance. Therefore, the performance of the full-key recovery will not decrease significantly with more sub-keys being taken into consideration, but maintain a high probability of successful key recovery and become more and more gentle. Obviously, the "snowball" effect occurs in this case, and we name this full-key recovery strategy as "Snowball".

3.3 Algorithm Description

Let num denote the number of combinations we maintain in the window after attacks performed on each S-box, $\Gamma_{l_1}^{l_j}$ denote the corresponding combinations containing collision candidates from the l_1-th to the l_j-th sub-keys, Thr denote the number of candidates of $\delta^{*(1,l)}$ under consideration and $\delta^{*(1;2\cdots l-1)}$ denote a combination of $(\delta^{*(1,2)}, \ldots, \delta^{*(1,l-1)})$ including $l-2$ δ-s. To attack the $\delta^{*(1,l)}$ from the extracted num optimal candidates $\Gamma_1^{l-1} = (\ \delta_1^{(1;2\cdots l-1)}, \delta_2^{(1;2\cdots l-1)}, \ldots, \delta_{num}^{(1;2\cdots l-1)})$ within window. Our Snowball is given in Algorithm 1.

Algorithm 1: Snowball algorithm

Input: Averaged samples $\boldsymbol{\tau} = \left(\tau^1, \ldots, \tau^{16}\right)$, the thresholds num and Thr, and $\Gamma_1^2 = \Gamma_1^3 = \ldots = \Gamma_1^{16} = \phi$.
Output: The full-key candidates Γ_1^{16}.

```
for δ^(1,2) in Corr(τ^(1), τ^(2), Thr) do
    Add δ^(1,2) to Γ_1^2;
end
Γ_1^2=Top(Γ_1^2, num);
for l from 3 to 16 do
    for δ^(1;2⋯l−1) in Γ_1^{l−1} do
        for u from 0 to 255 do
            Compute τ_u^{δ^(1;2⋯l−1)};
        end
        for δ^(1,l) in Corr(τ^{δ^(1;2⋯l−1)}, τ^(l), Thr) do
            Add (δ^(1;2⋯l−1), δ^(1,l)) to Γ_1^l;
        end
    end
    Γ_1^l=Top(Γ_1^l, num);
end
```

To optimize the brute force, we can exploit CECA to obtain the rank of collision candidates and the remaining Snowball only considers the best Thr collision candidates for each $\delta^{*(1,l)}$. We first perform this to extract a total of Thr best candidates from $\delta^{*(1,2)}$ (Steps $1 \sim 4$), then extract the best combinations in the window with a size of num and save them to Γ_1^2 (Step 4).

The thresholds Thr and num are very different in this case.

We further exploit the samples from the previously attacked S-boxes to assist attacks on the targeted S-box, and virtually increase the samples in attacks. In other words, we reuse samples of the prior S-boxes based on combinations in Γ_1^{l-1} to deduce candidates of the targeted $\delta^{*(1,l)}$. Specifically, we stack the samples of the first $2 \cdots l-1$ S-boxes into the first S-box according to the $\delta^{(1;2\cdots l-1)}$ in Γ_1^{l-1} and calculate the mean according to their plaintext byte values as follows:

$$\tau_u^{\delta^{(1;2\cdots l-1)}} = \frac{\sum_{q/t_q^{(1)}=u} x_q^{(1)} + \sum_{i=2}^{l-1} \sum_{q/t_q^{(i)}=u\oplus\delta^{(1,i)}} x_q^{(i)}}{\sum_{q/t_q^{(1)}=u} 1 + \sum_{i=2}^{l-1} \sum_{q/t_q^{(i)}=u\oplus\delta^{(1,i)}} 1} \tag{10}$$

(Steps $7 \sim 9$). These means can more accurately reflect the power consumptions of the same inputs of S-box and become more referential if the guessing $\delta^{(1;2\cdots l-1)}$ is correct.

We then use $\tau_u^{\delta^{(1;2\cdots l-1)}}$ to replace $\tau_u^{(1)}$, perform the CECA on the $\delta^{*(1,l)}$ and save the new combinations $(\delta^{(1;2\cdots l-1)}, \delta^{(1,l)})$ to Γ_1^l (Steps $10 \sim 12$). This assisted attack makes the new $\tau_u^{(1)}$ better resistance to noise and the correlation coefficient corresponding to $\delta^{*(1,l)}$ more obvious compared to the others. It is noteworthy that this reference will be no significant change if errors happen on a very small part of δ-s, thus our Snowball is robust in this case.

Finally, we extract the optimal num combined candidates $\delta^{(1;2\cdots l)}$ from Γ_1^l, update the window and begin the next iteration. Obviously, more and more samples will participate in the following assistance attacks, thus significantly facilitating the effectiveness. Benefiting from this "snowball" effect, the later an S-box is considered, the higher attack performance it achieves, and the more gentle the global performance is in this case. Therefore, our Snowball usually maintains a high attack performance. Experiment results in Section 6 will show these advantages in more details.

4 Collision Confusion Coefficient in Collision Attacks

4.1 Confusion Coefficient

Mathematical properties and physical implementation of an S-box determine its SCA-related properties. Confusion coefficient given by Fei et al. in [9] quantifies correlation between different sub-key values statistically and characterizes SCA-related properties of an S-box in theory. Let k_1 and k_2 denote two candidates of an sub-key, the confusion coefficient between them can be expressed as:

$$\kappa(k_1, k_2) = E\left\{(\psi_{k_1,t} - \psi_{k_2,t})^2\right\}. \tag{11}$$

Confusion coefficients are originally used in divide-and-conquer attacks (e.g. CPA) where each S-box is attacked independently. A small confusion coefficient illustrates that two candidates of an sub-key are 'close' to each other, and they would perform similarly in attacks. In other words, this makes it difficult to distinguish them in side-channel attacks.

4.2 Collision Confusion Coefficient

The mathematical properties of side-channel collision attacks were seldom discussed in the previous works, here we extend the confusion coefficient to collision attacks, propose collision confusion coefficient and try to fill this vacancy. Since $t^{(l_1)} = t^{(l_2)} \oplus \delta$ will collide if $\delta = \delta^*$, the

input of another S-box is extended to:

$$\begin{aligned} t^{(l_2)} \oplus k^{*(l_2)} &= \left(k^{*(l_1)} \oplus \delta^*\right) \oplus \left(t^{(l_1)} \oplus \delta\right) \\ &= \left(k^{*(l_1)} \oplus t^{(l_1)}\right) \oplus \left(\delta^* \oplus \delta\right). \end{aligned} \tag{12}$$

