International Journal of Smart Home
Vol. 10, No. 3, (2016), pp.259-270
http://dx.doi.org/10.14257/ijsh.2016.10.3.25

Second-Order Power Analysis Attacks against Precomputation based Masking Countermeasure

Weijian Li[1] and Haibo Yi[2]

[1]*School of Computer Science, Guangdong Polytechnic Normal University, Guangzhou, China*
[2]*School of Computer Engineering, Shenzhen Polytechnic, Shenzhen, China*
[1]*weijianlee@126.com,* [2]*haiboyi@126.com*

Abstract

Precomputation look-up table based masking countermeasure is low-cost and secure against first-order DPA, therefore is more suitable for lightweight ciphers in resource-constrained devices. In this paper, we investigate the resistance of this masking countermeasure against second-order power analysis attack under the attack context of the Hamming weight leakage and the precomputation masked S-box. We improve the Adapted CPA technique [1] to make a better use of this attack context. Our attack successfully reveals the secret key with and without electronic noise and algorithmic noise. The number of power traces required to reveal the secret key rises from 600(unprotected implementation) to 16,000.

Keywords: *Second-order SCA; Precomputation based masking; Adapted CPA; Lightweight cipher*

1. Introduction

Over the past 15 years, masses of researchers have focused on physical security of block ciphers, for it is obvious that the unprotected implementations of block ciphers can be broken by side channel attacks. Differential Power Analysis is one of the effective methods to retrieve secret keys, which including mono-bit DPA [2], multibit DPA [3-4] and Correlation Power Analysis (CPA) [5-6].

Masking technique is the most popular technique to prevent cryptographic algorithm from power analysis attacks, which is done by XORing intermediate value with random value to randomize the intermediate values [7-10]. Masking method is known to be low-cost and secure against first-order DPA.

To construct the masking scheme, the most important consideration has been to mask the *S-box* operation. Commonly, there are two strategies[10]:

1. **The precomputation look-up table** [7-10]: this method recomputes the masked *S-box* as a precomputed look-up table, and stores it in RAM or ROM. The look-up table $MS\text{-}box(A \oplus X;X) = S\text{-}box(A) \oplus q$ is precomputed according to the intermediate value A and random value X. The value of q could be different for different masking schema, and is equal to X in a simplified version, which is sufficient to prevent from first-order SCA.

2. **The *S-box* secure calculation**[8-9]: the *S-box* outputs are computed on-the-fly by using a mathematical (*e.g.* polynomial) representation of the *S-box*. Each time the masked value has to be computed, an algorithm is executed. The computation of algorithm is split into elementary operations (bit-wise addition, bit-wise multiplication, *etc.*,) performed by accessing one or several look-up table(s).

However, since the first successful attack appeared in [11], many publications have shown that software implementation (*e.g.,* smartcard) of masking countermeasures are still susceptible to high-order power analysis attacks. Meanwhile there are numerous works investigating the security of hardware implementation of masking

ISSN: 1975-4094 IJSH

countermeasures. Most high-order power analysis attacks target at the masked *S-box* as secure calculation [12-13]. It has been first shown in [12] that CMOS implementation of secure calculation *S-box* can be attacked because of glitches. All these discussions lead to the conclusion that glitches in masked circuits pose the biggest threat to masked hardware implementations in practice.

Otherwise, there are so few works focus on masked *S-box* as precomputation look-up table. Complementing the work in [14], which suggested that higher-order attacks are possible without any additional hypothesis than usually assumed for first-order attacks, Standaert *etc*., [1] proposed an improved technique that can be viewed as the higher-order Correlation Power Analysis, called Adapted CPA. They found that the average of square power consumptions was dependent on the masked intermediate value, and adapted CPA was able to make a better use of the power consumption measures to reveal the secret key. This method focused on the Hamming-distance leakage and the masked *S-box* described in figure 1(b), where $MS\text{-}box(A \oplus X;X) = S\text{-}box(A) \oplus q$ and $q = S\text{-}box(A \oplus X)$.

