International Journal of Computer Sciences and Engineering Systems
Vol. 10 No. 2 (December, 2016)

A Proposition to Enhance a Countermeasure Scheme Against Power Analysis Attack for AES

Yuan-Man TONG, Hong-Yi LU, Zhi-Ying WANG and Kui DAI

School of Computer Science, National University of Defense TechnologyChangsha, Hunan 410073, China
yuanmantong@yahoo.com.cn

Abstract: *The central thing of an AES implementation secure against power analysis attacks is to protect the only non-linear transformation, SubBytes. Z. Liu proposed a hardware implementation of SubBytes based on random permutation of all the bytes of the State and heterogeneous S-boxes. So the power consumption of this circuit is randomized and power analysis attack is thwarted. This paper points out that this scheme does not really prevent power analysis attacks and two feasible attacks are shown. Then two enhancements are proposed to improve this scheme. One is introducing truly random value but not the value computed on part of the State, the other is implementing completely heterogeneous S-boxes based on finite field isomorphism. The practical experiment results of the attacks on this circuit show the correctness of our security analysis and enhancements.*

Keywords: *side-channel attack, power analysis attack, AES, countermeasure, SubBytes, random permutation, heterogeneous S-boxes*

1. INTRODUCTION

Cryptographic algorithms, including symmetric ciphers and public-key ciphers, are essential building blocks of various secure chips such as smart cards, USB Key, etc. With the stored secret key, cryptography is used to perform encryption, digital signature and authentication. Side-channel attacks in general, and power analysis attacks in particular, can easily break the stored secret key of secure chips [1]. In a power analysis attack, the attacker records the power consumption of a secure chip while it performs cryptographic operations with a fixed secret key. This secret key can subsequently be revealed based on the recorded power traces and the corresponding plaintexts or ciphertexts. Power analysis attack is applicable to almost any cryptographic algorithms.

The AES is the worldwide de-facto standard for symmetric encryption [2]. And power analysis attack can be used to break an unprotected implementation of AES. To resist power analysis attack for AES, many countermeasures have been proposed. Countermeasures are principally divided into two groups: algorithmic countermeasures and hardware countermeasures. Algorithmic countermeasure includes masking scheme [3-6], randomization in time [7]. The group of hardware countermeasure includes the novel logic style with constant power consumption [8-10], reduction of signal-to-noise ratio by introducing additional circuit [11-12].

The most complicated thing to construct a secure implementation of AES is to protect the only non-linear transformation, SubBytes [4-7]. The authors of [13] presented a hardware implementation of SubBytes based on heterogeneous S-boxes and randomly permutation of all the bytes of the State. However, we find that this scheme does not really resist power analysis attacks. Two feasible attacks are deduced and proven to be applicable by practical experiment results.

To improve the security of the scheme in [13], two enhancements are proposed in this paper. One is introducing truly random value but not the value computed on part of the State, the other is implementing completely heterogeneous S-boxes based on finite field isomorphism. The security of the enhanced implementation is greatly improved. The experiment result shows that the attack that breaks the former design does not successfully break the enhanced one.

The rest of this paper is organized as follows. In section 2, we will give a brief introduction of AES and the feasible power analysis attack on AES. In section 3, we introduce the hardware implementation of SubBytes in [13]. In section 4, two feasible attacks on the scheme in [13] are pointed out and the practical experiment results of the attack are shown. In section 5, two enhancements are proposed and the corresponding attack result is shown. Section 6 is the summary of this paper.

2. REVIEW OF AES

The AES algorithm is a symmetric block cipher that operates on 128-bit data blocks [2]. AES uses a cipher key to encrypt

a 128-bit data block. The length of the cipher key can be 128 bits, 192 bits, or 256 bits. The input, output and intermediate cipher result called State are represented as 4×4 arrays of bytes. As most symmetric ciphers, AES encrypts an input data-block by applying the same round function iteratively. The round function alters the State by applying non-linear, linear, and key-dependent functions. In one round, the input state is mapped to the output state by performing the following four different transformations one after another.

(1) The **SubBytes** transformation is a non-linear byte substitution (which is called S-box) that operates independently on each byte of the State. It is defined by a multiplicative inversion in the finite field GF(2^8) followed by an affine transformation.

