An Optimization for Differential Power Analysis Based on Time Series Verification

Yaxin Zhao[1], Yerong Tao[1,*], Song Zhang[1] and Yinglong Liu[1]
[1]Luoyang Electronic Equipment Test Center of China, Luoyang, China
[*]Corresponding author

***Abstract*—As an effective method of side-channel analysis, DPA (short for differential power analysis) uses statistical analysis to obtain the key by analyzing the correlation between power consumption data and encryption key. If the power consumption data collected from the target contains a large amount of interference noise, it will affect the result of DPA. Even if preprocessing methods are used to process the noise, the result of DPA still has error. This paper proposes a differential power analysis method based on time series verification for the AES algorithm in smart card devices, which extracts intermediate values from the four steps of the single round operation in the AES algorithm, and uses the power model for differential power analysis to obtain the corresponding key, and then respectively calculates the correlation coefficients of the other three steps under each key in the order of time series, and finally obtains the optimal solution key by comprehensive judgment. Experiments show that the result of this method is significantly better than traditional DPA.**

Keywords—DPA; time series; AES algorithm; round transformation

I. INTRODUCTION

The smart cards are extensively used in finance, medical, e-commerce and other related industries, the reliability and security of which depend on the security of cryptographic chips in them. Cryptographic chips usually use DES, AES or RSA algorithms to protect smart cards. However, the circuit of the cryptographic chip usually leaks power consumption, electromagnetic radiation, running time and other information at runtime. These information can be used to analyze the sensitive data or key information of the encryption algorithm in the chip. This method is called side-channel analysis[1]. Therefore, whether the cryptographic chip in smart card is safe or not is not only related to the design of the cryptographic algorithm in the chip, but also needs to detect whether the generated power consumption leaks information of the key. DPA (short for differential power analysis) is an effective power analysis method, which analyzes the correlation between the encryption key and power consumption data, uses statistical analysis method to find useful information related to the key, and then obtains the encryption key.

When collecting power trace on a smart card, there are various types of noise, such as electronic noise and conversion noise, and it is unrealistic to completely remove the noise. During the execution, DPA needs to calculate power consumption of the encryption device when encrypting different plaintext data packets, which is mixed with a lot of interference noise. Although some noise can be removed by preprocessing methods[2], the remaining noise still affects the DPA's solution to the correlation of the encryption key and the power consumption, which results in acquiring an inaccurate encryption key.

This paper takes the AES algorithm in smart card chip as the target, and proposes a method of differential power analysis based on time series verification. In order to solve the problem of low accuracy of traditional differential power analysis, the intermediate values are extracted from four steps of single round transformation in AES algorithm and the corresponding key is obtained by DPA. The accuracy of each key is verified according to the sequence of time series, and the optimal solution key is finally determined.

II. ADVANCED ENCRYPTION STANDARD

As a new symmetric data encryption standard, AES (short for Advanced Encryption Standard) is issued by NIST in 2001. AES is designed to resist attacks based on mathematical characteristics, so it has good characteristics against linear attack[3].