In this paper, we will investigate the resistance of masking countermeasure against second-order power analysis attack under the attack context of the Hamming weight leakage and the masked *S-box* described in figure 1(a), where $MS\text{-}box(A \oplus X;X) = S\text{-}box(A) \oplus q$ and $q = X$. This masked *S-box* is brief, low-cost and sufficient to prevent from first-order SCA, therefore is suitable for lightweight ciphers in resource-constrained devices. To demonstrate how masked *S-box*es work and the experimental results, we will take KLEIN[15] for example.

The rest of this paper is organized as follows. Section 2 describes the masking countermeasures, especially the masked *S-box* that our second-order power analysis attack target at. In Section 3, we propose the second-order power analysis attack against masking countermeasure. Experimental results are given in Section 4. Section 5 concludes the paper.

2. Masking Countermeasures

For power analysis attacks, it is feasible because of the dependency between the power consumption of devices and the intermediate values of the cryptographic algorithms. There- fore, this dependency should be broken to prevent from power analysis attacks, which can be done by randomizing the power consumption of the device. The precomputation look-up table method is one of the most common ways to mask the intermediate values in cryptographic algorithm, which recomputes a precomputed table according to the *S-box* and the generation of one or several random value(s).

There are linear and non-linear functions within cryptographic algorithms[16]. A linear function f has the property that $f(x \oplus y)=f(x) \oplus f(y)$, otherwise non-linear functions $f(x \oplus y) \neq f(x) \oplus f(y)$. For most symmetric cryptographic algorithm, almost all of the operations are linear functions, the only non-linear operation is *S-box*: $S-box(x \oplus y) \neq S-box(x) \oplus S-box(y)$. Therefore, to construct the masking scheme, the most important consideration has been to mask the *S-box* operation. Masking countermeasures should rewrite the *S-box*, the precomputed look-up table method recomputes a precomputed table according to the *S-box* and the generation of one or several random value(s). Figure 1(a) and 1(b) show two kinds of precomputed table for masking *S-box*. The precomputed look-up table $MS\text{-}box(A \oplus X;X) = S\text{-}box(A) \oplus q$, where masked intermediate value $A = m \oplus k$, $q = X$ for Figure 1(a) and $q = S\text{-}box(A \oplus X)$ for Figure 1(b).

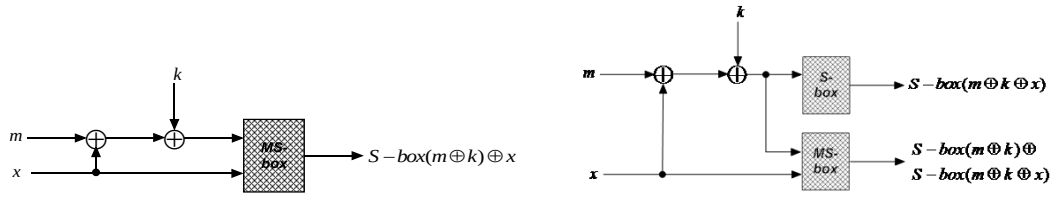


Figure 1. Two Kinds of Masked *S-box*es

In order to explain how masked *S-box*es work, we take KLEIN for example. Figure 2(a) and Figure 2(b) describe two structures of masking countermeasures corresponding to the masked *S-box* in Figure 1(a) and 1(b). Since our attack targets at the masking countermeasure in Figure 2(a), we explain this structure detailedly below.

A random 64/80/96-bit mask X will be XORed with the plaintext at the beginning of the algorithm, then the algorithm starts with the input value $A = M \oplus X$. It only needs to compute the value of mask at a fixed step (for example at the beginning of algorithm) and reestablishes the expected value at the end of the algorithm.

Figure 2(a) shows the differences between a KLEIN with and without countermeasure, where:

- A is the input of each round, equal to the plaintext XORed with key when first round;

- X is the random mask;

- B, C and D are the outputs of SubNibbles, RotateNibbles and MixNibbles respectively;

- *MS-box* is the masked *S-box* in Figure 1(a) that *MS-box*($A \oplus X$;X) = *S-box*(A) $\oplus$ X, where *S-box* is the original *S-box* of KLEIN;

- X_1 = RotateNibbles(X); X_2 = MixNibbles(X_1).