Let $\Delta \in \mathbb{F}_2^8$ denote $\delta \oplus \delta^*$. This deviation Δ determines the effectiveness of the collision attacks, thus we treat Δ as the only parameter and extend the confusion coefficient to collision confusion coefficient as:

$$\begin{aligned} \kappa\left(\Delta\right) &= \frac{1}{256} \sum_{t^{(l_1)}, t^{(l_2)} \in \mathbb{F}_2^8} \left(\psi_{k^{*(l_1)}, t^{(l_1)}} - \psi_{k^{*(l_2)}, t^{(l_2)}}\right)^2 \\ &= \frac{1}{256} \sum_{t \in \mathbb{F}_2^8} \left(\psi_{k^{*(l_1)}, t} - \psi_{k^{*(l_1)}, t \oplus \Delta}\right)^2. \end{aligned} \tag{13}$$

Here we further use $\theta = k^{*(l_1)} \oplus t$ to simplify the Equation (13) as:

$$\kappa\left(\Delta\right) = \frac{1}{256} \sum_{\theta \in \mathbb{F}_2^8} \left(\psi\left(\theta\right) - \psi\left(\theta \oplus \Delta\right)\right)^2. \tag{14}$$

The above Equation (14) indicates that we can only consider Δ instead of two sub-keys $k^{*(l_1)}$ and $k^{*(l_2)}$. Here Δ totally depends on the targeted δ^* and guessing δ. From this perspective, we believe our extension is much more suitable for side-channel collision attacks. It inspires analyzing the mathematical properties of S-box from another aspect, which we will introduce in detail in the next sub-section.

4.3 Relationship between Collision Correlation Coefficient and Collision Confusion Coefficient

In this section, we analyse the relationship between collision correlation coefficient and collision confusion coefficient in theory, thus laying the theoretical foundation for our optimizations in Section 5. For AES-128, for simplicity and without loss of generality, we regard ψ as the widely used Hamming weights of the outputs of an S-box. In this case, ψ follows Bernoulli distribution: $\psi\left(\theta\right) \sim \mathcal{B}\left(8, \frac{1}{2}\right)$. We extend Equation (14) as:

$$\begin{aligned} \kappa\left(\Delta\right) =& E\left[\psi^2\left(\theta\right)\right] + E\left[\psi^2\left(\theta \oplus \Delta\right)\right] \\ & - 2 * E\left[\psi\left(\theta\right)\psi\left(\theta \oplus \Delta\right)\right]. \end{aligned} \tag{15}$$

Here "$E(\cdot)$" denotes the expectation operator. We obtain:

$$E\left[\psi^2\left(\theta\right)\right] = E\left[\psi^2\left(\theta \oplus \Delta\right)\right] = 18, \tag{16}$$

and get:

$$E\left[\psi\left(\theta\right)\psi\left(\theta \oplus \Delta\right)\right] = 18 - \frac{\kappa\left(\Delta\right)}{2}. \tag{17}$$

Obviously, the relationship between correlation coefficient ρ and the deviation Δ can be given as:

$$\rho = 1 - \frac{\kappa(\Delta)}{4}. \tag{18}$$

We can generate a bijective Δ-to-$\kappa(\Delta)$ mapping table from Equation (18). This mapping is determined by the mathematical properties of the S-box and is completely independent of the specifical implementation. The larger collision correlation coefficients imply their corresponding candidates being the δ^*-s with a larger probability. It's noteworthy that ascending $\kappa(\Delta)$ is equivalent to descending correlation coefficients according to Equation (18). Therefore, we can sort the Δ-to-$\kappa(\Delta)$ mapping table by ascending $\kappa(\Delta)$, and finally obtain a constant sequence Δ-s denoted as $\boldsymbol{\Delta} = (\Delta^1, \Delta^2, \ldots, \Delta^{256})$ (as shown in Table 1). For better display, this sorted Δ-to-$\kappa(\Delta)$ mapping table is also given in Figure 2.

$\boldsymbol{\Delta}$	Δ^1	Δ^2	Δ^3	Δ^4	Δ^5	$\cdots$	Δ^{255}	Δ^{256}
Value	0	62	202	22	184	$\cdots$	63	245

Table 1: Constant sequence $\boldsymbol{\Delta}$ for AES-128.

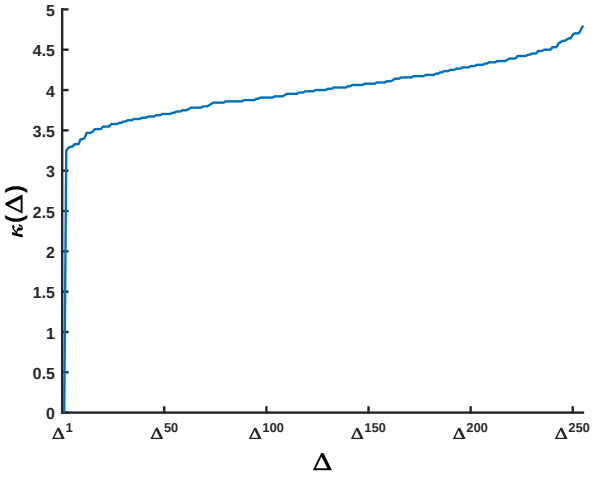

Figure 2: Δ-to-$\kappa(\Delta)$ mapping table.

We can derive a sequence of candidates corresponding to the descending sequence of correlation coefficients from $\boldsymbol{\Delta}$ in theory, and denote it as:

$$\boldsymbol{\delta^*} = \delta^* \oplus \left(\Delta^1, \Delta^2, \ldots, \Delta^{256}\right).$$

The descending sequence of correlation coefficients ρ-s in theory (without noise) can be expressed as:

$$\begin{aligned}\boldsymbol{\rho_{\delta^*}} &= \left(\rho_{\delta^* \oplus \Delta^1}, \rho_{\delta^* \oplus \Delta^2}, \ldots, \rho_{\delta^* \oplus \Delta^{256}}\right) \\ &= \left(1 - \frac{\kappa\left(\Delta^1\right)}{4}, 1 - \frac{\kappa\left(\Delta^2\right)}{4}, \ldots, 1 - \frac{\kappa\left(\Delta^{256}\right)}{4}\right),\end{aligned}$$

which is also a constant sequence as shown in Figure 3. Here we draw a conclusion on AES-128 that collision confusion coefficients $\kappa(\Delta)$ all fall between 3.25 and 4.92 with mean 4 and variance 0.174 except $\kappa\left(\Delta^1\right) = 0$ (i.e., $\delta = \delta^*$). Small variance implies relatively high security, since it brings almost the same difficulty when attempting to distinguish the correct δ^* from any other candidates.