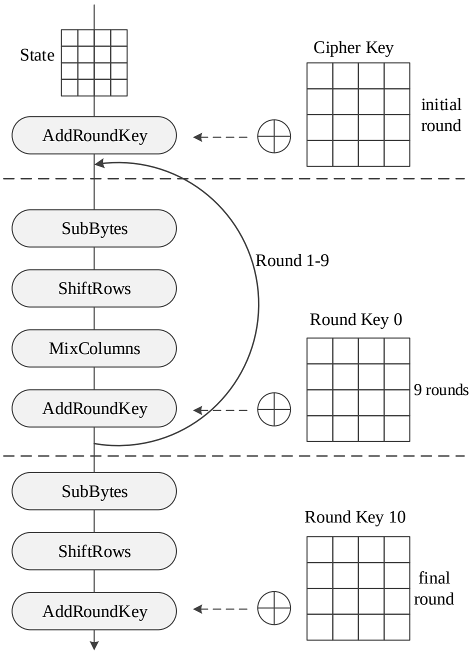


FIGURE I. FLOW CHART OF AES

AES algorithm can encrypt a packet with a length of 128 bits using an optional length of 128, 192 or 256 bits. This paper mainly studies 128-bit AES algorithm, namely AES-128. The data and key are respectively represented by a matrix of 4×4 byte, and the matrix is called state. All changes in AES are based on changes of state. The calculation contains 10 iterations, and each round of iteration uses one round key. Each round transformation includes the following four steps: firstly, AddRoundKey operation; secondly, SubBytes operation; thirdly, ShiftRows operation; fourthly, Mixcolumns operation. The 10th round is different from previous round transformation. The first 9 rounds perform all the operations. But the 10th round only performs the first three operations without Mixcolumns. AES-128 algorithm description is shown in Figure I. .

III. Differential Power Analysis

DPA is an analysis method that uses the data dependency of the power consumption of smart card chips to derive the key in a statistical manner. This method requires a great quantity of power traces for analyzing the power consumption of the chip in fixed moments, and then uses the power consumption as a function of the processed data[4]. The process of DPA is as follows:

Step 1: Select an intermediate value of the encryption in chip. The value must be a correlation function $f(d, k)$ of the plaintext and the key, in which the parameter d is a known non-constant value and the parameter k is a small fraction of the key.

Step 2: Measure the power trace. After selecting the intermediate value of the encryption in chip to be measured, it is necessary for measuring power consumption when the encryption device encrypts a large number of different plaintext groups, and the number of groups is recorded as D. When the chip performs encryption, it is necessary to record the corresponding plaintext value d, and these values are recorded as a vector $\boldsymbol{d}=(d_1, . . . , d_D)'$, in which the parameter d_i represents plaintext data corresponding to the i-th encryption operation. Every group of plaintext data input by the chip will record an power trace. D groups of plaintext data are collected at T time, and the power traces are obtained as a vector $\boldsymbol{t_i}'=(t_{i,1},...,t_{i,T})$, in which the parameter i indicates the i-th group of plaintext data, and finally a $D\times K$ matrix $\boldsymbol{T}$ of is obtained.

Step 3: Calculate the hypothetical intermediate value. After measuring power traces, the corresponding hypothetical intermediate value need be calculated for each possible k value. These possible key hypotheses are denoted as $\boldsymbol{k} = (k_1, ..., k_K)$, in which the parameter K represents possible number of k. For the vector $\boldsymbol{k}$ of key hypothesis and the vector $\boldsymbol{d}$ of plaintext data, a $D\times K$ matrix $\boldsymbol{V}$ of can be obtained.

Step 4: Convert the intermediate value to the hypothetical power consumption. After the matrix $\boldsymbol{V}$ is calculated, it is converted into a matrix $\boldsymbol{H}$, elements of which represent hypothetical power consumption. Assuming that power consumption value is only an estimate of actual power consumption value, power model should be built by the simulation technique to convert the intermediate value. There are some common power models, such as the Hamming-distance model[5], the Hamming-weight model[6] and so on.

Step 5: Perform comparison of real power traces and hypothetical power consumption values. After converting the matrix $\boldsymbol{V}$ into the matrix $\boldsymbol{H}$, comparing each column h_i in $\boldsymbol{H}$ with each column t_j in $\boldsymbol{T}$, a $K\times T$ matrix $\boldsymbol{R}$ can be obtained, the element $r_{i,j}$ of which represents the result of comparing column h_i with column t_j. The comparison can be achieved by linear correlation analysis and other methods. When the subkey is guessed correctly and the time points are consistent, there is a certain relationship between hypothetical and real power consumption. The relationship can be expressed by numerical values through the statistical analysis method. So in the matrix $\boldsymbol{R}$, the larger the value of $r_{i,j}$, the higher the matching degree of the column h_i and the column t_j. By finding the maximum value in the matrix $\boldsymbol{R}$, the index and the corresponding moment of the correct subkey can be guessed.