(2) In the **ShiftRows** transformation, the bytes in the last three rows of the State are cyclically shifted over different numbers of bytes (offsets). The first row is not shifted.

(3) **MixColumns** transforms each column of the State. Each byte in a column is interpreted as the coefficients of a polynomial in an extension field over GF(2^8). This polynomial is multiplied by the constant polynomial $c(x) = \{03\}x^3 + \{01\}x^2 + \{01\}x + \{02\}$, where the coefficients are elements of GF(2^8) in hexadecimal notation. The modular remainder of the resulting product modulo $x^4 + 1$ is the output of MixColumns. In the last round, the MixColumns is not needed.

(4) In the AddRoundKey transformation, a round key is added to the State by a simple bitwise XOR operation (⊕). Each round key is derived from the key schedule and the size of a round key is equal to the State's. Before the first round, the AddRoundKey is performed on the initial key and input block.

Let M, K, and I be the input data block (plaintext), key and State. A byte of I in the i-th row and j-th column is denoted by I_{ij} ($0 \le i \le 3$, $0 \le j \le 3$). So $I_{ij} = M_{ij} \oplus K_{ij}$ at the beginning of the first round. In round 1, the output of S-box of each byte is denoted by $A_{ij} = S(I_{ij}) = S(M_{ij} \oplus K_{ij})$. The output A_{ij} depends on a byte of key, K_{ij}. With a guess of K_{ij}, the value of A_{ij} can be determined too. So A_{ij} can be used to perform a DPA attack to determine the value of K_{ij}. In a mean test based DPA, an attacker divide the sampled power traces into two sets according to the value of A_{ij}, then the mean difference trace of the two sets is obtained [2, 14]. In a correlation power analysis (CPA), an attacker calculates the correlation coefficient between the power traces and the Hamming Weight of A_{ij} [15]. If the key guess is correct, the obvious peak in the mean difference trace or the large correlation coefficient can be found. After 256 guess of a key byte, K_{ij}, the correct value of K_{ij} can be determined.

The intermediate result of other transformations can also be the target which may be attacked. And in the successive rounds, there are similar power analysis attacks.

3. THE CORRESPONDING COUNTERMEASURE OF AES

Many kinds of countermeasures are proposed to protect AES implementing units including software units and hardware units against power analysis attack. The most difficult thing to implement an AES unit securely against power analysis is to protect the only non-linear operation, SubBytes [4-7]. The authors of [13] proposed a hardware implementation of SubBytes to resist power analysis attacks. The following is the brief introduction of this countermeasure.

The key of [13] is to randomize the power consumption caused by A_{ij} which is the output of an S-box. The architecture of the SubBytes unit in [13] is shown in Fig. 1. In the rest of this paper, we call this unit S-PAT. S-PAT has n ($4 \le n \le 16$) heterogeneous S-boxes denoted by S1, S2, …, Sn. In the following, n is 16. Each byte of I (State) is randomly transmitted to an S-box. In each encryption (or decryption), each byte of I is processed by a randomly chosen S-box, but not by a fixed S-box. So the power consumption caused by A_{ij} is randomized and power analysis attack is thwarted. In S-PAT, the Randomized Permutation generates a random permutation of all bytes of I and delivers them to the heterogeneous S-boxes. The Recovering Unit generates the right ordered permutation of all the bytes of I. The random value needed by the Randomized Permutation and Recovering Unit is generated by the Coding Unit which takes the State as the input. In other word, the random value is not from a TRNG (true random number generator). In S-PAT, the Randomized Permutation is a cyclically shift over random numbers of bytes. For example, if the random value generated by the Coding Unit is 4, then all the bytes of I are left shifted for 4 times cyclically, so the first byte is transmitted to the fifth S-box, the second byte to the sixth S-box, and so on. In the Recovering Unit, the outputs of all S-boxes are reordered by right shift cyclically.