Remark: All of *MS-box*, X_1 and X_2 can be precomputed only once at the beginning of algorithm. According to the definition of *MS-box*, *MS-box*($A \oplus X$;X) = *S-box*(A) $\oplus$ X = $B \oplus X$. And because of the property of linear function, RotateNibbles($B \oplus X$) = RotateNibbles(B) $\oplus$ RotateNibbles(X) = $C \oplus X_1$, MixNibbles($C \oplus X_1$) = MixNibbles(C) $\oplus$ MixNibbles(X_1) = $D \oplus X_2$. At the end of every round, the output will be XORed with X_2 to reestablish the expected value of E, and XORed X again to have the masked round output $E \oplus X$. At the end of algorithm, the output will be XORed with X to reestablish the cipher.

3. Second-Order Power Analysis Attacks

In this section we will describe our second-order power analysis attack against precomputed look-up table masking countermeasure in Figure 2(a), according to [1].

As shown in Figure 1(a), plaintext m is XORed with the random mask x and key k, after that both the mask x and the masked data are sent into a non-linear *MS-box*. *MS-box* is a precomputed table that *MS-box*(x;$m \oplus k \oplus x$) = *S-box*($m \oplus k$) $\oplus$ x, *S-box* is the original *S-box* of cryptographic algorithm.