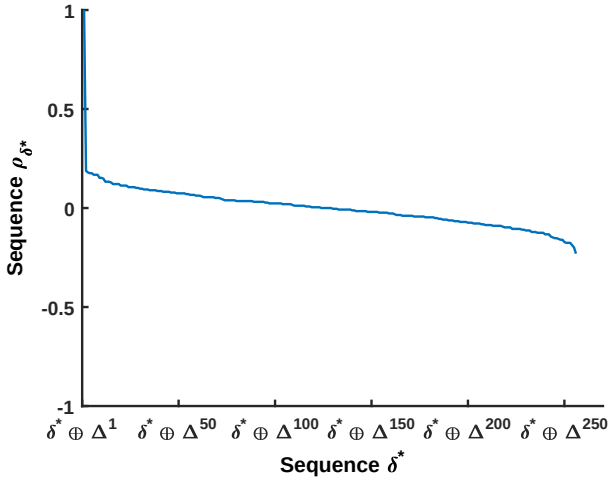

Figure 3: Relationship between sequences $\boldsymbol{\rho_{\delta^*}}$ and $\boldsymbol{\delta^*}$.

5 Optimizations on Snowball

Noise will disturb the sequences $\boldsymbol{\rho_{\delta^*}}$ and $\boldsymbol{\delta^*}$ we introduced in Section 4. Let $\boldsymbol{\delta^{\#}} = \left(\delta^1, \delta^2, \ldots, \delta^{256}\right)$ and their corresponding Pearson correlation coefficients $\boldsymbol{\rho_{\delta^{\#}}} = (\rho_{\delta^1}, \rho_{\delta^2}, \ldots, \rho_{\delta^{256}})$ denote the corresponding disordered sequences observed in noise scenarios. It becomes a challenging task to recover the unknown sequence $\boldsymbol{\delta^*}$ from its disordered version (i.e., sequence $\boldsymbol{\delta^{\#}}$). However, δ^* is determined once we successfully recover it, since sequence $\boldsymbol{\Delta}$ is constant and known. To solve this, the previous works exploited a very straightforward strategy by directly regarding the first element δ^1 as δ^*. This wastes the information contained in other elements of $\boldsymbol{\delta^{\#}}$ and results in a relatively low performance. To compensate this, we give two novel optimizations named Least Deviation from Pearson correlation coefficient (PLD) and Least Deviation from confusion coefficient (CLD) to overcome noise and make the recovery of the sequence $\boldsymbol{\delta^*}$ in our Snowball more efficiently.

5.1 Deviation from Sequence ρ_{δ^*}-s in PLD

The location of δ^* (i.e., $\delta^* \oplus \Delta^1$) in $\boldsymbol{\delta^{\#}}$ in a collision attack is somewhat blurry in noise scenarios. To address this, we search the sequence $\boldsymbol{\delta^{\#}}$ from the beginning to end. It's noteworthy that one can balance the depth of search and time consumption according to his computing power. We construct a new sequence $\boldsymbol{\delta^n}$ with δ^n as its first element, and compare it with the $\boldsymbol{\delta^*}$ in noiseless scenario in Table 2. Obviously, $\boldsymbol{\delta^n}$ is an re-ordered sequence of $\boldsymbol{\delta^{\#}}$. We can turn it to collision correlation coefficients ρ-s, and re-sort the descending sequence $\boldsymbol{\rho_{\delta^{\#}}}$ to

$\boldsymbol{\rho_{\delta^n}} = (\rho_{\delta^n \oplus \Delta^1}, \rho_{\delta^n \oplus \Delta^2}, \ldots, \rho_{\delta^n \oplus \Delta^{256}})$ according to sequence $\boldsymbol{\delta^n}$. Our first optimization named Least Deviation from Pearson correlation coefficient (PLD) aims to exploit the least deviation between $\boldsymbol{\rho_{\delta^*}}$ and $\boldsymbol{\rho_{\delta^n}}$, which indicates the correct collision value.

$\boldsymbol{\delta^*}$	$\delta^* \oplus \Delta^1$	$\delta^* \oplus \Delta^2$	$\delta^* \oplus \Delta^3$	$\cdots$	$\delta^* \oplus \Delta^{256}$
$\boldsymbol{\delta^n}$	$\delta^n \oplus \Delta^1$	$\delta^n \oplus \Delta^2$	$\delta^n \oplus \Delta^3$	$\cdots$	$\delta^n \oplus \Delta^{256}$

Table 2: Sequences $\boldsymbol{\delta^*}$ and $\boldsymbol{\delta^n}$.

In principle, deviations in our PLD between sequence $\boldsymbol{\delta^n}$ and sequence $\boldsymbol{\delta^*}$ consist of two parts:

(i) Deviation from noise. This is reflected in correlation coefficients thrashing around $1 - \frac{\kappa(\Delta)}{4}$ randomly, and is inevitable.

(ii) Deviation from assumption error. This additional part comes from the truth that $\delta^n \neq \delta^*$, and $\boldsymbol{\rho_{\delta^n}}$ we constructed is not the descending sequence $\boldsymbol{\rho_{\delta^*}}$ in noiseless scenario. This second part of deviations provides valuable information for judgment.

In summary, if $\delta^n = \delta^*$ in our PLD, there only exists noise-induced deviation and we can infer that the overall deviation is likely to be relatively small. Thus, we can distinguish δ^* from other candidates. Here we quantify the deviations as:

$$\begin{aligned} \Phi_{\boldsymbol{\delta^n}} &= \sum_{m=1}^{256} \left(\rho_{\delta^n \oplus \Delta^m} - \rho_{\delta^* \oplus \Delta^m}\right)^2 \\ &= \sum_{m=1}^{256} \left(\rho_{\delta^n \oplus \Delta^m} - \left(1 - \frac{\kappa\left(\Delta^m\right)}{4}\right)\right)^2, \end{aligned} \tag{19}$$

and regard the δ^n corresponding to the minimum Φ_{δ^n} as the collision value δ^*.

For intuitive feeling of the noise-induced deviations in our PLD, we compare the sequence $\boldsymbol{\rho_{\delta^*}}$ under different noise levels with the constant sequence $\boldsymbol{\rho_{\delta^*}}$ in theory (i.e., noiseless scenarios) in Figure 4. Here $Q = 2000$ samples are randomly generated from Equation (1), the orange line represents sequence $\boldsymbol{\rho_{\delta^*}}$ in noisy scenarios and the blue line represents $\boldsymbol{\rho_{\delta^*}}$ in theory. Obviously, the descending tendency of sequence $\boldsymbol{\rho_{\delta^*}}$ gradually disappears when the standard deviation σ of noise is from 2 to 10, and ρ_{δ^*} ($\rho_{\delta^* \oplus \Delta^1}$) under $\sigma = 8$ and $\sigma = 10$ is even not the highest. In spite of this, $\Phi_{\boldsymbol{\delta^*}}$ is still almost smaller than others in these cases.

