Computers and Mathematics with Applications 65 (2013) 1291–1299

Contents lists available at SciVerse ScienceDirect

Computers and Mathematics with Applications

journal homepage: www.elsevier.com/locate/camwa

Differential power analysis of stream ciphers with LFSRs[☆]

Bo Qu[a], Dawu Gu[a,*], Zheng Guo[b], Junrong Liu[c]

[a] Department of Computer Science and Engineering, Shanghai Jiao Tong University, 200240 Shanghai, China
[b] School of Microelectronics, Shanghai Jiao Tong University, 200240 Shanghai, China
[c] School of Information Security Engineering, Shanghai Jiao Tong University. 200240 Shanghai, China

ARTICLE INFO

Keywords:
Side-channel attack
Differential power analysis
Correlation coefficient
Stream cipher
Linear feedback shift register
Crypto-1

ABSTRACT

Side-channel attacks on block ciphers and public key algorithms have been discussed extensively, but only a few systematic studies on the applicability of side-channel attacks to stream ciphers could be found. The objective of the present study is to develop general differential power analysis techniques which can be employed to attack the stream ciphers with linear feedback shift registers. To illustrate the new approach, a common structure of a stream cipher with the basic components is given. Then the approach is employed to analyze the given structure. The results show that the linear feedback shift registers may leak the information of the secret key. The approach is also applied to Crypto-1 and the experimental results show that it is very effective. 28-bit information of the 48-bit secret key can be obtained just by analyzing some power traces. Furthermore, the present work may be helpful in analyzing a variety of stream ciphers with LFSRs.

© 2012 Elsevier Ltd. All rights reserved.

1. Introduction

Side-channel analysis (SCA) [1,2], regarded as a new research area, has gained more and more attention since the mid-nineties. SCA is any kind attack based on the information leakages from the software or hardware implementation of the cryptosystem, rather than brute force or theoretical weaknesses in the algorithms. For instance, timing information [2], power consumption [1,3], electromagnetic radiation [4,5] or even acoustic information can provide an extra source of information which can be used to break the system. Among kinds of SCA, power analysis techniques [6–11] are well-known and powerful, especially for differential power analysis (DPA) [1,3,12] which involves statistically analyzing power consumption measurements from a cryptosystem.

DPA can be devastating effective, and it is a thoroughly studied method to analyze the implementations of block ciphers, like DES [13] and AES [14], and public key algorithms, like ECC [15]. Even so, many software or hardware implementations of the new systems have been made more consideration to the power consumption, rather than the defense to SCA [16–19].

But in the field of stream ciphers, this topic is rather rare, and it is even true for any kind of SCA. That is because, for a long time, the stream ciphers were thought to be hard to attack with SCA techniques. This idea was derived from the fact that, in general, stream ciphers produced a non-repeatable and unpredictable stream of key bits that were subsequently XOR-ed to the data being encrypted. Only a few research papers about power analysis and stream ciphers have been

[☆] This work is supported by the National Natural Science Foundation of China (No. 61073150).

[*] Correspondence to: Department of Computer Science and Engineering, Shanghai Jiao Tong University, 800 Dongchuan Road, 200240 Shanghai, China. Tel.: +86 21 34205416; fax: +86 21 3420516.
E-mail address: dwgu@sjtu.edu.cn (D. Gu).

0898-1221/$ – see front matter © 2012 Elsevier Ltd. All rights reserved.
doi:10.1016/j.camwa.2012.02.024

published so far, such as template attacks [20,21] and fault attacks [22] against stream ciphers. Some references providing a theoretical analysis of stream ciphers are available, such as the resynchronization mechanism of stream ciphers [23] and implementation properties of stream ciphers [24,25].

In this paper, a general DPA method based on the correlation coefficient which can be used to analyze the linear feedback shift registers (LFSRs) is firstly discussed. It mainly takes advantage of the features of LFSR that the power consumption of the shift operations to LFSR is considerable and data-dependent. For example, when the LFSR is shifted a bit to the left, one bit of the LFSR changes from 1 to 0 while another bit remains 0, and the power consumption of the two bits changes are quite different. The results of our analysis will illustrate the vulnerabilities of LFSRs in the analysis of DPA. Due to the fact that LFSR is a very common part in stream ciphers, our approach can be used to attack many stream ciphers. Furthermore, this technique is also applied to attack a stream cipher named Crypto-1 which is widely used in contactless smart cards. Though some studies about the cryptanalysis of Crypto-1 [26–29] could be found, no DPA attacks on it are available. By analyzing the power consumption of the hardware emulation of Crypto-1, the effectiveness of our DPA method for the analysis of the stream ciphers with LFSRs could be shown.

The paper is structured as follows. Section 2 introduces the notations used throughout this paper and describes DPA attacks and its general steps briefly. A simple but common structure of a stream cipher is given and our approach is applied to the structure step by step to demonstrate how to use it to analyze the stream ciphers with LFSRs in Section 3. Section 4 chooses Crypto-1 as an example to be attacked to show the effectiveness of our approach. Finally, conclusions and findings are drawn in Section 5.

2. Preliminaries

The following notations are used throughout the paper.

- $\oplus$ denotes bitwise exclusive OR (XOR).
- $|x|$ denotes the absolute value of a real number x.
- 0x denotes the hexadecimal notation.
- $\sim$ denotes all the values from an integer to a larger integer.

2.1. *Differential power analysis*

Differential power analysis was introduced by Kocher et al. in [1,3]. Because of the requirement of almost no detailed knowledge about the attacked cryptosystem, DPA is the most popular approach in the power analysis. Furthermore DPA attacks are effective even if the recorded power traces are extremely noisy. To reveal secret keys of the attacked cryptosystem, DPA exploit the data dependency of the power consumption by using a large number of power traces to analyze the power consumption at a fixed moment of time as a function of the processed different data blocks.

A general attack strategy used by all DPA attacks consists of several steps as follows.