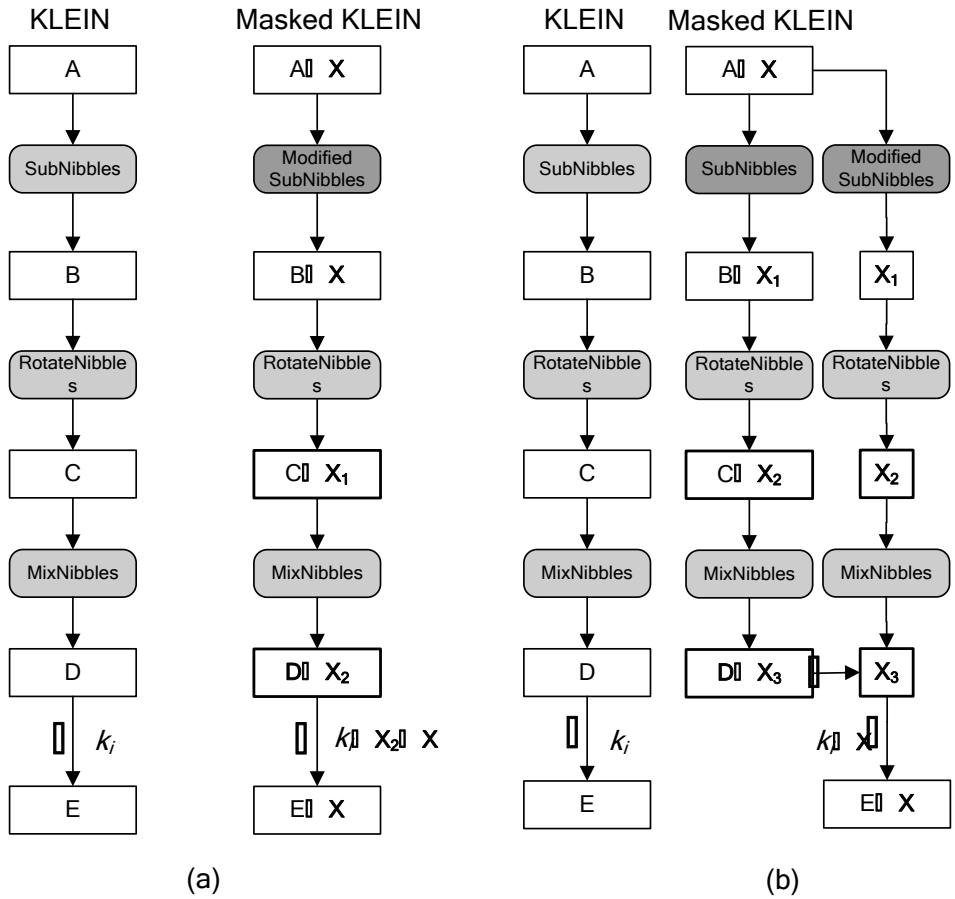


Figure 2. Two Structures of Precomputed Look-up Table Masking Countermeasures

We implement *MS-box* as a precomputed look-up table with 8 bits input(4 bits of mask x and 4 bits of masked intermediate data $x \oplus m \oplus k$ combined together), and 4 bits of output equal to *S-box*$(m \oplus k) \oplus x$. Power consumption of *MS-box* depends on the mask x and the masked data $x \oplus m \oplus k$ that it processes. According to the Hamming-weight power model, power consumption of *MS-box* $P_{MS-box} = \varepsilon \cdot (HW(x) + HW(x \oplus m \oplus k)) + L$. Since random value x is unknown, an adversary can not predict the power consumption of the *MS-box*. Therefore an adversary must find another dependence between the key and power consumption of the *MS-box* to reveal the key.

To simplify our discussion, we consider only the single bit input of *MS-box*. Table 1 shows the power consumptions of *MS-box* with different input of *m*, *k* and *x*.

As shown in column 5 of Table 4, a first-order Power Analysis Attack is infeasible because the power consumptions are nothing different in the case of $p \oplus k = 0$ and $p \oplus k = 1$. However, quadratic mean of power consumptions in column 6 are obviously dependent on value of $p \oplus k$. In other words, if there are two power consumptions $P^{(1)}$, $P^{(2)}$ with $p \oplus k = 0$, the mask m_1 of $P^{(1)}$ is 0, meanwhile the mask m_2 of $P^{(2)}$ is 1, we have $(P^{(1)} + P^{(2)})^2 / 2 = P(p \oplus k = 0)^2 / 2 = 2(P_0^2 + P_1^2)$.

Table 1. Power Consumption of *MS-box*

$p \oplus k$	m	$p \oplus k \oplus m$	$P(MS-box(m, p \oplus k \oplus m))$	$Mean(P)$	$Mean(P^2)$
0	0	0	$2P_0$	P_0+P_1	$2P_0^2 + 2P_1^2$
0	1	1	$2P_1$		
1	0	1	P_0+P_1	P_0+P_1	$(P_0 + P_1)^2$
1	1	0	P_0+P_1		

Since mask x is random with uniform distribution, when generating a large enough number of x, it will tend to be one half of xs equal to 0, the other half equal to 1. We can conclude our discussion by the equation (1):

$$\begin{cases} \lim_{n_1 \to \infty} \left(\frac{1}{n_1} \sum_{b \oplus k = 0, i=1}^{n_1} P^2\right) = 2(P_0^2 + P_1^2) \\ \lim_{n_2 \to \infty} \left(\frac{1}{n_2} \sum_{b \oplus k = 1, j=1}^{n_2} P^2\right) = (P_0 + P_1)^2 \end{cases} \quad (1)$$

Quadratic means of power consumptions of single bit *MS-box* are different and dependent on the value of $m \oplus k$. When considering an 8 bits *MS-box*, quadratic means of power consumptions are dependent on the number of bits of $m \oplus k$ equal to 0.

In order to perform a CPA attack with such leakage model, we adopt and improve the adapted CPA method [1] to make a better use of our attack context. Let s be the bit size of original *S-box*, the precomputed table *MS-box* has the size of $2^{2s} \times s$ bits. The attack is performed iteratively against one after one *MS-box*, therefore number of key candidates is 2^s. Our adapted CPA attack holds in three steps.