S-PAT has five different kinds of S-boxes denoted by A, B, C, D, E. A-type S-box is based on the composite field arithmetic [16-18]. That is to say, the element of GF(2^8) is mapped to GF((2^4)^2) or GF(((2^2)^2)^2) since the inversion in the latter field is much simpler than the operation in GF(2^8). B-type S-box is called PPRM S-box which is the enhancement of a composite field S-box [19]. The delay element is inserted to a B-type S-box in order to balance the delay of different branches. So the glitches are reduced and power consumption also decreases. C-type S-box is a DSE (Decoder Switch Encoder) based S-box with low power consumption [20]. D-type S-box is the most common LUT (look up table) based S-box [21]. E-type S-box is a BDD (Binary Decision Diagram) based S-box [22]. In S-PAT, an array of various S-boxes is shown in Fig. 2. There are 4 A-type S-boxes, 3 B-type, 3 C-type, 3 D-type and 3 E-type S-boxes.

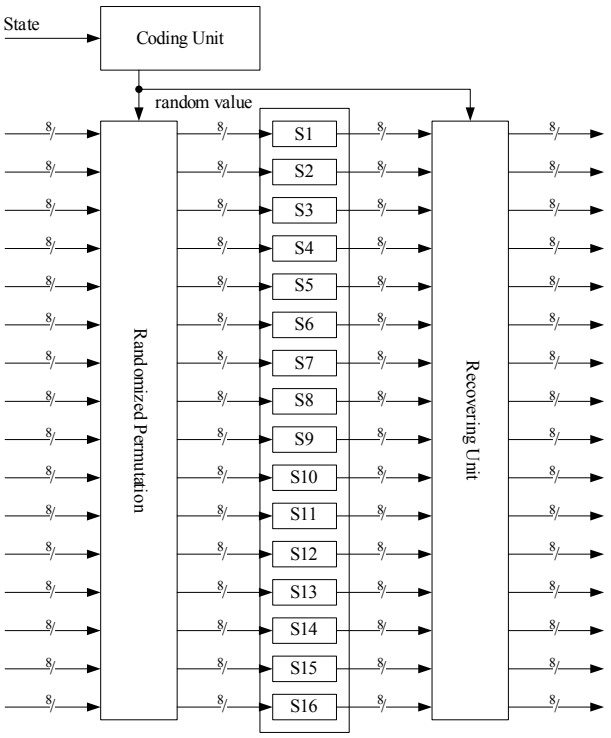

Figure 1: The Architecture of S-PAT

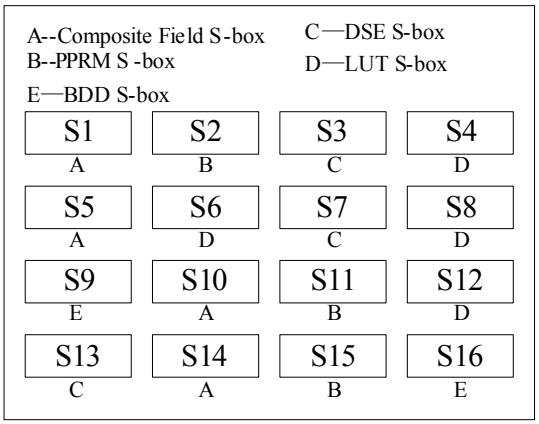

Figure 2: An Array of S-boxes in S-PAT.

Unlike masking scheme, the input and output of S-PAT are not masked. The resistance to power analysis attack of S-PAT is based on the random choosing of different S-boxes for each byte of I.

4. SECURITY ANALYSIS OF THE RELATED COUNTERMEASURE

The following hypothesis shows that when an adversary can perform a standard first order DPA attack [2,14].

Fundamental hypothesis: If an intermediate result z depends on a few bits (in practice less than 32 bits) of key k such that knowing the value of k and the plaintext (or the ciphertext) may allow an adversary to divide the measured power traces into at least two sets according to z.

According to the above hypothesis, S-PAT is vulnerable to DPA attacks. The feasible attacks are explained as follows.

(1) Attack on the Coding Unit

Since the Coding Unit takes the State as input, the random value generated by the Coding Unit is absolutely depends on the round key so that a DPA attack is feasible. Now we describe the related DPA attack on the Coding Unit in the first round.