IV. Time Series Verification

A. Choice of Intermediate Value

In order to verify the guessed keys based on time series, this paper chooses to extract the intermediate values in the four steps of the first round in the AES algorithm.

1) Output of AddRoundKey

The input are the round key and the state data, and AddRoundKey executes bitwise XOR on them. In the first round transformation, the round key used is the original key of the AES algorithm itself, and the state data is the initial plaintext. If a certain byte of the plaintext is bitwise XORed with the corresponding key, the output of AddRoundKey can be calculated.

2) Output of SubBytes

SubBytes replaces each byte of the state data, and a 8-bit input produces the corresponding 8-bit output through S-box. In the first round transformation, a certain byte of the plaintext data is XORed with the corresponding key, the result of which is then inputed to the S-box, and output of SubBytes can be calculated.

3) Output of ShiftRows

ShiftRows shifts each row to left by a set number. The top row is not shifted, and the next row is shifted by one and so on. Since top row is not shifted, if no operation is performed on the top row in the program implementation of the encryption algorithm, no power consumption is generated. Therefore, after SubBytes outputs state matrix in the first round transformation, a byte is selected as the intermediate value from results generated after the shift transformation of the second row, the third row or the fourth row.

4) Output of Mixcolumns

Mixcolumns mixes the elements in every column of the state matrix, multiplies every column with one designated matrix, and outputs a new column. In the first round transformation, the output of Mixcolumns can be calculated after ShiftRows.

B. Model Building

Measure power consumption when encrypting D groups of data, and the plaintext data value is recorded as $\boldsymbol{d}=(d_1,\dots,d_D)'$, in which the parameter d_i represents plaintext value corresponding to i-th encryption operation. The values of key hypotheses are recorded as $\boldsymbol{k}=(k_1,\dots,k_K)$, in which the parameter K represents possible number of k.

AddRoundKey operation is defined as *ARK*, SubBytes operation is *S*, ShiftRows operation is *SR*, and Mixcolumns is *MC*. For the plaintext byte x and the key k in the first round transformation, the output of AddRoundKey is denoted by:

$$f_1(x,k)=ARK(x)=x\oplus k \tag{1}$$

The outputs of SubBytes, ShiftRows and Mixcolumns are respectively denoted by:

$$f2(x,k)=S(ARK(x))=S(x\oplus k) \tag{2}$$

$$f3(x,k)=SR(S(x\oplus k)) \tag{3}$$

$$f4(x,k)=MC(SR(S(x\oplus k))) \tag{4}$$

After calculating the intermediate values of the four steps, four hypothetical intermediate value matrices of size $D\times K$ can be obtained, which are denoted by $\boldsymbol{V}^m$, where m represents the m-th step, and the range of m is 1,…,4. The elements in the matrix are denoted by $v_{i,j}^m$, where $i=1,...,D$, $j=1,\dots K$, $m=1,..,4$. When the variable m takes values of 1, 2, 3 and 4, it respectively indicates AddRoundKey, SubBytes, ShiftRows and Mixcolumns. The calculation steps for $v_{i,j}^m$ are as follows:

$$v_{i,j}^m= f_m(d_i ,k_j) \tag{5}$$

Calculate the hypothetical power consumption value matrix for intermediate values output by the four steps according to the given power model, and the matrix is denoted by $\boldsymbol{H}^m$, where $m=1,...,4$. Define the power model conversion operation as *PM*, and the calculation steps of $\boldsymbol{H}^m$ are as follows:

$$\boldsymbol{H}^m=PM(\boldsymbol{V}^m) \tag{6}$$

The power trace corresponding to the data group d_i is denoted by $t_i'=(t_{i,1},\dots,t_{i,T})$, in which the parameter T represents length of the power trace, and the power trace is represented by the $D\times T$ matrix $\boldsymbol{T}$. The trace from time i to j is denoted by the matrix $\boldsymbol{T}_{i\text{-}j}$, where $1\leq i<j\leq T$. So the measured power trace matrix $\boldsymbol{T}$ of size $D\times T$ can again be represented as $\boldsymbol{T}_{1\text{-}T}$.