Algorithm 1 Precomputation

1) for key guess $k = 0 : 2^s-1$
  2) for input $m = 0 : 2^s-1$
    sum = 0;
    3) for mask $x = 0 : 2^s-1$
      /*Gather statistics of quadratic sum of Hamming-weight for each input key guess k, plaintext m and all possible masks x of target *MS-box*:*/
        sum = sum + (HW(x) + HW($m \oplus k \oplus x$))^2
    end 3);
    /*Predict the quadratic mean of power consumptions for key guess k and input m of target *MS-box*:*/
    precomputation[k;m] = sum/2^s;
  end 2);
end 1)

1. Precomputation: An adversary first gathers statistics of quadratic sum of Hamming weight for each input key guess k, plaintext m and all possible masks x of target *MS-box*.

He predicts the quadratic mean of power consumptions for each key guess k and input m of target *MS-box*, because power consumption of *MS-box* is linear dependent on the number of bits of $m \oplus k$ equal to 0. Pseudo-code is given in Algorithm 1, where *HW* function is the Hamming weight function. Result of precomputation phase is stored in a 2^{2s} precomputation matrix.

2. Measurement: During this phase, the adversary computes exactly the same means as during the precomputation phase, with two significant differences. First, the average is made on the real, measured, squared power consumptions. Secondly, since the masks x are unknown, according to equation 1, the adversary should run the encryption algorithm for each plaintext and the same key large enough times, denoted as n. This coefficient is an important parameter of the attack and increases in case of noisy measurements. Pseudo-code of this process is shown in Algorithm 2. P(targetdevice) is the power trace vector of each encryption, with sp sample points.

Algorithm 2 Measurement

1) for input m = 0 : 2^s-1

sum = 0;

2) for t = 0 : n-1

/* Measure and average the square of the power consumptions for input m of target *MS-box*: */

3) for i = 0 : sp-1

sum[i] = sum[i] + (P(targetdevice)[m][t][i])^2

end 3);

end 2);

measurement[m; i] = sum[i] / n;

end 1);

Result of the measurement phase is stored in a $2^s \times sp$ measurement matrix, where sp is the number of samples for each power trace.

3. CPA: During the last phase, the adversary compares the Pearson correlations between precomputation matrix and measurement matrix column by column. Since the adversary has no idea about the exact position of the power traces where *MS-box* is executed, he can perform attack for every sample position of the power traces. There will be a peak in the CPA trace for the right key guess and right sample position. Pseudo-code of this process is described in Algorithm 3.

The adapted CPA attack makes use of the leakage model described in equation 1, and has the ability to reveal the secret key of the precomputed look-up table masked KLEIN. We will give the successful experimental results in the next section.

4. Preprocessing: Using the second-order power analysis attack method we have described in Algorithm 1,2 and 3, we find it works for a single masked *S-box*(section 4.1), but does not work for a full version of masking countermeasure. The attack method can not reveal.

Algorithm 3 CPA

1) for key guess $k = 0 : 2^s-1$

2)for $i = 0 : sp-1$

/*Compare the Pearson correlations between precomputation matrix and measurement matrix for each candidate key k and all possible sample positions*/

cortmp(i) = corrcoef(precomputation[k; :]; measurement[:; i]);

end 2);

cor(k) = max(cortmp);

end 1);

candidate key = indexofmax(cor);

any secret key even when the number of power traces increases to 160,000 without any noise. The reason could be that the Measurement in Algorithm 2 squares the power consumptions and greatly increases the algorithmic noise. Therefore we add a preprocess shown in Algorithm 4 to the power traces before Algorithm 2. Experimental result (Section 4.3) shows that it can effectively reduces the algorithmic noise.

Algorithm 4 Preprocessing

1) for input m = $0 : 2^s-1$

/* Average power consumptions of each input m */

avgPower = mean(P(targetdevice)[m][:]);

2) for counter = $0 : n-1$

P(targetdevice)[m][counter] = P(targetdevice)[m][counter]- avgPower;

end 2);

end 1);

4. Experimental Results

In practical attacks, there exist two kinds of noise, non-algorithmic noise and algorithmic noise[4]. Non-algorithmic noise is a random fluctuation in an electrical signal generated by all electronic devices, including external, intrinsic, sampling and quantization noise. Non-algorithmic noise is usually considered as a white noise. When a power consumption of a fixed operation on some fixed data is repeated, the measure is different for every repetition. Algorithmic noise is due to the variation of the data bytes being processed by the device. Increasing the number of *S-box* in the design will increase the algorithmic noise.