For the second part of deviations in our PLD from assumption error, we can accurately quantify it. Specifically, let λ denote the deviation $\delta^* \oplus \delta^n$, it can be quantified as following:

$$\Phi_{\boldsymbol{\lambda}} = \sum_{m=1}^{256} \left(\left(1 - \frac{\kappa\left(\Delta^m \oplus \lambda\right)}{4}\right) - \left(1 - \frac{\kappa\left(\Delta^m\right)}{4}\right)\right)^2. \tag{20}$$

The smaller $\Phi_{\boldsymbol{\lambda}}$ is, the less confidence we have on $\boldsymbol{\delta^n} = \boldsymbol{\delta^*}$. Here we generate a λ-to-$\Phi_{\boldsymbol{\lambda}}$ mapping table, and get another constant sequence $\boldsymbol{\lambda} = \left(\lambda^1, \lambda^2, \ldots, \lambda^{256}\right)$ after sorting $\Phi_{\boldsymbol{\lambda}}$ in ascending order. Based on this, puzzled candidate sequence is written as $\boldsymbol{\delta^p} = \delta^* \oplus \left(\lambda^1, \lambda^2, \ldots, \lambda^{256}\right)$. Here $\delta^* \oplus \lambda^1$ equals to δ^*, $\delta^* \oplus \lambda^2$ is the most puzzled candidate, $\delta^* \oplus \lambda^3$ is the next and so on. A part of the constant sequence $\boldsymbol{\lambda}$ is given in Table 3, and the sorted λ-to-$\Phi_{\boldsymbol{\lambda}}$ mapping table

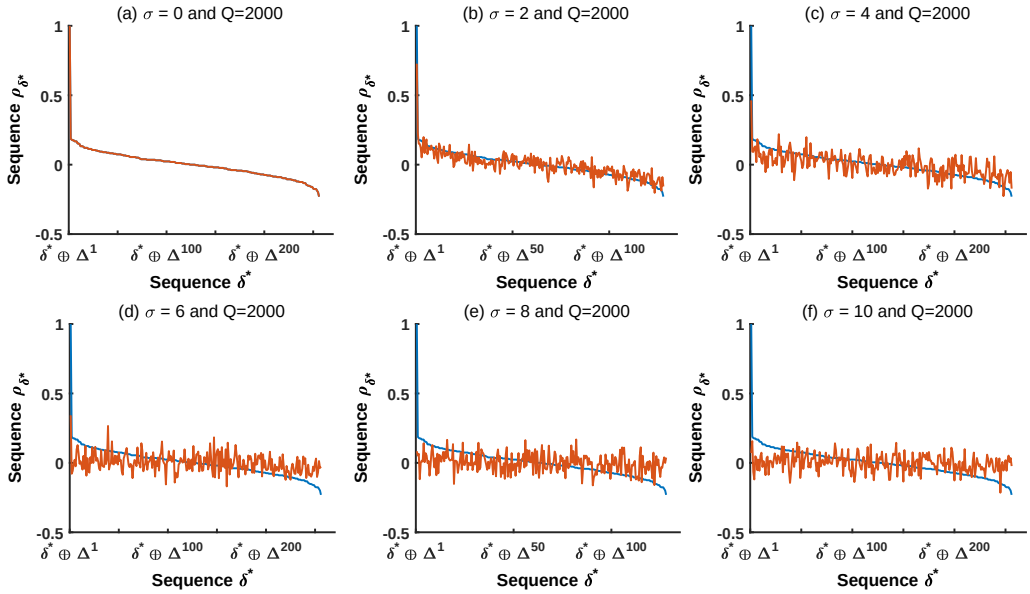

Figure 4: Sequence $\boldsymbol{\rho_{\delta^*}}$ under different noise levels.

is given in Figure 5. Obviously, all Φ_λ-s fall between 3.2764 and 7.8094 with mean $\mu = 5.5557$ and variance $\sigma_2 = 0.8824$ except Φ_{λ^1} corresponding to the correct assumption.

$\boldsymbol{\lambda}$	λ^1	λ^2	λ^3	λ^4	λ^5	$\cdots$	λ^{255}	λ^{256}
Value	0	62	156	184	202	$\cdots$	167	245

Table 3: Constant sequence $\boldsymbol{\lambda}$ for AES-128.

The sequence $\boldsymbol{\Delta}$ and sequence $\boldsymbol{\lambda}$ look similar. To illustrate their difference, we use sequence $\boldsymbol{\lambda}$ to subtract sequence $\boldsymbol{\Delta}$, and a simple comparison of them is shown in Figure 6. It is obvious that the difference rolls up and down around 0 randomly. Actually, these two sequences have very different meanings and are suitable for different cases.

5.2 Deviation from Sequence Δ-s in CLD

The deviation in our PLD given in Section 5.1 exploits different Δ in a specific order (of sequence $\boldsymbol{\Delta}$). Therefore, instead of relying on the specific order observed in noiseless scenario, one can investigate in which exact order different Δ-s actually appear in noisy scenario. Similarly, we can derive the following sequence $\boldsymbol{\Delta_n} = \left(\Delta_n^1, \Delta_n^2, \ldots, \Delta_n^{256}\right) = \delta^n \oplus \left(\delta^1, \delta^2, \ldots, \delta^{256}\right)$ under the assumption that δ^n is the δ^*. In this case, sequence $\boldsymbol{\Delta_n}$ will be similar to sequence $\boldsymbol{\Delta}$ to some extent if δ^n is actually the δ^*. To better quantify this similarity, we turn to their equivalent sequences $\boldsymbol{\kappa}(\boldsymbol{\Delta})$ and $\boldsymbol{\kappa}(\boldsymbol{\Delta_n})$. It is indisputable that sequence $\boldsymbol{\kappa}(\boldsymbol{\Delta_n})$ is a reordered version of sequence $\boldsymbol{\kappa}(\boldsymbol{\Delta})$. However, unlike the above sequences $\boldsymbol{\delta^n}$ and $\boldsymbol{\delta^*}$, both sequences $\boldsymbol{\Delta}$ and $\boldsymbol{\Delta_n}$ are known. Thus, our second optimization named CLD exploits the deviation between $\boldsymbol{\Delta_n}$ and $\boldsymbol{\Delta}$ to detect the collision value.

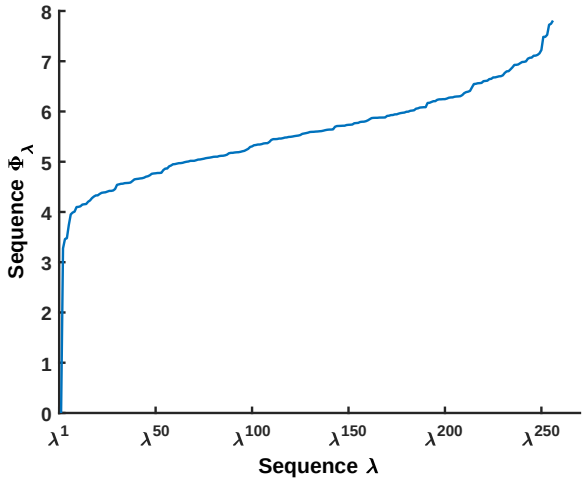

Figure 5: λ-to-$\Phi_{\boldsymbol{\lambda}}$ mapping table in our PLD.