Choose an intermediate value. The first step of DPA is to choose an intermediate value of the cryptographic algorithm executed by the cryptosystem. Let d be a known non-constant data value and k be a small part of the secret key, then the intermediate value v needs to be a function of v and k, $v = f(d, k)$. Such an intermediate value v can be used to reveal k. In most attacks on block ciphers, d is either the plaintext or the ciphertext, but on stream ciphers, d is usually the initial value.

Measure the power consumption. This step is to measure the power consumption of the cryptosystem, when it processes different data blocks. The adversary can make the cryptosystem process D different data blocks and record a power trace t for each data block. For each power trace, the adversary needs to know the corresponding data value d which is involved in the previous step. We write D known data values as vector $\mathbf{d} = (d_1, \dots, d_D)'$, where d_i denotes the data value while the cryptosystem processes the ith data block. We refer to the power trace of processing the ith data block as $\mathbf{t}_i = (t_{i,1}, \dots, t_{i,T})$, where T denotes the length of one trace. So, for all the power traces of D different data blocks, we can write a matrix $\mathbf{T}$ of size $D \times T$.

Calculate hypothetical intermediate values. In this step, a hypothetical intermediate value for every possible choice of k that is involved in the first step is calculated. We write all the possible K choices of k as vector $\mathbf{k} = (k_1, \dots, k_K)$, and we have D known data values $\mathbf{d} = (d_1, \dots, d_D)'$, so we can calculate as (1). This calculation results in a matrix $\mathbf{V}$ of size $D \times K$. Fig. 1 demonstrates the process of calculating $\mathbf{V}$.

$$v_{i,j} = f(d_i, k_j), \quad i = 1, \dots, D, \ j = 1, \dots, K. \tag{1}$$

It is a remarkable fact that all the intermediate values that have been calculated based on the key hypothesis k_j are in the jth column of $\mathbf{V}$. Each column of $\mathbf{V}$ contains the intermediate values that have been calculated in the cryptosystem while D different data blocks are processed. So the goal of DPA is to find out which column of $\mathbf{V}$ has been processed in the attacked cryptosystem, the corresponding k is just the right small part of the secret key.

Map intermediate values to power consumption values. The fourth step of DPA is to map the hypothetical intermediate values $\mathbf{V}$ to a matrix $\mathbf{H}$ which consists of hypothetical power consumption values. To make such a mapping, the adversary needs to build a power model, such as Hamming-distance, Hamming-weight model or some other models. By using a power

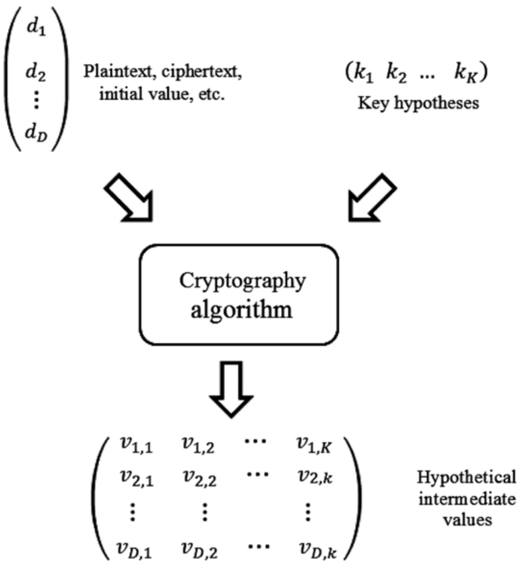
Plaintext, ciphertext, initial value, etc.
Key hypotheses
Cryptography algorithm
Hypothetical intermediate values

Fig. 1. Calculate the hypothetical intermediate values.

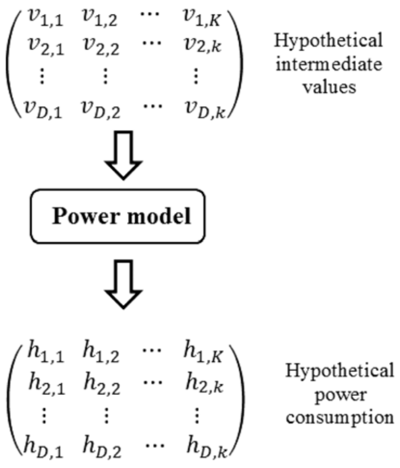
Hypothetical intermediate values
Power model
Hypothetical power consumption

Fig. 2. Map intermediate values to power consumption values.

model, the power consumption of the cryptosystem for each hypothetical intermediate value $v_{i,j}$ is simulated in order to obtain a hypothetical power consumption value $h_{i,j}$. The better the simulation of the adversary matches the actual power consumption characteristics of the cryptosystem, the more effective is the DPA attack. That is to say, the power model plays a very important role in the whole analysis. It is also remarkable that the jth column of $\mathbf{H}$ indicates the key hypothesis k_j. Fig. 2 shows the mapping process.

Compare the hypothetical power consumption values with the power traces. The final step of DPA can be performed after having mapped $\mathbf{V}$ to $\mathbf{H}$. Each column $\mathbf{h}_i$ of the matrix $\mathbf{H}$ should be compared with each column $\mathbf{t}_j$ of the matrix $\mathbf{T}$. That is to say, the adversary compares the hypothetical power consumption values of each key hypothesis with the power traces at every position. The result of this comparison is a matrix $\mathbf{R}$ of size $K \times T$, and the result of the comparison between the columns $\mathbf{h}_i$ and $\mathbf{t}_j$ is the element $r_{i,j}$ in $\mathbf{R}$. The values $r_{i,j}$ with the highest absolute value in $\mathbf{R}$ reveals the correct guessing key. The most common comparison is done based on the correlation coefficient, see (2). In (2), the values $\overline{h}_i$ and $\overline{t}_j$ denote the mean values of the columns $\mathbf{h}_i$ and $\mathbf{t}_j$. The comparison can be illustrated as shown in Fig. 3.