Since power analysis attacks are mainly based on the power traces, Signal to Noise Ratio (SNR) may significantly influence the result of the key guess. The attacks may be perturbed if undesirable noise is important. Therefore, in order to demonstrate that the techniques we have described in this work perform well, we will first evaluate the exactitude of second-order attack without any noise, namely non-algorithmic noise and algorithmic noise. Secondly, we will evaluate the

exactitude of second-order attack with non-algorithmic noise. Lastly we will evaluate the exactitude of second-order attack with algorithmic noise.

In order to perform experiments described above and compare for different noise levels, we carefully simulated the deterministic power consumptions in Synopsys PrimePower using dedicated power simulation libraries and added Gaussian nose of different amplitudes to it. PrimePower is a dynamic, full-chip power analysis tool for complex multimillion-gate designs. Its high-capacity power analysis includes gate-level average and peak power verification. PrimePower supports industry-standard synthesis libraries and comprises a powerful and flexible methodology that is fully integrated with existing design flows. It provides a high degree of accuracy, performance, ease of use and comprehensive power diagnostics. The synthesis library we use is TSMC 0.18 μm Process 1.8-Volt SAGE-X Standard Cell Library, which appear in many published papers.

4.1. Single *MS-box* without Noise

To verify the validity of our attack method, we first perform our second-order attack against masked S-box without any noise, *i.e.*, without electronic noise and algorithmic noise. Power consumptions generated by Primepower are electronic noise-free. To attain power consumptions without algorithmic noise, we implement a verilog version of single *MS-box* with input of plaintext *m*, secret key *k* and mask *x*, each of which is 4 bits.

According to Algorithm 2, we generate all possible plaintext $0 \le m \le 15$. And because the masks *x* are unknown, we have to run the encryption algorithm with same key and plaintext large enough time *n*, ranging 1 to 1000. Since the parameter *n* significantly influence the result of the attack, we will discuss the attack result with different *n*.

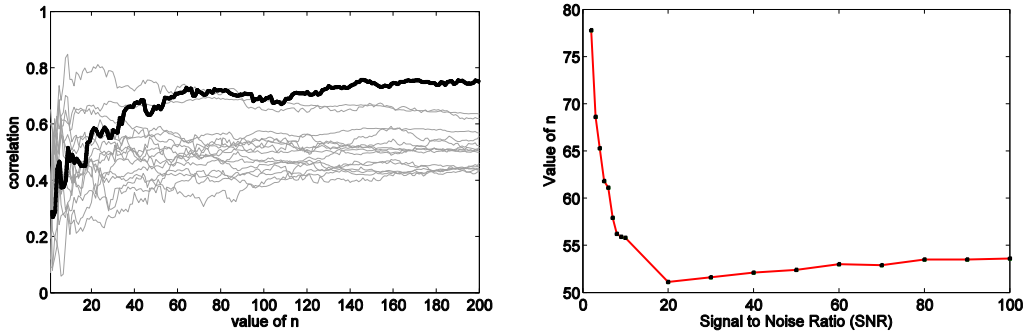
The result is shown in Figure 4(a), where horizontal ordinate represents the values of n, vertical coordinate represents the correlations of adapted CPA. It is clearly observed that with $n \ge 53$, our attack correctly reveal the secret key, which means that our attack requires $n \times 16 = 848$ power traces.

4.2. Single *MS-box* with White Noise

In this subsection, we will evaluate the exactitude of second-order attack with electronic noise, which is the case for the real-world measurements of power consumption. The simulated traces with added white noise are used for an initial analysis of the efficiency of our attack techniques.

The power traces of experiment 1 are added with random noise due to normal distribution with the zero mean value and a standard deviation σ whose value characterizes the noise amplitude. Figure 3(a) shows our attack result with SNR equal to 3.