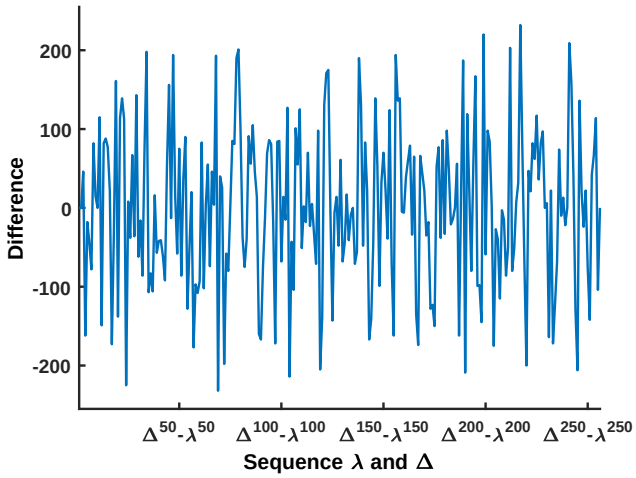

Figure 6: The difference of sequence $\boldsymbol{\lambda}$ subtracting sequence $\boldsymbol{\Delta}$.

Deviations in our CLD consist of two parts as well. The first part is inevitable and closely associated with the deviation given in Section 5.1, for they both are noise-induced. The difference is that, instead of concerning exact thrashed value of every correlation coefficient in $\boldsymbol{\delta^*}$, noise-induced deviations in this method concern thrashed order of $\boldsymbol{\delta^*}$(e.g. order of sequence $\boldsymbol{\delta^{\#}}$), since each value of $\kappa(\Delta)$ stays constant. The second part of deviations in our CLD is caused by the assumption error that $\delta^n = \delta^*$. Both sequences $\boldsymbol{\Delta_n}$ we construct in noisy scenario and the constant sequence $\boldsymbol{\Delta}$ are shown in Table 4. Noise-induced deviations in this Table reflect as whether $\delta^m = \delta^* \oplus \Delta^m$ $(m = 1, 2, \ldots, 256)$ (i.e., $\boldsymbol{\delta^*}$ discussed above in noiseless scenario) in each column, and the second part reflects as whether $\delta^n = \delta^*$.

Figure 7 compares sequence $\boldsymbol{\kappa(\Delta_n)}$ in different noise scenarios with the constant sequence

$\boldsymbol{\Delta}$	$\delta^* \oplus \left(\delta^* \oplus \Delta^1\right)$	$\delta^* \oplus \left(\delta^* \oplus \Delta^2\right)$	...	$\delta^* \oplus \left(\delta^* \oplus \Delta^{256}\right)$
$\boldsymbol{\Delta_n}$	$\delta^n \oplus \delta^1$	$\delta^n \oplus \delta^2$	...	$\delta^n \oplus \delta^{256}$

Table 4: Sequences $\boldsymbol{\Delta}$ and $\boldsymbol{\Delta_n}$.

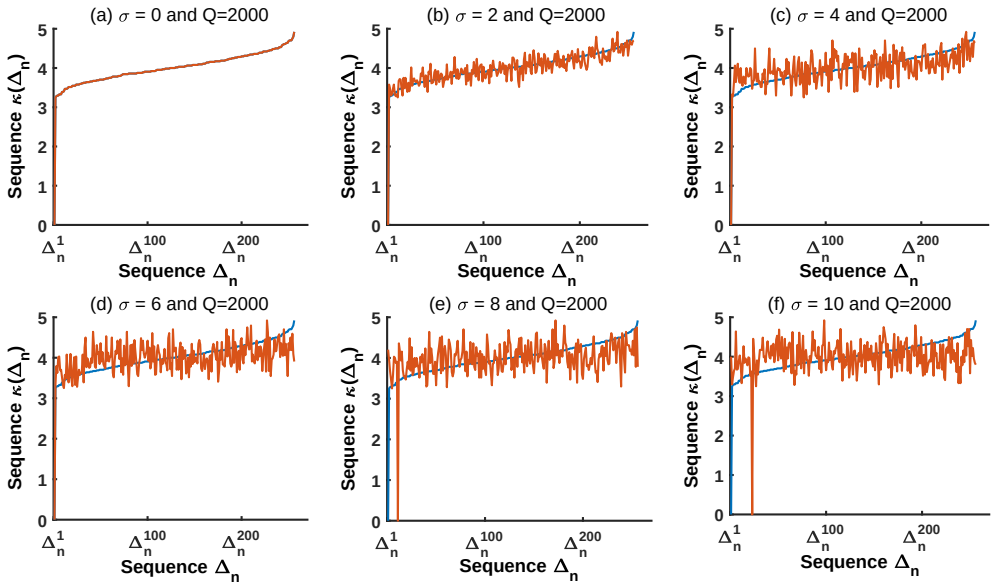

Figure 7: Sequence $\boldsymbol{\Delta_n}$ under different noise levels.

$\boldsymbol{\kappa}\left(\boldsymbol{\Delta}\right)$ in noiseless scenario. Here the orange line represents sequence $\boldsymbol{\kappa}\left(\boldsymbol{\Delta_n}\right)$ and the blue line represents sequence $\boldsymbol{\kappa}\left(\boldsymbol{\Delta}\right)$. For Δ^a $(a = 1, 2, \ldots, 256)$, deviations are caused by another different Δ^b $(b = 1, 2, \ldots, 256$, and $b \neq a)$ taking its position a and thus playing a role of Δ_n^a in sequence $\boldsymbol{\Delta_n}$. The position of Δ^1 is thrashed to 5 and 15 when $\sigma = 8$ and $\sigma = 10$, respectively. We draw a conclusion that correlation coefficients thrash slightly in low noise scenarios and their corresponding positions thrash in a small range as well. These positions thrash violently in a larger range under large noise and enlarge deviations on the whole.

Following the above analysis, deviations from sequence $\boldsymbol{\Delta}$ in our CLD can be quantified by:

$$P_{\boldsymbol{\delta^n}} = \sum_{m=1}^{256} \left(\kappa\left(\Delta^m\right) - \kappa\left(\delta^n \oplus \delta^m\right)\right)^2. \tag{21}$$

$\delta^m = \delta^* \oplus \Delta^m$ is always satisfied in the noiseless cases. Hence, the second part of deviations is quantified as:

$$P_{\boldsymbol{\lambda}} = \sum_{m=1}^{256} \left(\kappa\left(\Delta^m\right) - \kappa\left(\lambda \oplus \Delta^m\right)\right)^2 \tag{22}$$

This equation is essentially the same as Equation (20).