In S-PAT, the scheme to generate random value is not fixed. A scheme in [13] just computes the Hamming Weight of two bytes of State. That is to say, the random value r is computed as

$$r = HW(I_{pq}) + HW(I_{rs}) = \sum_{i=0}^{7} b_i + c_i, r \in \{0,1,2,...,16\} \quad (1)$$

In Eq. (1), b_i, c_i $(0 \le i \le 7)$ is the i^{th} bit of two bytes, I_{pq} and I_{rs}, and $0 \le p, q, r, s \le 3$. So r depends on two bytes of key, K_{pq} and K_{rs}. Besides, r depends on two bytes of plaintext (which is chosen by the adversary), M_{pq} and M_{rs}. According to the fundamental hypothesis, an adversary can perform a DPA attack on r to break the value of these two bytes of key. He (or she) must guess 2^{16} possible values of K_{pq} and K_{rs} to find the correct guess.

(2) Attack on the S-boxes

As shown in section 2, the output of an S-box is vulnerable to DPA attacks. In S-PAT, the intermediate results are not masked. According to the above hypothesis, the output of each S-box is vulnerable to power analysis attacks. However, the random permutation of the S-boxes thwarts the power analysis attacks. So long as the randomization is eliminated, the attacks can be improved greatly.

Since the random value generated by the Coding Unit is not really randomized, but depends on some bytes of the State. An adversary can make the input to the Coding Unit a fixed value, and then the output of the Coding Unit is also a fixed value so that the expected randomization will not occur. That is to say, in each encryption (or decryption), each byte of I is processed by a certain S-box, not by randomly chosen S-box. For example, let the Coding Unit takes the first two bytes of I as the input, an adversary may choose the plaintext so that the first two bytes of M are fixed but the rest is randomly generated. Let (M_{00}, M_{01}) be a fixed value $\acute{a}$, so the input of the Coding Unit is also a fixed but unknown value β $(= \alpha \oplus (M_{00}, M_{01}))$, and the output is also a fixed value r $(= HW(\beta))$. Now, the adversary can perform DPA attacks on the outputs of the 14 S-boxes except for the two S-boxes processing the first two bytes of I. And the corresponding 14 bytes of key can be determined.

Based on the above theoretic analysis, we perform two correlation power analysis attacks on the third byte ($A_{02} = S(M_{02} \oplus K_{02})$) in the output of S-PAT. In the first attack, we

do not make the first two bytes of M fixed values. In the second one, we eliminate the randomization by fixing the first two bytes of M. Here the correct value of K_{02} is 0x9C (156), and 1000 power traces are measured. The results (the correlation coefficient between the power traces and the Hamming Weight of A_{02} for each guess of K_{02}) of the two attacks are shown in Fig. 3.

Figure 3: The Results of the Two Attacks.

The power traces used in the attacks are simulated by the commercial EDA tools such as PowerMill™. With the 0.25ìm CMOS technology, we implement an S-PAT based SubBytes unit. Then the spice netlist with parasitic parameter (resistance and capacitor) is extracted from the GDS layout of the circuit. So the instantaneous power trace with high accuracy can be simulated.

As shown in the top of Fig. 3, the correct key can not be distinguished since the correlation coefficient while the key guess is 0x9C is not the maximum one in the first attack. However, the second attack is successful to find the correct key (see the bottom of Fig. 3). The result shows that the attack presented above is practical.

5. ENHANCEMENTS OF THE COUNTERMEASURE

In this section, two simple schemes are proposed to enhance S-PAT so that the security against power analysis attack is greatly improved.

(1) Truly Random Value

As shown in section 4, the Coding Unit in S-PAT is vulnerable to power analysis attacks. So we remove the Coding Unit but use the truly random value from a TRNG. The random value then is not dependent on the key and the attack on the random value is not applicable.

(2) Completely Heterogeneous S-boxes

In S-PAT, there are five different types of S-boxes. And the number of each type of S-box exceeds 1. For example, there are 4 A-type S-boxes. So the probability of a certain byte of I is processed by an A-type S-box is (1/4), and the probability for other types of S-boxes is (3/16). So it is clear that each byte of I is not processed by a complete randomly chosen S-box. This may help an adversary to perform power analysis attacks to some degree.

The above analysis implies that the randomization in S-PAT is not achieved to a maximum degree. If 16 completely different S-boxes are used in S-PAT, the maximum randomness is achieved. So the probability of each byte of I is processed by a certain S-box is (1/16). And the security against power analysis attacks is greatly improved. In fact, the security of this enhancement is at least 3 times greater than the original S-PAT.