$$r_{i,j} = \frac{\sum_{d=1}^{D} (h_{d,i} - \overline{h}_i) \cdot (t_{d,j} - \overline{t}_j)}{\sqrt{\sum_{d=1}^{D} (h_{d,i} - \overline{h}_i)^2 \cdot \sum_{d=1}^{D} (t_{d,j} - \overline{t}_j)^2}}. \tag{2}$$

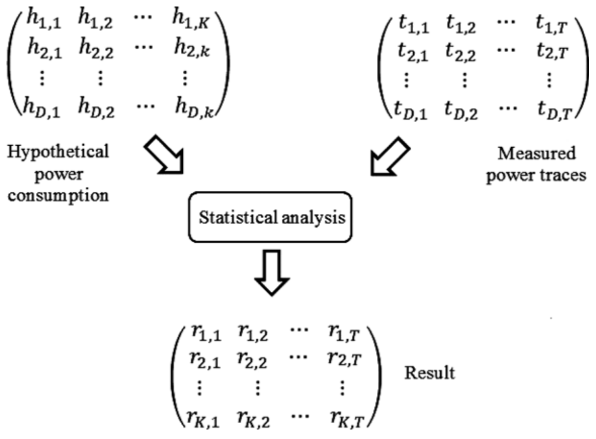

Fig. 3. Compare the hypothetical power consumption values with the power traces.

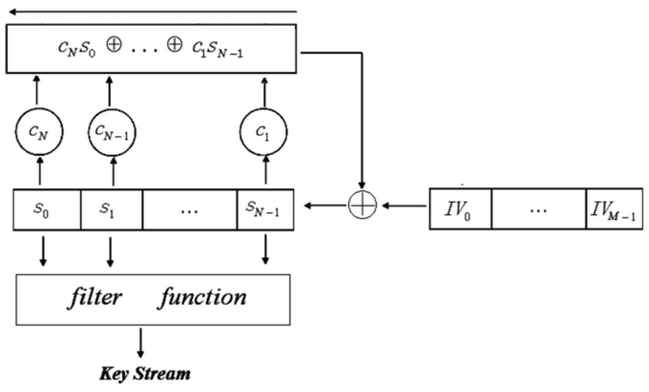

Fig. 4. A simplified common structure of the stream ciphers with LFSR.

3. Our DPA attack on LFSR structure

3.1. A brief description of stream ciphers with LFSRs

Because LFSRs can be easily implemented in hardware and can be readily analyzed mathematically, binary stream ciphers are often constructed by using them. But using only LFSR cannot provide good security, various schemes have been proposed to increase the security of LFSRs. One popular approach is to use filter functions. According to the fact above, a common structure of stream ciphers based on the LFSR and filter function can be described as Fig. 4.

In order to develop general DPA techniques to analyze the stream ciphers with LFSRs, attentions are firstly focused on using DPA techniques to analyze the LFSR structure. So a possible but simple structure of binary stream ciphers with the most basic parts of a stream cipher is shown in Fig. 4. It can be seen that, in this structure, there is only one LFSR, one filter function, etc. But those are sufficient to form a stream cipher and enough to show the weakness of the stream ciphers with LFSR. In Fig. 4, $\mathbf{s} = (s_0, \dots, s_{N-1})$ denotes the internal state of N bits, and $\mathbf{IV} = (IV_0, \dots, IV_{M-1})$ denotes the initial value of M bits. Furthermore, **IV** can be known and chosen by us, but **s** cannot be. Without loss of generality, we suppose that the length of the secret key is smaller than N bits, M is smaller than N and M, N are both the multiples of 8. At the beginning, the state **s** is initialized with some form of the secret key, that is to say, we can obtain the secret key if we can get the initial value of **s** (denoted $\mathbf{s}_0$). At every clock tick, **s** is shifted one bit to the left. The leftmost bit is discarded and the feed back bit is computed according to $c_N s_0 \oplus \cdots \oplus c_1 s_{N-1}$. Additionally, one bit of **IV** is XOR-ed with the feedback bit and then fed into **s** on the right. To generate the key stream, selected bits of **s** are put through the filter function.

3.2. Differential power analysis of stream ciphers with LFSRs

As described above, if we can get the initial value of **s**, we can obtain the key. In the rest of this chapter, we will show how to reveal the seckey by using DPA step by step.

Choose an intermediate value. Let us define the feedback bits of LFSR as vector $\mathbf{k} = (k_0, \dots, k_{M-1})$, where

$$\begin{aligned} &k_0 = c_1 s_{N-1} \oplus c_2 s_{N-2} \oplus \cdots \oplus c_N s_0, \\ &k_1 = c_1 k_0 \oplus c_2 s_{N-1} \oplus \cdots \oplus c_N s_1, \\ &\dots \\ &k_{M-1} = c_1 k_{M-2} \oplus \cdots \oplus c_{M-1} k_0 \oplus c_M s_{N-1} \oplus \cdots \oplus c_N s_{M-1}. \end{aligned}$$

It can be observed that if $\mathbf{k}$ is known, M bits of $\mathbf{s}$ can be calculated by solving a simple equation set, because $\mathbf{c}$ is given in the definition of $\mathbf{k}$. In Fig. 4, **IV** are the values can be known, so we can choose an intermediate value as an octet

$$\mathbf{v}_i = (IV_{4(i-1)} \oplus k_{4(i-1)}, \dots, IV_{4(i-1)+7} \oplus k_{4(i-1)+7}) \tag{3}$$

and define $\mathbf{k}^{(i)} = (k_{4(i-1)}, \dots, k_{4(i-1)+7})$, where $i = 1, \dots, M/4 - 1$. All $\mathbf{k}^{(i)}$ are the values that we can guess, and every bit of v_i is just the bit which is fed into $\mathbf{s}$ on the right.