The calculation of correlation matrix for real power trace and hypothetical power consumption is defined as *ANA*. For the power trace $\boldsymbol{T}_{i\text{-}j}$ from time i to j, the correlation matrices of AddRoundKey, SubBytes, ShiftRows, and Mixcolumns are as follows:

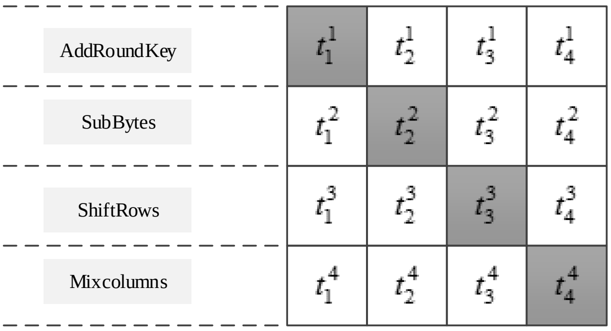
$$\boldsymbol{R}_{i-j}^m=ANA(\boldsymbol{H}^m, \boldsymbol{T}_{i\text{-}j}),\ m=1,\dots,4 \tag{7}$$

Define the k-th column of the matrix $\boldsymbol{R}_{i-j}^m$ as $\boldsymbol{R}_{i-j}^m[k]$, that is, the correlation vector of the power trace from time i to j and hypothetical power consumption when the key is k, where$1\leq i<j\leq T$, $k=1,\dots,K$. The function to calculate the maximum correlation coefficient in the correlation matrix or vector is defined as *MaxRelative*. *MaxKey* and *MaxTime* are defined to calculate the key index and the corresponding time.

C. Solution Design

After completing the differential power analysis for the outputs of four steps in the first round, the corresponding key and time can be respectively obtained. If the measured power trace is precise enough, the guessed key corresponding to the four steps should match. However, when measuring the smart card chip, there usually exists various types of noise, and the guessed key corresponding to each step may have different results. Therefore, this paper designs a differential power analysis method based on time series verification.

For the key k and the corresponding power trace time t guessed by one step in the first round transformation, the value of t is in the range of (1, T). If the key guessed by the current step is correct, the value range of the power trace time in the previous step should be (1, t), and the value range of the power trace time in the next step should be (t, T). The previous steps and subsequent steps can be respectively applied by DPA to solve their maximum correlation coefficients in the power trace time range (1, t) and (t, T). Therefore, for each step in the first round transformation, the maximum correlation coefficients of other steps can be obtained in the corresponding power trace time range by the sequence of time series.



AddRoundKey	t_1^1	t_2^1	t_3^1	t_4^1
SubBytes	t_1^2	t_2^2	t_3^2	t_4^2
ShiftRows	t_1^3	t_2^3	t_3^3	t_4^3
Mixcolumns	t_1^4	t_2^4	t_3^4	t_4^4

FIGURE II. TIME SERIES CHART OF EACH STEP

The corresponding power trace times obtained by DPA in the four steps of the first round transformation are recorded as t_m^m as shown in Figure II. , and the guessed keys are recorded as k^m, and the maximum correlation coefficient is denoted as r_m^m, where $m=1,\dots,4$. The calculation results of t_m^m, k^m and r_m^m are as follows:

$$t_m^m = MaxTime(\boldsymbol{R}_{1-T}^m) \tag{8}$$

$$k^m = MaxKey(\boldsymbol{R}_{1-T}^m) \tag{9}$$

$$r_m^m = MaxRelative(\boldsymbol{R}_{1-T}^m) \tag{10}$$

The guessed key of DPA for AddRoundKey is k^1, and the corresponding time is t_1^1. If the current guessed key is correct, the remaining three steps should be in the power trace time range (t_1^1,*T*). It is respectively recorded as t_2^1, t_3^1 and t_4^1, and only needs to be calculated in the condition where the key is k^1 and the power trace time range is (t_1^1,*T*), and the corresponding correlation coefficients are r_2^1, r_3^1 and r_4^1.