6 Experimental Results

6.1 Simulated Experiments

Our first experiment is performed on the simulated samples of AES-128 generated from the Equation (1), the standard deviation of noise σ is set to 7, 8 and 9, thus the corresponding signal-to-noise ratio is about from 0.025 to 0.041, respectively. We exploit 2000 samples in each repetition and perform a total of 500 repetitions in our experiment. The window *num* , which indicates how many combinations of candidates we decide to keep after attack performed on each sub-key, is set to 1 and 5. The corresponding results of the ad-hoc evaluation function on CECA given by Wiemers et al. in [27], the optimal collision attack given by Cezary et al. in [10], our original Snowball, PLD and CLD-optimized Snowball (labelled as SUM, OPT, Sb_ORI, Sb_PLD and Sb_CLD, respectively) are shown in Figure 8.

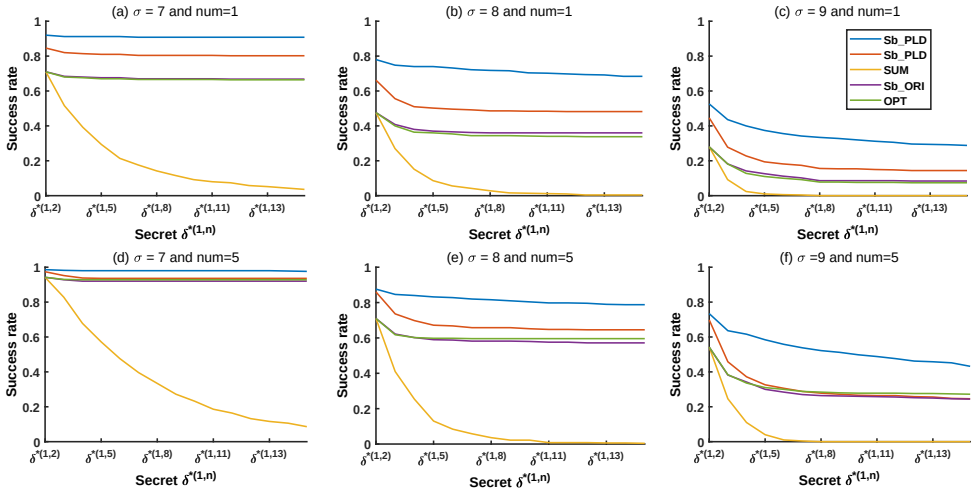

Figure 8: Success rate under different noise levels and different window sizes *num*.

Key recovery in all side-channel attacks including collision attacks means that the combination of all sub-keys must be within the window currently under consideration. In other words, the key recovery fails if any of them falls outside this window. Similarly, collision attacks targeting at $\delta^{*(1,l)}$ are considered successful only if they derive the correct $\delta^{*(1,n)}$ from the correct combination of candidates $\delta^{*(1;2\cdots l)} = \left(\delta^{*(1,2)}, \delta^{*(1,3)}, \ldots, \delta^{*(1,l-1)}\right)$ given by the previous attacks. Therefore, once an attack fails, all of its following attacks can be considered failures at once. Since the combined candidates $\delta^{*(1;2\cdots 16)} = \left(\delta^{*(1,2)}, \delta^{*(1,3)}, \ldots, \delta^{*(1,16)}\right)$ is already impossible to be within the window *num* under consideration.

Firstly, we take the ***single collision value*** $\delta^{*(1,2)}$ between the first two sub-keys as an example, and compare the five attacks on a to illustrate the superiority of our Snowball and its two optimizations PLD and CLD. The success rates of the ad-hoc evaluation function on CECA given by Wiemers et al., the optimal collision attack and our Snowball shown in Figure 8 are always the same when attacking this collision $\delta^{*(1,2)}$, for they are all actually the original CECA in this case. However, our Snowball optimized by CLD and PLD achieves the highest success rates for attacks on this first collision value in all noisy scenarios. This experimental result sufficiently verifies our previous analysis that the strategies of directly maintaining can-

didates with the highest collision correlation coefficients are highly inefficient and inaccurate. By applying our optimizations PLD and CLD on collision correlation coefficients computation, we make use of information on all of them. The collision value $\delta^{*(1,2)}$ can be extracted out with a significantly higher probability, resulting in the corresponding success rates climbing up significantly.

We then compare the five attacks in ***full-key recovery*** to illustrate the superiority of our Snowball and its two optimizations PLD and CLD. The descending tendencies of success rates of our snowball and its two optimizations PLD and CLD become more and more gentle when taking more sub-keys into consideration. This implicitly illustrates the significantly improved accuracy due to the "snowball" effect we design when concentrating the samples of the previous S-boxes to assist attacks on the targeted S-box. It is noteworthy that the optimal collision attack always achieves the same performance as our original Snowball scheme under different scenarios and windows, implying the "optimization" as well. However, the success rates of the ad-hoc evaluation function on CECA given by Wiemers et al. are significantly lower than our Snowball and its two optimizations PLD and CLD after the second collision value $\delta^{*(1,3)}$. Moreover, it decays sharply. We draw a conclusion that the "snowball" effect makes our schemes more advantageous.

6.2 Experiments on an ATmega328p Micro-controller

Our second experiment is performed on an ATmega328p micro-controller with a clock operating frequency of 16 MHz. We implemented an AES-128 algorithm with the same S-box operations (due to code reuse). We randomly encrypt $80,000$ plaintexts and use a WaveRunner 8104 oscilloscope to sample the power traces. The sampling rate is set to 1 GS/s. The leakages of S-boxes are well aligned in this implementation. We perform CPA to extract a Point-Of-Interest (POI) [7] for the first S-box with a correlation coefficient about 0.29. The samples of the remaining 15 S-boxes are well aligned to this first one. We randomly extract 1000 power traces, then compare the ad-hoc evaluation function on CECA given by Wiemers et al. in [27], the optimal collision attack given by Cezary et al. in [10], our original Snowball, PLD and CLD-optimized Snowball (labelled as SUM, OPT, Sb_ORI, Sb_PLD and Sb_CLD, respectively) when the window num is set to 1, 3, 5, 7, 9 and 11, respectively. We run a total of 500 repetitions for each experiment, and the corresponding results are shown in Figure 9.

We take the ***single collision value*** $\delta^{*(1,2)}$ as an example and compare the five attacks, conclusions similar to those given in Section 6.1 can be drawn. The success rates of the ad-hoc evaluation function on CECA given by Wiemers et al., the optimal collision attack and our Snowball in Figure 9 are the same since they all come from the original CECA. However, our Snowball optimized by CLD and PLD achieves the success rates higher than the other attacks. Moreover, PLD exploiting the "values" information are more precise than the "rankings" information exploited in CLD, thus its performance is significantly better. Similar conclusions can be drawn from Section 6.1. This fully illustrates the superiority of our Snowball and its two optimizations PLD and CLD.