Measure the power consumption. In this step, some different values should be given to **IV**, and then the corresponding power traces should be recorded when the cryptosystem process the different values of **IV**. Let $\mathbf{IV_i}$ have the values as follow, where $i = 1, \dots, M/4 - 1$: (X means arbitrary values which do not affect the whole analysis)

$$\begin{aligned} &\mathbf{IV}_1 : \text{0x00 XX} \cdots \text{XX} \sim \text{0xFF XX} \cdots \text{XX} \\ &\mathbf{IV}_2 : \text{0x000X} \cdots \text{XX} \sim \text{0x0F FX} \cdots \text{XX} \\ &\dots \\ &\mathbf{IV}_M/4 - 1 : \text{0x0000} \dots \text{00} \sim \text{0x0000} \dots \text{FF}. \end{aligned}$$

Obviously, each group contains 2^8 different values (0x00 $\sim$ 0xFF), as a result we can get $(M/4 - 1) * 2^8$ power traces. And those traces can be written as $\mathbf{t}_j = (t_{j,1}, \dots, t_{j,T})$, where $j = 1, \dots, 2^8$, T denotes the length of one trace. We can obtain the matrix $\mathbf{T}_i$ of size $2^8 \times T$, where $i = 1, \dots, M/4 - 1$.

Calculate hypothetical intermediate values. In the previous steps, we have already defined the intermediate value and given the values of **IV**. To calculate hypothetical intermediate values, we should give all the possible values of $\mathbf{k}$. As the definition of the intermediate value $\mathbf{v}_i$ in (3), it is an octet and we need a corresponding $\mathbf{k}^{(i)}$, which is also an octet, to calculate $\mathbf{v}_i$. All $\mathbf{k}^{(i)}$ should be guessed. Every $\mathbf{k}^{(i)}$, where $i = 1, \dots, M/4 - 1$, have 2^8 possible values from 0x00 to 0xFF. So we can get the matrix $\mathbf{V}_i$ of size $2^8 \times 2^8$ according to (3), and we also have $M/4 - 1$ such matrices.

Map intermediate values to power consumption values. In fact, the largest part of the power consumption of a stream cipher with the structure as shown in Fig. 4 must be the LFSR, because the basic part of LFSR is a register and the power consumption of every bit changes of the register is considerable. And for a certain bit of the register, value changes from 0 to 1 or 1 to 0 need much more power than changes from 0 to 0 or 1 to 1 when the LFSR is shifted. Through such characters of LFSR, we can map intermediate values to power consumption values. We map a hypothesis intermediate value in the matrix $\mathbf{V}$, $\mathbf{v}_{i,j} = (v_{i,j,0}, \dots, v_{i,j,7})$ to a hypothesis intermediate power consumption value $h_{i,j}$, by calculating $h_{i,j} = \sum_{n=0}^{6} \left| v_{i,j,n} - v_{i,j,n+1} \right|$, and the possible values of $h_{i,j}$ are $0, \dots, 7$. We should notice that this mapping is quite important, and it will affect the effectiveness of our approach. Then we can obtain the matrix $\mathbf{H}$ of size $2^8 \times 2^8$, and have $M/4-1$ such matrices. Furthermore, if we suppose the power consumption of the register when the value of one bit changes from 1 to 0 or from 0 to 1 are the same (in general, that is true for most hardware implementations), we can record only half of the power traces and the hypothesis values of $\mathbf{k}$ that we have in the previous steps. That is to say, we can give $(M/4-1)*2^7$ values to **IV** and guess 2^7 possible values of $\mathbf{k}^{(i)}$, because a certain value of **IV** or $\mathbf{k}$ will lead to the same results as its inversion value when we calculate $h_{i,j}$. For example, let $\mathbf{IV} = (1, 0, 1, 0, 1, 1, 0, 0, \dots)$ and $\mathbf{k} = (0, 1, 1, 0, 1, 0, 0, 1, \dots)$, then we can get $\mathbf{v_0} = (1 \oplus 0, \dots, 0 \oplus 1) = (1, 1, 0, 0, 0, 1, 0, 1)$ and the corresponding $h = |1 - 1| + \cdots + |0 - 1| = 4$. To compare, we get the inversion value of **IV**, let $\mathbf{IV}' = (0, 1, 0, 1, 0, 0, 1, 1, \dots)$ and $\mathbf{k}$ be the same, then $\mathbf{v}'_\mathbf{0} = (0 \oplus 0, \dots, 1 \oplus 1) = (0, 0, 1, 1, 1, 0, 1, 0)$ is also the inversion value of $\mathbf{v_0}$ and $h' = |0 - 0| + \cdots + |1 - 0| = 4 = h$.

Compare the hypothetical power consumption values with the power traces. In the final step, we should compare $\mathbf{H}$ with $\mathbf{T}$ and try to reveal the real value of $\mathbf{k}$. We have $\mathbf{H}_i$ of size $2^8 \times 2^8$ and $\mathbf{T}_i$ of size $2^8 \times T$, where $i = 1, \dots, M/4 - 1$, so we can compare each column of $\mathbf{H}_i$ with each column of $\mathbf{T}_i$ based on the correlation coefficient according to (2). As the comparison results of $\mathbf{H}_i$ and $\mathbf{T}_i$, we can obtain $\mathbf{R}_i$ of size $2^8 \times T$. There must be some elements with much larger absolute values than others in $\mathbf{R}_i$ which reveal the candidates of $\mathbf{k}$.

After the five steps above, we can obtain a much smaller set of possible $\mathbf{k}$. That is to say, after using our approach, we can decrease the number of possible $\mathbf{k}$ from 2^M to a much smaller value and then we can use brute force to find out the right $\mathbf{k}$, so that, we can obtain M bits information about the initial state $\mathbf{s}$. Then the number of possible values of the initial state of the LFSR reduce to around 2^{N-M} from 2^N. It is worth noting that we assume M is smaller than N, but if N is smaller than M, we can even obtain the unique value of the initial state $\mathbf{s_0}$.

In the next section, we will give an example to show how effective the DPA method we described above is.