Then the key problem is how to implement so many different S-boxes. In this paper, we take the measure based on finite field isomorphism. In AES, each byte is an element of the finite field GF(2^8). And here GF(2^8) can be treated as GF(2)[x]/($P(x)$) in fact, where $P(x)$ is an irreducible polynomial of degree 8. This irreducible polynomial is

$$P(x) = x^8 + x^4 + x^3 + x + 1 \tag{2}$$

In fact, the field GF(2)[x]/($P(x)$) is isomorphic with GF(2)[x]/($Q(x)$) while $Q(x)$ is also a irreducible polynomial of degree 8. In the following, the isomorphic mapping between two fields is denoted by σ. Let $u \in$ GF(2)[x]/($P(x)$), then $\sigma(u) \in$ GF(2)[x]/($Q(x)$).

For an element x of GF(2)[x]/($P(x)$), $S(x)$ is computed as

$$S(x) = \begin{cases} L \cdot 0 + c, x = 0 \\ L \cdot x^{-1} + c, x \neq 0 \end{cases} \tag{3}$$

In Eq. (3), L and c are the parameters of the affine transformation in S-box. L is an 8×8 matrix, and c is a byte.

With the finite field isomorphism, the modified S-box is computed as

$$S'(x) = \begin{cases} L \cdot (\sigma^{-1}(\sigma(0))) + c = c, x = 0 \\ L \cdot (\sigma^{-1}((\sigma(x))^{-1})) + c, x \neq 0 \end{cases} \tag{4}$$

In this paper, the inversion $(\sigma(x))^{-1}$ is also computed in the composite field GF((((2^2)^2)^2). Let the isomorphic mapping between GF(2)[x]/($Q(x)$) and GF((((2^2)^2)^2) be φ. So x^{-1} is calculated as

$$x^{-1} = \sigma^{-1}(\varphi^{-1}((\varphi(\sigma(x)))^{-1})) = \psi^{-1}((\psi(x))^{-1}) \tag{5}$$

In Eq. (5), ψ is the composition of φ and σ. And ψ, ψ^{-1} can be pre-computed. The computation of the composite mapping ψ is identical with the computation of a single mapping. So the hardware complexity and computation time of the modified S-box is nearly equal to the A-type S-box's, but the power consumption of the two kinds S-boxes is obviously not identical.

Since there are 30 irreducible polynomials of degree 8, we can choose 11 irreducible polynomials different from $P(x)$

so that we can implement 11 modified S-boxes. Then 16 completely different S-boxes are obtained. For example, the irreducible polynomial $Q(x)$ may be

$$Q(x) = x^8 + x^5 + x^4 + x^3 + 1 \text{ or}$$
$$Q(x) = x^8 + x^7 + x^6 + x^4 + x^2 + 1$$

The presented enhancement does not cause any extra cost of hardware complexity and computation time. Based on the two enhancements, we redesign S-PAT and perform the same attack shown in section 4 on S-PAT to validate the security improvement. In this attack, 5000 power traces are measured. The result of the attack is shown in Fig. 4. And we can see that the correlation coefficient between the power trace and the correct key guess is not the maximum one, so the att

Figure 4: The Result of the Attack on the Modified S-PAT.

6. CONCLUSIONS

In this paper, we analyze the security of S-PAT firstly. Then two feasible attacks are shown. The practical experiment result shows that the mentioned attack really breaks the key. In order to improve the security of S-PAT, two enhancements are proposed. The first one is removing the Coding Unit, and the second one is redesigning several completely different from the S-boxes in S-PAT. With finite field isomorphism, to redesign and implement completely different S-boxes is easy and applicable. The result of the CPA attack on the modified S-PAT shows that the presented enhancements do really work.

ACKNOWLEDGMENTS

The authors would like to thank the Natural Science Foundation of China (NSFC) for funding this research project (No. 60706026).

REFERENCES

[1] P. Kocher, J. Jaffe and B. Jun, "Differential Power Analysis", *Advances in Cryptology*, 1999, LNCS, vol. 1666, pp. 388-397.

[2] National Institute of Standards and Technology (U.S.). Advanced Encryption Standard (AES). FIPS Publication 197, *NIST*, 2001.

[3] T. Messerges, "Securing the AES Finalists Against Power Analysis Attacks", in *Fast Software Encryption*, 2001, LNCS, Vol. 1978, pp. 293-301.