We then compare the five attacks in the ***full-key recovery*** scenarios. The experimental results given in Figure 9 fully illustrate that it is far from enough for the ad-hoc evaluation function on CECA given by Wiemers et al. and the optimal collision attack to accumulate collision correlation coefficients and break the independence between sub-keys. Moreover, these strategies result in a waste of valuable information from samples. To maintain the success rates in full-key recovery when attacking the targeted S-box, we must exploit the samples of the previous S-boxes to assist this attack. The descending tendencies of success rates of

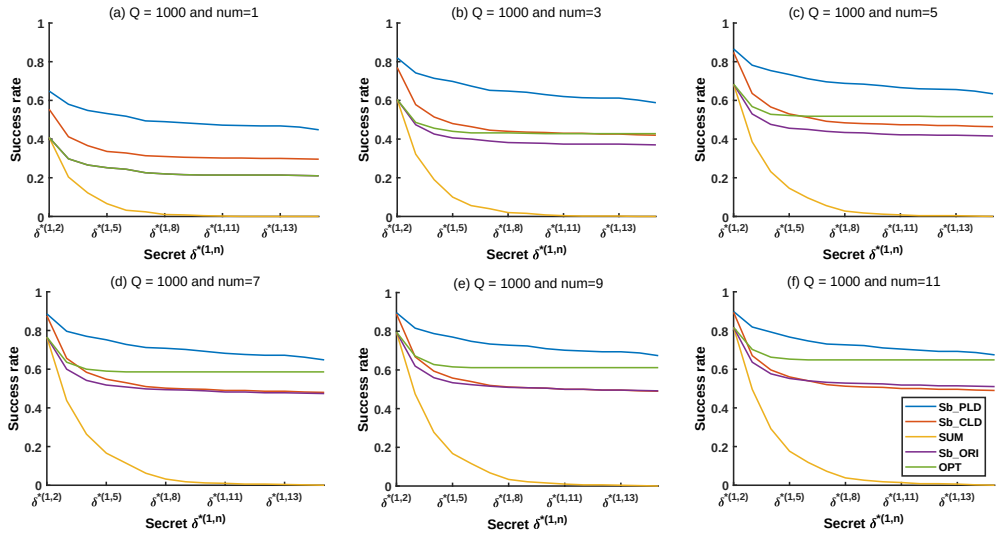

Figure 9: Success rate under different window sizes num.

our Snowball and its two optimizations PLD and CLD become more and more gentle when considering more sub-keys. This fully illustrates the superiority of the "snowball" effect in them.

It's noteworthy that the optimal collision attack is the one most vulnerable to the window num, which usually puts the correct candidates in a position to be considered with a high priority. Therefore, its success rate increases significantly with window num. Our Snowball and its optimization CLD usually push the collision values to the top, so their performance growth is not so obvious when enlarging the window num. Our Snowball optimized by PLD captures more detailed information from "values" of collision correlation coefficients, and its performance is always the best.

7 Conclusions

To improve the performance of the existing side-channel full-key recovery schemes, we considered the similarity of samples of different S-boxes in the code reuse scenarios, exploited the samples from the previously attacked S-boxes to assist attacks on the targeted S-box. This brought us the "snowball" effect, and our Snowball achieved more gentle attenuation trend performance when more S-boxes were under consideration. To further optimize our Snowball, we extended confusion coefficient to collision confusion coefficient, and deduced its relationship with collision correlation coefficient in CECA. Based on this theoretical relationship, We gave two optimizations PLD and CLD exploiting the "values" and "rankings" information of collision correlation coefficients and collision confusion coefficients, respectively. Our experimental results fully illustrated their superiority.

Our Snowball is very simple with strict theoretical proof, and we believe it brings us a new road for efficient full-key recovery. Snowball has strong robustness, and one or several incorrectly guessed collision values will not significantly reduce its performance in full-key recovery.

To further optimize it, we will first introduce a recognition mechanism in our future work to theoretically analyze the impact of wrong collision values on Snowball and identify them. Secondly, we will design the corresponding fault-tolerant strategies so that they can enhance the ability of our schemes to resist such errors. Thirdly, our Snowball's security boundary is also interesting, and we will explore it further and look forward to the "surprises" brought by it. Finally, our Snowball only depends on the confusion coefficient of distinguishers, and the authors in [12] have provided the confusion coefficient expressions of many side-channel distinguishers, this facilitates our attempt to extend snowball to them.

References

[1] Andrey Bogdanov. Improved Side-Channel Collision Attacks on AES. In *Selected Areas in Cryptography, 14th International Workshop, SAC 2007, Ottawa, Canada, August 16-17, 2007, Revised Selected Papers*, volume 4876 of *Lecture Notes in Computer Science*, pages 84–95. Springer, 2007.

[2] Andrey Bogdanov. Multiple-Differential Side-Channel Collision Attacks on AES. In *Cryptographic Hardware and Embedded Systems - CHES 2008, 10th International Workshop, Washington, D.C., USA, August 10-13, 2008. Proceedings*, volume 5154, pages 30–44. Springer, 2008.

[3] Andrey Bogdanov and Ilya Kizhvatov. Beyond the Limits of DPA: Combined Side-Channel Collision Attacks. *IEEE Trans. Computers*, 61(8):1153–1164, 2012.

[4] Eric Brier, Christophe Clavier, and Francis Olivier. Correlation Power Analysis with a Leakage Model. In *Cryptographic Hardware and Embedded Systems - CHES 2004: 6th International Workshop Cambridge, MA, USA, August 11-13, 2004. Proceedings*, pages 16–29, 2004.

[5] Nicolas Bruneau, Claude Carlet, Sylvain Guilley, Annelie Heuser, Emmanuel Prouff, and Olivier Rioul. Stochastic Collision Attack. *IEEE Trans. Inf. Forensics Secur.*, 12(9):2090–2104, 2017.

[6] Suresh Chari, Josyula R. Rao, and Pankaj Rohatgi. Template Attacks. In *Cryptographic Hardware and Embedded Systems - CHES 2002, 4th International Workshop, Redwood Shores, CA, USA, August 13-15, 2002, Revised Papers*, pages 13–28, 2002.

[7] François Durvaux and François-Xavier Standaert. From Improved Leakage Detection to the Detection of Points of Interests in Leakage Traces. In *Advances in Cryptology - EUROCRYPT 2016 - 35th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Vienna, Austria, May 8-12, 2016, Proceedings, Part I*, pages 240–262, 2016.

[8] Thomas Espitau, Pierre-Alain Fouque, Benoît Gérard, and Mehdi Tibouchi. Side-Channel Attacks on BLISS Lattice-Based Signatures: Exploiting Branch Tracing against strongSwan and Electromagnetic Emanations in Microcontrollers. In *Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security, CCS 2017, Dallas, TX, USA, October 30 - November 03, 2017*, pages 1857–1874, 2017.