$$t_2^1= MaxTime\ (\boldsymbol{R}^2_{t_1^1-T}[k^1]),\quad r_2^1= MaxRelative\ (\boldsymbol{R}^2_{t_1^1-T}[k^1])$$
$$t_3^1= MaxTime\ (\boldsymbol{R}^3_{t_2^1-T}[k^1]),\quad r_3^1= MaxRelative\ (\boldsymbol{R}^3_{t_2^1-T}[k^1])$$
$$t_4^1= MaxTime\ (\boldsymbol{R}^4_{t_3^1-T}[k^1]),\quad r_4^1= MaxRelative\ (\boldsymbol{R}^4_{t_3^1-T}[k^1])$$

And so forth, the values of the variables in SubBytes are solved as follows:

$$t_1^2= MaxTime\ (\boldsymbol{R}^1_{1-t_2^2}[k^2]),\quad r_1^2= MaxRelative\ (\boldsymbol{R}^1_{1-t_2^2}[k^2])$$
$$t_3^2= MaxTime\ (\boldsymbol{R}^3_{t_2^2-T}[k^2]),\quad r_3^2= MaxRelative\ (\boldsymbol{R}^3_{t_2^2-T}[k^2])$$
$$t_4^2= MaxTime\ (\boldsymbol{R}^4_{t_3^2-T}[k^2]),\quad r_4^2= MaxRelative\ (\boldsymbol{R}^4_{t_3^2-T}[k^2])$$

The values of the variables in ShiftRows are solved as follows:

$$t_1^3= MaxTime\ (\boldsymbol{R}^1_{1-t_3^3}[k^3]),\quad r_1^3= MaxRelative\ (\boldsymbol{R}^1_{1-t_3^3}[k^3])$$
$$t_2^3= MaxTime\ (\boldsymbol{R}^2_{t_1^3-t_3^3}[k^3]),\quad r_2^3= MaxRelative\ (\boldsymbol{R}^2_{t_1^3-t_3^3}[k^3])$$
$$t_4^3= MaxTime\ (\boldsymbol{R}^4_{t_3^3-T}[k^3]),\quad r_4^3= MaxRelative\ (\boldsymbol{R}^4_{t_3^3-T}[k^3])$$

The values of the variables in Mixcolumns are solved as follows:

$$t_1^4= MaxTime\ (\boldsymbol{R}^1_{1-t_4^4}[k^4]),\quad r_1^4= MaxRelative\ (\boldsymbol{R}^1_{1-t_4^4}[k^4])$$
$$t_2^4= MaxTime\ (\boldsymbol{R}^2_{t_1^4-t_4^4}[k^4]),\quad r_2^4= MaxRelative\ (\boldsymbol{R}^2_{t_1^4-t_4^4}[k^4])$$
$$t_3^4= MaxTime\ (\boldsymbol{R}^3_{t_2^4-t_4^4}[k^4]),\quad r_3^4= MaxRelative\ (\boldsymbol{R}^3_{t_2^4-t_4^4}[k^4])$$

Based on the above steps, for the guessed key k^m(*m=1,…,4*), the corresponding correlation coefficients are solved in time sequence of the four steps in the first round transformation of the AES algorithm. Taking the guessed key k^i as an example, the four corresponding correlation coefficients are r_1^i, r_2^i, r_3^i, and r_4^i. If current key k^i is guessed correctly, its comprehensive weights should be higher than results of other keys. Therefore this paper measure the accuracy of the key by the sum of the four correlation coefficients corresponding to the guessed key k^m, which is denoted by Z^m, and its calculation result is as follows:

$$Z^m= \sum_{j=1}^{4} r_j^m,\ m=1,\ldots,4 \tag{11}$$

Through comprehensive judgment, when the variable *i* satisfies $Z^i = Max(Z^m)$, the correct key result is k^i.