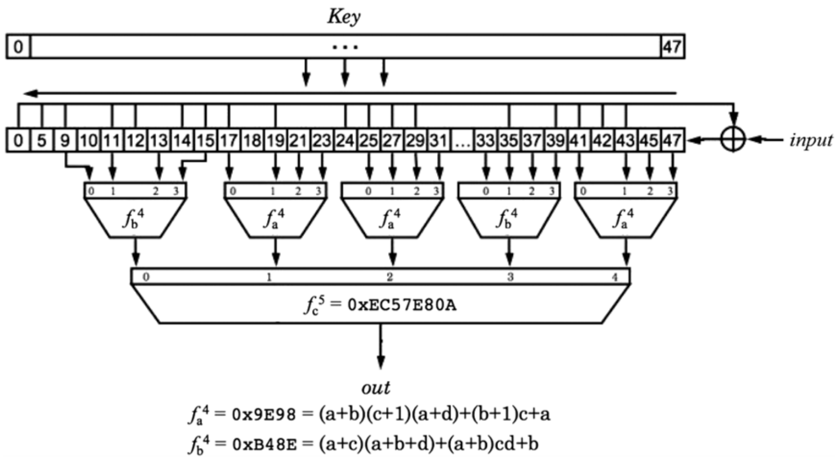
Key
0
47
input
f_b^4
f_a^4
f_a^4
f_b^4
f_a^4
f_c^5 = 0xEC57E80A
out
f_a^4 = 0x9E98 = (a+b)(c+1)(a+d)+(b+1)c+a
f_b^4 = 0xB48E = (a+c)(a+b+d)+(a+b)cd+b

Fig. 5. Crypto-1.

4. Differential power analysis of Crypto-1

4.1. Introduction of Crypto-1

Crypto-1 is a proprietary encryption algorithm created by NXP Semiconductors specifically for Mifare Classic contactless smart cards, including Oyster card, CharlieCard and OV-chipkaart. According to NXP, there are about 200 million Mifare cards in use around the world, covering 85% of the contactless smart card market. Because of the huge amount of Mifare cards, there are widespread concerns about their security. Recently, several papers [26–29] showed that Mifare cards had been attacked. The cryptanalysis in those papers are effective, but all the attacks on Mifare cards are based on the vulnerability of the random number generator and the authority protocol of the cards or the special bits of the register which are inputs to the filter function. Just a little improvement can help Mifare cards resist these attacks, or we can say these attacks are specifically targeting Mifare cards and Crypto-1, and they are even not valid for cards with a similar algorithm. Furthermore, no public research about the side-channel analysis of Mifare cards or Crypto-1 could be obtained at hand. The DPA attack described in the rest of this section on Crypto-1 will be also effective to all the variants of it.

Crypto-1 is a stream cipher whose core is a 48-bit LFSR and two levels of filter functions, as shown in Fig. 5. Crypto-1 also has a 32-bit input as the initial value.

To facilitate the description, we also use **s** to denote the state of the register, **IV** to denote the known input bits, and **k** to denote the feedback bits before being XOR-ed with input bits. Then the length of **s** and **IV** are 48 bits and 32 bits, respectively. Correspondingly, we only use 32 bits of feedback bits, so **k** is 32 bits.

At the beginning of Crypto-1, the 48-bit secret key is put into the register that means the initial state exactly contains all the bits of the secret key. The initial state can be written as $\mathbf{s}_0 = (s_0, \dots, s_{47})$. At every clock **s** is shifted one bit to the left and one bit is fed into **s** on the right. We can write the values of **s** at different times as $\mathbf{s}_i$, where i denotes the ith clock:

$$\mathbf{s}_1 = (s_1, \dots, s_{47}, k_0 \oplus IV_0),$$
$$\dots$$
$$\mathbf{s}_8 = (s_8, \dots, s_{47}, k_0 \oplus IV_0, \dots, k_7 \oplus IV_7),$$
$$\dots$$
$$\mathbf{s}_{32} = (s_{32}, \dots, s_{47}, k_0 \oplus IV_0, \dots, k_{31} \oplus IV_{31}).$$

32 clocks are enough for analyzing. At every clock there is also one bit output which is just one bit of the key streams produced by the current secret key. The 32-bit outputs during the 32 clocks can be known, too.

4.2. Differential power analysis of Crypto-1

Similar to Section 3.2, we will show how to use DPA to reveal the secret key of Crypto-1 in this part.

We also choose the intermediate value $\mathbf{v}_i$ in (3), but here $i = 1, \dots, 7$. It is obvious that $\mathbf{v}_i$ is just the last octet of $\mathbf{s}_{4(i+1)}$.

To measure the power consumption, we should implement the algorithm by hardware or software. In this paper, we use hardware emulation of Crypto-1 to get the data of power consumption. We implement the functional simulation of Crypto-1 by programming in the Verilog hardware description language. After synthesis and placing the route, we use VCS to get the power traces. We choose a certain secret key in this section to obtain the corresponding power traces.

We also give the same values to $\mathbf{IV}_i$ with those in Section 3.2, where $i = 1, \dots, 7$, so we can record 7×256 power traces. Fig. 6 shows parts of two power traces of the LFSR module. In practical situations, the power traces must be much noisier,