[9] Yunsi Fei, A. Adam Ding, Jian Lao, and Liwei Zhang. A statistics-based success rate model for DPA and CPA. *J. Cryptogr. Eng.*, 5(4):227–243, 2015.

[10] Cezary Glowacz and Vincent Grosso. Optimal Collision Side-Channel Attacks. In *Smart Card Research and Advanced Applications - 18th International Conference, CARDIS 2019, Prague, Czech Republic, November 11-13, 2019, Revised Selected Papers*, volume 11833 of *Lecture Notes in Computer Science*, pages 126–140. Springer, 2019.

[11] Vincent Grosso. Scalable Key Rank Estimation (and Key Enumeration) Algorithm for Large Keys. In *Smart Card Research and Advanced Applications, 17th International Conference, CARDIS 2018, Montpellier, France, November 12-14, 2018, Revised Selected Papers.*, pages 80–94, 2018.

[12] Sylvain Guilley, Annelie Heuser, and Olivier Rioul. A key to success - success exponents for side-channel distinguishers. In Alex Biryukov and Vipul Goyal, editors, *Progress in Cryptology - INDOCRYPT 2015 - 16th International Conference on Cryptology in India, Bangalore, India,*

December 6-9, 2015, Proceedings, volume 9462 of *Lecture Notes in Computer Science*, pages 270–290. Springer, 2015.

[13] Paul C. Kocher. Timing Attacks on Implementations of Diffie-Hellman, RSA, DSS, and Other Systems. In *Advances in Cryptology - CRYPTO '96, 16th Annual International Cryptology Conference, Santa Barbara, California, USA, August 18-22, 1996, Proceedings*, pages 104–113, 1996.

[14] Hervé Ledig, Frédéric Muller, and Frédéric Valette. Enhancing Collision Attacks. In *Cryptographic Hardware and Embedded Systems - CHES 2004: 6th International Workshop Cambridge, MA, USA, August 11-13, 2004. Proceedings*, pages 176–190, 2004.

[15] Yang Li, Shuang Wang, Zhibin Wang, and Jian Wang. A Strict Key Enumeration Algorithm for Dependent Score Lists of Side-Channel Attacks. In *Smart Card Research and Advanced Applications - 16th International Conference, CARDIS 2017, Lugano, Switzerland, November 13-15, 2017, Revised Selected Papers*, pages 51–69, 2017.

[16] Moritz Lipp, Andreas Kogler, David F. Oswald, Michael Schwarz, Catherine Easdon, Claudio Canella, and Daniel Gruss. PLATYPUS: software-based power side-channel attacks on x86. In *42nd IEEE Symposium on Security and Privacy, SP 2021, San Francisco, CA, USA, 24-27 May 2021*, pages 355–371. IEEE, 2021.

[17] Keyu Man, Zhiyun Qian, Zhongjie Wang, Xiaofeng Zheng, Youjun Huang, and Haixin Duan. DNS cache poisoning attack reloaded: Revolutions with side channels. In Jay Ligatti, Xinming Ou, Jonathan Katz, and Giovanni Vigna, editors, *CCS '20: 2020 ACM SIGSAC Conference on Computer and Communications Security, Virtual Event, USA, November 9-13, 2020*, pages 1337–1350. ACM, 2020.

[18] Keyu Man, Xin'an Zhou, and Zhiyun Qian. DNS cache poisoning attack: Resurrections with side channels. In Yongdae Kim, Jong Kim, Giovanni Vigna, and Elaine Shi, editors, *CCS '21: 2021 ACM SIGSAC Conference on Computer and Communications Security, Virtual Event, Republic of Korea, November 15 - 19, 2021*, pages 3400–3414. ACM, 2021.

[19] Ahmad Moghimi, Gorka Irazoqui, and Thomas Eisenbarth. CacheZoom: How SGX Amplifies the Power of Cache Attacks. In *Cryptographic Hardware and Embedded Systems - CHES 2017 - 19th International Conference, Taipei, Taiwan, September 25-28, 2017, Proceedings*, pages 69–90, 2017.

[20] Amir Moradi, Oliver Mischke, and Thomas Eisenbarth. Correlation-Enhanced Power Analysis Collision Attack. In *Cryptographic Hardware and Embedded Systems, CHES 2010, 12th International Workshop, Santa Barbara, CA, USA, August 17-20, 2010. Proceedings*, pages 125–139, 2010.

[21] Maxime Nassar, Youssef Souissi, Sylvain Guilley, and Jean-Luc Danger. RSM: A Small and Fast Countermeasure for AES, Secure against 1st and 2nd-Order Zero-Offset SCAs. In *2012 Design, Automation & Test in Europe Conference & Exhibition, DATE 2012, Dresden, Germany, March 12-16, 2012*, pages 1173–1178, 2012.

[22] Romain Poussier, François-Xavier Standaert, and Vincent Grosso. Simple Key Enumeration (and Rank Estimation) Using Histograms: An Integrated Approach. In *Cryptographic Hardware and Embedded Systems - CHES 2016 - 18th International Conference, Santa Barbara, CA, USA, August 17-19, 2016, Proceedings*, pages 61–81, 2016.

[23] Kai Schramm, Gregor Leander, Patrick Felke, and Christof Paar. A Collision-Attack on AES: Combining Side Channel- and Differential-Attack. In *Cryptographic Hardware and Embedded Systems - CHES 2004: 6th International Workshop Cambridge, MA, USA, August 11-13, 2004. Proceedings*, pages 163–175, 2004.

[24] Kai Schramm, Thomas J. Wollinger, and Christof Paar. A New Class of Collision Attacks and Its Application to DES. In *Fast Software Encryption, 10th International Workshop, FSE 2003, Lund, Sweden, February 24-26, 2003, Revised Papers*, volume 2887 of *Lecture Notes in Computer Science*, pages 206–222. Springer, 2003.

[25] François-Xavier Standaert, Tal Malkin, and Moti Yung. A Unified Framework for the Analysis

of Side-Channel Key Recovery Attacks. In *Advances in Cryptology - EUROCRYPT 2009, 28th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Cologne, Germany, April 26-30, 2009. Proceedings*, volume 5479 of *Lecture Notes in Computer Science*, pages 443–461. Springer, 2009.

[26] Weijia Wang, Yu Yu, François-Xavier Standaert, Junrong Liu, Zheng Guo, and Dawu Gu. Ridge-Based DPA: Improvement of Differential Power Analysis For Nanoscale Chips. *IEEE Trans. Information Forensics and Security*, 13(5):1301–1316, 2018.

[27] Andreas Wiemers and Dominik Klein. Entropy Reduction for the Correlation-Enhanced Power Analysis Collision Attack. In *Advances in Information and Computer Security - 13th International Workshop on Security, IWSEC 2018, Sendai, Japan, September 3-5, 2018, Proceedings*, pages 51–67, 2018.