To evaluate the influence of electronic noise in terms of Signal to Noise Ratio(SNR), we perform our attack 1,000 times and average the values of n for each SNR. The reason why we perform our attack 1,000 times for each SNR is that the white noise is randomly generated, and is different every time, which can lead to different attack result of *n*. Figure 3(b) illustrates the value of n required to distinguish the right CPA.



(a) Correlation with SNR Equal to 3 (b) Number of Plaintexts Required with Different SNR

Figure 3. Second-Order Attack against One Single *MS-box* with Electronic Noise

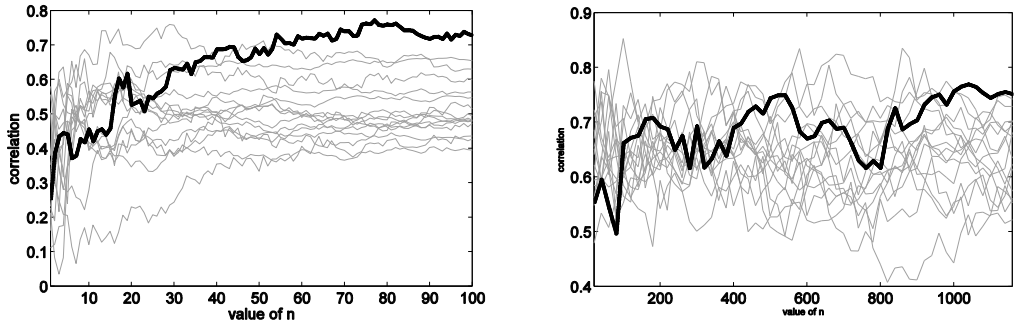
trace(black) from other traces(gray) with different SNR. The value of n rises from 53 to about 78, which means the number of power traces required rises from 848 to about 1248.

4.3. Attack against Masked KLEIN

In this subsection, we will discuss a more practical issue, attacking not only a single masked *S-box*, but a full version of masking countermeasure. To simplify our discuss, we only choose the KLEIN-64 version, which has both 64 bit plaintext and secret key. The masked KLEIN-64 is described in Figure 1(a) and figure 2(a). It is similar to attack other masked block ciphers that have the same masked *S-box*.

Similar to what we have done in experiment 1, we generate all possible half byte of plaintexts m, $0 \le m \le 15$, and run the encryption algorithm n = 10,000 times for each plaintext with the same key k = 9, measuring and storing all power traces. We will discuss the attack result against full version of masked KLEIN with different $n \le 10,000$.

With the second-order power analysis attack method we have described in Section 3, we find it works for a single masked *S-box*, but does not work for a full version of masked KLEIN. The attack method can not reveal any secret key even when the number of power traces increases to 160,000 without any noise. The reason could be that the Measurement in Algorithm 2 squares the power consumptions and greatly increases the algorithmic noise. Therefore we add a preprocess shown in Algorithm 4 to the power traces before Algorithm 2.



(a)Second-order Attack Against One Single MS-box

(b) Second-order Attack Against Masked KLEIN

Figure 4. Second-Order Attack against MS-box and Masked KLEIN

The experimental result is shown in Figure 4(b), when $n \geq 1000$, our attack correctly reveal a 4 bits secret key, which means that our attack requires $n \times 16 =$ 16,000 power traces.

5. Conclusion

Unprotected implementations of block ciphers can be broken by side channel attacks, and masking technique is the most popular technique to prevent from first-order SCA. Precomputation look-up table method is one of the masking countermeasures which is inexpensive and secure against first-order DPA, therefore is more suitable for lightweight ciphers in resource-constrained devices. There are so few works focus on masked S-box as precomputation look-up table. According to the work in [1], we improve the Adapted CPA and perform attack against masking countermeasure with the Hamming-weight leakage and the masked S-box described in Figure 1(a). Using the deterministic power consumptions generating by Synopsys Primepower, we investigate the resistance of the precomputation look-up table masking countermeasure against second-order power analysis attack. Our attack successfully reveals the secret key with and without electronic noise and algorithmic noise. The number of power traces required to reveal the secret key rises from 600 (unprotected implementation) to 16,000 (masking countermeasure).