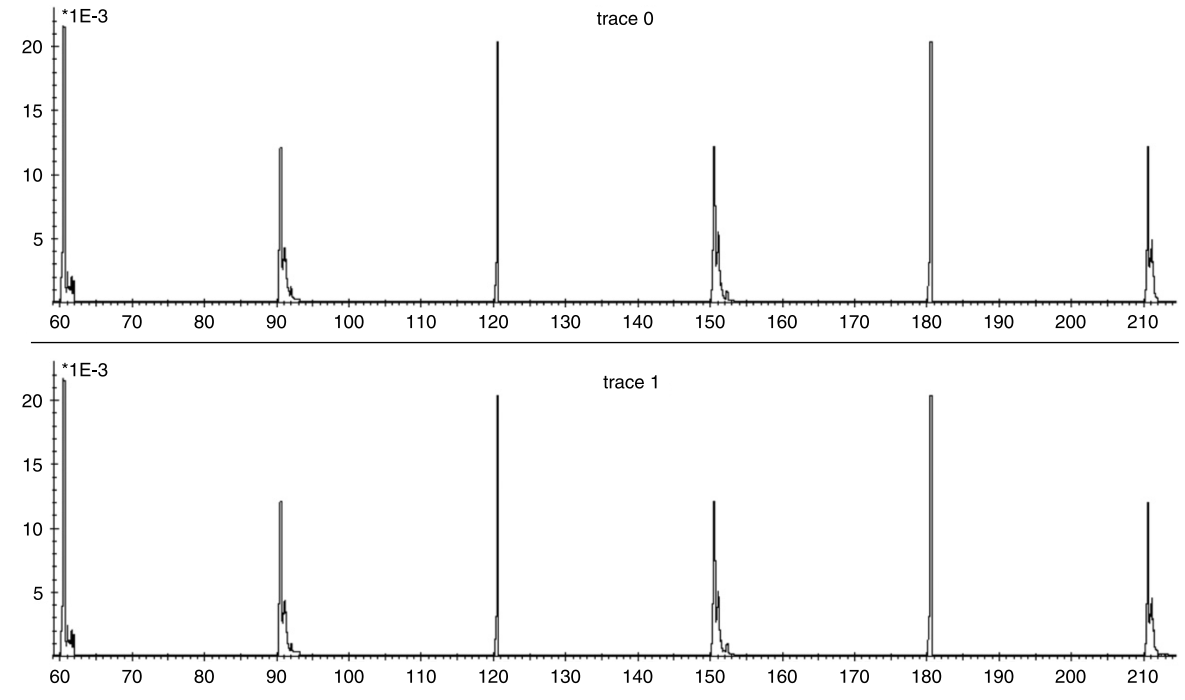

Fig. 6. Parts of the traces.

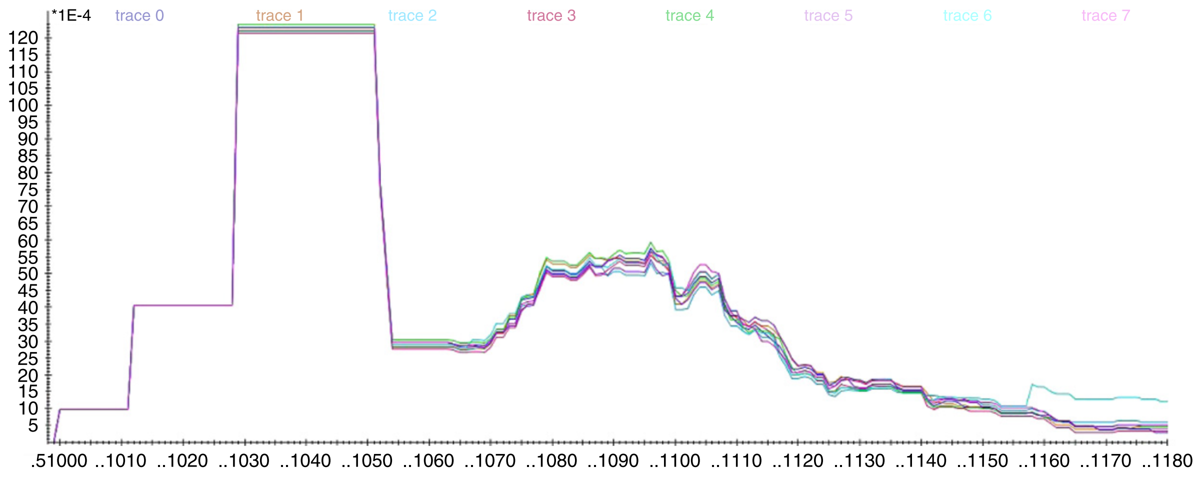

Fig. 7. Several lower peaks.

and maybe it is difficult to determine what positions of the power traces indicate the power consumption of LFSR. But we can use some mature methods to reduce the noise, and determine the positions by comparing different power traces.

Two peaks could be seen clearly in each trace at every clock. And it is easy to know that the higher peaks indicate the power consumption of input and output operations and the lower peaks indicate the power consumption which we are concerned about. We focus on the lower peaks and overlap several of them, as shown in Fig. 7.

By comparing different **IV**'s and the corresponding traces, we can find that the highest peak in Fig. 7 indicates the power consumption of the LFSR. We just pick up the largest power consumption value at the fixed time which we concern from each power trace, and those values are enough for the next steps. For example, for all the values of $\mathbf{IV}_1$, we concern the power consumption of the shifting operation to $\mathbf{s}_8$ which we defined in Section 4.1, so we can just pick up the power consumption value at the fixed time that the register with the state $\mathbf{s}_8$ is shifted one bit to the left. So for each $\mathbf{IV}_i$, there is a corresponding $\mathbf{T}_i$ of size $2^8 \times 1$, where $i = 1, \ldots, 7$.

As we described in Section 3.2, $\mathbf{k}^{(i)}$, $(k_{4(i-1)}, \ldots, k_{4(i-1)+7})$, has 2^8 possible values from 0x00 to 0xFF, so we can obtain the matrix $\mathbf{V}_i$ of size $2^8 \times 2^8$ according to (3) and the 2^8 values of $\mathbf{IV}_i$, where $i = 1, \ldots, 7$. And we have already defined how to map $\mathbf{V}_i$ to $\mathbf{H}_i$, so for each $\mathbf{V}_i$ we can obtain a corresponding $\mathbf{H}_i$.

Table 1
Some correlation coefficients.

	0x0	0x8		0x7	0xF
0x0	0.01770	0.2342	0x8	0.3694	**0.5857**
0x1	−0.09277	−0.3091	0x9	0.2589	0.04238
0x2	−**0.5858**	−0.3693	0xA	−0.2341	−0.01775
0x3	−0.04242	−0.2588	0xB	0.3093	0.09273
0x4	0.09273	0.3093	0xC	−0.2588	−0.04242
0x5	−0.01775	−0.2341	0xD	−0.3693	−**0.5858**
0x6	0.04238	−0.2589	0xE	−0.3091	−0.09277
0x7	**0.5857**	−0.3694	0xF	0.2342	0.01770

Now we should try to reveal the real value of $\mathbf{k}$ by comparing $\mathbf{H}$ with $\mathbf{T}$. We have $\mathbf{H}_i$ of size $2^8 \times 2^8$ and $\mathbf{T}_i$ of size $2^8 \times 1$, where $i = 1, \ldots, 7$. And the jth column of $\mathbf{H}$ indicates the key hypothesis k_j. Then we can calculate $\mathbf{R}_i$ of size $2^8 \times 1$ based on (2) and find candidate values of $\mathbf{k}$.