V. Experimental results and analysis

A. Experiment Steps

The experiment uses the simulation environment to simulate the AES encryption process. After placement and routing, we can use the Hspice tool to collect power consumption data from the network netlist file in the circuit. To verify the accuracy of the method in this paper, we add simulated random noise to the circuit. Besides, we set a specified key for the AES program in the circuit, which is reserved for verification.

There are 1000 groups of plaintext input for encryption, and we collect the power consumption data during the encryption process to obtain 1000 groups of power trace data, and each group of power trace includes 800 collection points of power consumption. The AES algorithm uses a ten-round encryption process, which can easily preprocess the power trace and extract the interval of the first round transform from it, thus we can obtain 1000 groups of power traces in the first round transform. Firstly, we use the traditional DPA method to analyze the output of AddRoundKey, and guess the key. Then we use the differential power analysis method based on time series verification to analyze the key result. Finally, we compare the two guessed keys with the specified key set in the circuit.

For the currently specified key, we repeat the above process 100 times, and calculate the accuracy of the traditional DPA method and the DPA method based on time series verification.

Then we modify the AES encryption key in the circuit, set a total of 10 different keys, and repeat the above process to obtain the accuracy of analyzing 10 different keys. The result is as shown in TABLE I. .

TABLE I. COMPARISON RESULTS OF DPA

Subkey byte	Accuracy rate of traditional DPA	Accuracy rate of traditional DPA based on time series verification
0x01	95%	98%
0x15	94%	99%
0x22	95%	99%
0x32	93%	98%
0x49	94%	99%
0x50	92%	97%
0x67	93%	98%
0x7E	92%	98%
0x88	94%	99%
0x9F	96%	97%

B. Experimental Analysis

By adding simulated noise to the simulation circuit, 10 different keys are set for the AES encryption program in the circuit. The results of the traditional DPA method and the DPA method based on time series verification are respectively calculated for each set key. The results of TABLE I. show that the differential power analysis optimization method based on time verification has higher accuracy than the traditional differential power analysis method, which verifies the effectiveness of the proposed method in this paper.

VI. Conclusion

Based on the analysis of errors in traditional differential power analysis, this paper proposes an optimization method for differential power analysis based on time series verification. This method extracts intermediate values for the four steps of the first round transformation in AES algorithm, uses power model to analyze and acquire the corresponding key, then calculates the correlation coefficients of the other three steps under each key according to the sequence of time series, and finally obtains the optimal solution key by comprehensive judgment. The simulation results show that the proposed method has better effect than the traditional differential power analysis method.

REFERENCES

[1] Stefan Mangard, Elisabeth Oswald, Thomas Popp. Power Analysis Attacks[M]. Springer Science+Business Media,2007.

[2] Paul Kocher,Joshua Jaffe,and Benjamin Jun.Differential Power Analysis[C].Lecture Notes in Computer Science ,1999,1666:388-397.

[3] National Institute of Standards and Technology(NIST).FIPS-197: Advanced Encryption Standard, November 2001. Available online at http://www.itl.nist.gov/fipspubs/.

[4] Stefan Mangard, Elisabeth Oswald, Thomas Popp. Power Analysis Attacks[M]. Springer Science+Business Media,2007.

[5] Moradi A, Barenghi A, Kasper T, et al. On the vulnerability of FPGA bitstream encryption against power analysis attacks: extracting keys from Xilinx Virtex-II FPGAs[C]//Proc of the 18th ACM Conference on Computer and Communication Security. New York: ACM Press, 2011: 111-124.

[6] Katashita T, Hori Y, Sakane H, et al. Side-channel attack standard evaluation board SASEBO-W for smartcard testing[J].Power,201,3:400.