Acknowledgments

This work is Supported by Foundation for Distinguished Young Talents in Higher Education of Guangdong, China (No.2014KQNCX177), and Quality Engineering Project of Department of Education of Guangdong Province (2014).

References

[1] F.-X. Standaert, E. Peeters, and J. J. Quisquater, “On the masking countermeasure and higher-order power analysis attacks”, Information Technology: Coding and Computing, 2005, ITCC, International Conference, IEEE, vol. 1, **(2005)**, pp. 562-567.
[2] P. Kocher, J. Jaffe and B. Jun, “Differential power analysis”, Advances in Cryptology CRYPTO Springer, no. 99, **(1999)**, pp. 789-789.
[3] T. S. Messerges, E. A. Dabbish and R. H. Sloan, “Investigations of power analysis attacks on smartcards”, In USENIX workshop on Smartcard Technology, vol. 1999, **(1999)**.
[4] E. A. Messerges, T. S. Dabbish and R. H. Sloan, “Examining smart-card security under the threat of power analysis attacks”, Computers, IEEE Transactions, vol. 51, no. 5, **(2002)**, pp. 541-552.
[5] T. H. Le, J. Clédière, C. Canovas, B. Robisson, C. Servière and J. L. Lacoume, “A proposition for correlation power analysis enhancement”, Cryptographic Hardware and Embedded Systems-CHES, **(2006)**, pp. 174-186.

[6] E. Brier, C. Clavier and F. Olivier, "Correlation power analysis with a leakage model", Cryptographic Hardware and Embedded Systems-CHES 2004, **(2004)**, pp. 135-152.
[7] M. L. Akkar and C. Giraud, "An implementation of DES and AES, secure against some attacks", In Cryptographic Hardware and Embedded Systems, CHES 2001, Springer, **(2001)**, pp. 309-318.
[8] E. Oswald, S. Mangard, N. Pramstaller and V. Rijmen, "A side-channel analysis resistant description of the AES S-box", Fast Software Encryption, Springer, **(2005)**, pp. 199-228.
[9] H. S. Kim, S. Hong and J. Lim, "A fast and provably secure higher-order masking of AES S-box", Cryptographic Hardware and Embedded Systems–CHES, **(2011)**, pp. 95-107.
[10] H. Maghrebi, E. Prouff, S. Guilley and J.-L. Danger, "A first- order leak-free masking countermeasure", In Orr Dunkelman, editor, Topics in Cryptology CT-RSA 2012, Lecture Notes in Computer Science, Springer Berlin Heidelberg, vol. 7178, **(2012)**, pp. 156-170.
[11] T. S. Messerges, "Using second-order power analysis to attack dpa resistant soft- ware", In Thomas S Messerges, editor, Cryptographic Hardware and Embedded System- s CHES 2000, Lecture Notes in Computer Science, Springer Berlin Heidelberg, vol. 1965, **(2000)**, pp. 238-251.
[12] S. Mangard, T. Popp and B. M Gammel, "Side-channel leakage of masked cmos gates", Topics in Cryptology–CT-RSA 2005, Springer, **(2005)**, pp. 351-365.
[13] S. Mangard, N. Pramstaller and E. Oswald, "Successfully attacking masked aes hardware implementations", In JosyulaR. Rao and Berk Sunar, editors, Cryptographic Hardware and Embedded Systems CHES 2005, Lecture Notes in Computer Science, Springer Berlin Heidelberg, vol. 3659, **(2005)**, pp. 157-171.
[14] D. Wagner and J. Waddle, "Towards efficient second-order power analysis", Cryptographic Hardware and Embedded Systems-CHES, **(2004)**, pp. 1-15.
[15] Z. Gong, S. Nikova and Y. Law, "Klein: A new family of lightweight block ciphers", In RFID. Security and Privacy, Lecture Notes in Computer Science, Springer Berlin Heidelberg, vol. 7055, **(2012)**, pp. 1-18.
[16] S. Mangard, E. Oswald and T. Popp, "Power analysis attacks: Revealing the secrets of smart cards", Springer, vol. 31, **(2007)**.