[4] M. Akkar and C. Giraud, "An Implementation of DES and AES, Secure against Some Attacks", in *CHES*, 2001, LNCS, Vol. 2162, pp. 309-318.

[5] Jovan Dj. Golić and Christophe Tymen, "Multiplicative Masking and Power Analysis of AES", in *CHES*, 2003, LNCS, Vol. 2523, pp. 198-212.

[6] Johannes Blõmer, Jorge Guajardo Merchan, and Volker Krummel, "Provably Secure Masking of AES", in *SAC*, 2004, LNCS, Vol. 3357, pp. 69-83.

[7] Christoph Herbst, Elisabeth Oswald, and Stefan Mangard, "An AES Smart Card Implementation Resistant to Power Analysis Attacks", in *ACNS*, 2006, LNCS 3989, pp. 239-252.

[8] Thomas Popp, Stefan Mangard, "Masked Dual-Rail Pre-charge Logic: DPA Resistance Without Routing Constraints", in *CHES*, 2005, LNCS, Vol. 3659, pp. 172-186.

[9] K. Tiri, "Design for Side-channel Attack Resistant Security ICs", Philosophy thesis, University of California, California U.S., 2005.

[10] Yuanman Tong, Zhiying Wang, Kui Dai, Hongyi Lu, "Designing Power Analysis Resistant and High Performance Block Cipher Coprocessor Using WDDL and Wave-Pipeling", in *Inscypt*, 2006, LNCS Vol. 4318, pp. 66-77.

[11] A. Shamir, "Protecting Smart Cards from Passive Power Analysis with Detached Power Supplies", in *CHES*, 2000, LNCS Vol. 1965, pp. 71-77.

[12] Girish B. Ratanpal, Ronald D. Williams, and Travis N. Blalock, "An On-Chip Signal Suppression Countermeasure to Power Analysis Attacks", *IEEE Transactions on Dependable and Secure Computing*, Vol. 1, No. 3, 2004, pp. 179-189.

[13] Z. Liu, X. Zhou, Y. Chen, J. Liu, "A Power Analysis Resistant Circuit of SubBytes", *Chinese Patent*, 2007, No. 200710051298.9. (in Chinese).

[14] T. Messerges, E. A. Dabbish, R. H. Sloan, "Examining Smart-Card Security under the Threat of Power Analysis Attacks", *IEEE Transaction on Computers*, Vol. 51, No. 5, 2002, pp. 541-552.

[15] E. Brier, C. Clavier, F. Olivier, "Correlation Power Analysis with a Leakage Model", in CHES, 2004, LNCS Vol. 3156, pp. 16-29.

[16] Akashi Satoh, Sumio Morioka, Kohji Takano, and Seiji Munetoh, "A Compact Rijndael Hardware Architecture

with S-Box Optimization", in *ASIACRYPT*, 2001, LNCS Vol. 2248, pp. 239-254.

[17] Wolkerstorfer J., Oswald E. Lamberger M., "An ASIC Implementation of the AES SBoxes", in *Topics in Cryptology* CT-RSA, 2002, LNCS Vol. 2271, pp. 67-78.

[18] ZENG Yong-hong, ZOU Xue-cheng, LIU Zheng-lin, LEI Jian-ming, "A low-power Rijndael S-Box Based on Pass Transmission Gate and Composite Field Arithmetic", *Journal of Zhejiang University SCIENCE A*, Vol. 8, No. 10, 2007, pp. 1553-1559.

[19] S. Morioka, A. Satoh, "An Optimized S-Box Circuit Architecture for Low Power AES Design", in *CHES*, 2002, LNCS Vol. 2523, pp. 172-186.

[20] Bertoni G., Macchetti M. Negri L. Fragneto P., "Power-efficient ASIC Synthesis of Cryptographic Sboxes", in *14th ACM Great Lakes Symposium on VLSI*, 2004, ACM Press pp. 277-281.

[21] Bryant R. E., "Graph-Based Algorithms for Boolean Function Manipulation", *IEEE Trans. Computers*, Vol. C-35, No. 8, 1986, pp. 677-691.

[22] S. Morioka and A. Satoh, "A 10-Gbps Full-AES Crypto Design With a Twisted BDD S-Box Architecture", *IEEE Trans. VLSI System*, Vol. 12, No. 7, 2004.