According to the certain secret key we used in this section, we can get the exact values of $\mathbf{R}_i$. For example, some elements in $\mathbf{R}_1$ are presented in Table 1.

In Table 1, the elements in the first columns of the left and the right part, "0x0, ..., 0xF", indicate the values of (k_0, k_1, k_2, k_3) and the elements in the first row, "0x0, 0x8" in the left part and "0x7, 0xF" in the right, indicate the values of (k_4, k_5, k_6, k_7). Other elements are the corresponding correlation coefficients. For example, in the table the number underlined "0.5857" is the correlation coefficient corresponding to $(k_0, \ldots, k_7)$ with the value "0x20". After observing all the elements of $\mathbf{R_1}$, we can find several features:

(1) A certain value of $(k_0, \ldots, k_7)$ and its inversion value generate the same correlation coefficient, for example, 0x00 and 0xFF generate the same correlation coefficient "0.01770".
(2) From 0x00, every 8 consecutive values of $(k_0, \ldots, k_7)$ generate very close correlation coefficients, for example, the correlation coefficients corresponding to 0x00 $\sim$ 0x07 are 0.01770, 0.01770, 0.01765, 0.01765, 0.01775, 0.01775, 0.01780, 0.01780 which are very close to 0.01775 and those corresponding to 0x70 $\sim$ 0x77 are very close to −0.5857.

The first feature tells us that we can just measure half of the power consumption. By picking up the correlation coefficients which have much larger absolute values and their corresponding values of $(k_0, \ldots, k_7)$, we can reduce the possible values of $(k_0, \ldots, k_7)$. We can find that $(k_0, \ldots, k_7)$ with the values 0x20 $\sim$ 0x27, 0x70 $\sim$ 0x77 and their inversion values, have the absolute values of the correlation coefficients close to 0.586, which is much larger than others. It means that there are also 2^5 candidate values of $(k_0, \ldots, k_7)$ instead of 2^8 possible values. It is not a good result.

4.3. An improved approach for the analysis of Crypto-1

We should find a way to further reduce the candidate values. Considering the definition of $\mathbf{H}_i$, the elements with larger values in $\mathbf{H}_i$ should indicate more power consumption which can be viewed in $\mathbf{T}_i$. And each row of $\mathbf{H}_i$ and $\mathbf{T}_i$ corresponds to the same value of $\mathbf{IV}_i$, so the trend of the magnitude of the elements in each column of $\mathbf{H}_i$ should be similar to that of $\mathbf{T}_i$. We can compare the first-order differential of the elements in each column of $\mathbf{H_i}$ with that of $\mathbf{T}_i$. According to this, the candidate values of $(k_0, \ldots, k_7)$ left are 0x70 $\sim$ 0x77 and their inversion values, 0x88 $\sim$ 0x8F. There are also 2^4 candidate values, but the value of $(k_0, \ldots, k_4)$ has only 2 possibilities and the value of (k_5, k_6, k_7) cannot be determined.

We continue to compare $\mathbf{H}_2$ with $\mathbf{T}_2$, then we can also obtain the candidate values of $(k_4, \ldots, k_{11})$ which are 0x10 $\sim$ 0x17 and their inversion values. So the value of (k_4, k_5, k_6, k_7) must be 0x1 or 0xE. We have already known that the candidate values of $(k_0, \ldots, k_7)$ are 0x70 $\sim$ 0x77 and 0x88 $\sim$ 0x8F. By combining the results, we can obtain that the value of $(k_0, \ldots, k_7)$ must be 0x71 or 0x8E. In fact, 0x71 is correct according to the chosen secret key.

After comparing all $\mathbf{H_i}$ with $\mathbf{T}_i$, there are only two possibilities of $(k_0, \ldots, k_{28})$, but the value of (k_{29}, k_{30}, k_{31}) cannot be determined, either. It means that at first $\mathbf{k}$ has 2^{32} possible values, while after being analyzed by our DPA method, there are only 2^4 possibilities left.

To reveal the secret key, we have to know all the bits of $\mathbf{s_0}$. Because the bits in $\mathbf{k}$ are the linear combination of the bits in $\mathbf{s_0}$ and we have equivalently 28 bits information of $\mathbf{k}$, we can obtain 28 bits information of $\mathbf{s_0}$. The remaining bits of $\mathbf{s_0}$ can be obtained by brute force or other attacks.

5. Conclusion and further work

In this paper, we firstly make a brief introduction to differential power analysis and its common steps. Then we show how to use our DPA method to analyze the stream ciphers with LFSRs. To describe our DPA method, we start off on trying to give a general and simple stream cipher model which contains the basic components of a stream cipher. Although the structure we given in Section 3.1 is quite simple, it is sufficient to illustrate our purpose that we mainly try to show how to use our DPA method to analyze the LFSRs. Our approach illustrates the vulnerabilities of LFSR in the analysis of DPA and can be regarded as general DPA techniques. As an example, we choose to analyze Crypto-1 which has never been attacked by

DPA, and never by SCA. Only through the analysis of some power consumption data, the complexity of an exhaustive search of Crpyto-1 is reduced to 2^{20} from 2^{48}. Though Crypto-1 is not very complicated, we just want to demonstrate how to use our approach and its effectiveness in a practical example. Because of the widespread use of LFSR in stream ciphers, our work is significant. Finally, we hope that our work would be helpful in analyzing a variety of stream ciphers with LFSRs by using side-channel analysis methods. As a scope for further research, we will work on the issues about the expected performance of our approach. For example, for the structure given in Section 3.1, we can reduce the complexity of exhaustive search to around 2^{N-M}, but which is above 2^{N-M}, not exactly equal to 2^{N-M}. And in some situations 2^{N-M} could be considerably large, so we should find a way to improve our approach. Furthermore we will apply our approach to other stream ciphers in practical situations.

Acknowledgments

The authors are grateful to the editor and all the anonymous reviewers for valuable suggestions which improved the paper. Also, they would like to thank the engineers from Shanghai Huahong Integrated Circuit Company for helping them to finish the hardware stimulations in this paper.

References

[1] P. Kocher, J. Jaffe, B. Jun, Differential power analysis, Lecture Notes in Computer Science (1999) 388–397.
[2] P. Kocher, Timing attacks on implementations of Diffie–Hellman, RSA, DSS, and other systems, Crypto 96 (1996) 104–113.
[3] P. Kocher, J. Jaffe, B. Jun, Introduction to differential power analysis and related attacks, 1998.
[4] D. Agrawal, B. Archambeault, J.R. Rao, P. Rohatgi, The EM side-channel (s), Lecture Notes in Computer Science (2002) 29–45.
[5] J.J. Quisquater, D. Samyde, Electromagnetic analysis (EMA): measures and countermeasures for smart cards, Lecture Notes in Computer Science (2001) 200–210.
[6] E. Brier, C. Clavier, F. Olivier, Correlation power analysis with a leakage model, Lecture Notes in Computer Science (2004) 16–29.
[7] S. Chari, C. Jutla, J. Rao, P. Rohatgi, Towards Sound Approaches to Counteract Power-Analysis Attacks, Springer, 1999, pp. 791–791.
[8] J.S. Coron, P. Kocher, D. Naccache, Statistics and Secret Leakage, Springer, 2001, pp. 157–173.
[9] T.H. Le, J. Clédière, C. Canovas, B. Robisson, C. Serviere, J.L. Lacoume, A proposition for correlation power analysis enhancement, in: Cryptographic Hardware and Embedded Systems—CHES 2006, 2006, pp. 174–186.
[10] Z. Liu, X. Guo, Y. Chen, Y. Han, X. Zou, On the ability of AES S-boxes to secure against correlation power analysis, in: Information Security Practice and Experience, 2007, pp. 43–50.
[11] R. Mayer-Sommer, Smartly Analyzing the Simplicity and the Power of Simple Power Analysis on Smartcards, Springer, 2000, pp. 78–92.
[12] W. Fischer, B. Gammel, O. Kniffler, J. Velten, Differential power analysis of stream ciphers, in: Topics in Cryptology—CCT-RSA 2007, 2006, pp. 257–270.
[13] L. Goubin, J. Patarin, DES and Differential Power Analysis the "Duplication" Method, Springer, 1999, pp. 728–728.
[14] J. Goli, C. Tymen, Multiplicative masking and power analysis of AES, in: Cryptographic Hardware and Embedded Systems—CHES 2002, 2003, pp. 31–47.
[15] J.S. Coron, Resistance Against Differential Power Analysis for Elliptic Curve Cryptosystems, Springer, 1999, pp. 725–725.
[16] J.M. Chang, H.Y. Yang, H.C. Chao, J.L. Chen, Multipath design for 6LoWPAN ad hoc on-demand distance vector routing, International Journal of Information Technology, Communications and Convergence 1 (2010) 24–40.
[17] K. Jerbi, M. Wipliez, M. Raulet, O. Déforges, M. Babel, M. Abid, Automatic method for efficient hardware implementation from RVC-CAL dataflow: a LAR coder baseline case study, Journal of Convergence 1 (2010) 85–92.
[18] W.Y. Liang, P.T. Lai, C.W. Chiou, An energy conservation DVFS algorithm for the Android operating system, Journal of Convergence 1 (2010) 93–100.
[19] D. Kumar, T.C. Aseri, R. Patel, Multi-hop communication routing (MCR) protocol for heterogeneous wireless sensor networks, International Journal of Information Technology, Communications and Convergence 1 (2011) 130–145.
[20] C. Rechberger, E. Oswald, Stream ciphers and side-channel analysis, in: Citeseer, 2004.
[21] S. Chari, J. Rao, P. Rohatgi, Template attacks, in: Cryptographic Hardware and Embedded Systems—CHES 2002, 2003, pp. 51–62.
[22] J.J. Hoch, A. Shamir, Fault analysis of stream ciphers, in: Cryptographic Hardware and Embedded Systems—CHES 2004, 2004, pp. 1–20.
[23] J. Lano, N. Mentens, B. Preneel, I. Verbauwhede, Power analysis of synchronous stream ciphers with resynchronization mechanism, 2004, pp. 327–333.
[24] S. Kumar, K. Lemke, C. Paar, Some thoughts about implementation properties of stream ciphers, in: Citeseer, 2004.
[25] B. Gierlichs, L. Batina, C. Clavier, T. Eisenbarth, A. Gouget, H. Handschuh, T. Kasper, K. Lemke-Rust, S. Mangard, A. Moradi, Susceptibility of eSTREAM candidates towards side channel analysis, in: Proceedings of SASC, 2008, pp. 123–150.
[26] F. Garcia, G. de Koning Gans, R. Muijrers, P. Van Rossum, R. Verdult, R. Schreur, B. Jacobs, Dismantling MIFARE classic, in: Computer Security—ESORICS 2008, 2008, pp. 97–114.
[27] F.D. Garcia, P. van Rossum, R. Verdult, R.W. Schreur, Wirelessly Pickpocketing a Mifare Classic Card, IEEE, 2009, pp. 3–15.
[28] G. de Koning Gans, J.H. Hoepman, F. Garcia, A practical attack on the MIFARE classic, in: Smart Card Research and Advanced Applications, 2008, pp. 267–282.
[29] T. Kasper, M. Silbermann, C. Paar, All you can eat or breaking a real-world contactless payment system, in: Financial Cryptography and Data Security, 2010, pp. 343–350.