CONCURRENCY AND COMPUTATION: PRACTICE AND EXPERIENCE
Concurrency Computat.: Pract. Exper. 2014; **26**:1531–1545
Published online 20 June 2013 in Wiley Online Library (wileyonlinelibrary.com). DOI: 10.1002/cpe.3068

SPECIAL ISSUE PAPER

Power analysis based reverse engineering on the secret round function of block ciphers

Ming Tang [1,2,*,†], Zhenlong Qiu [2], Weijie Li [2], Weijin Sun [2], Xiaobo Hu [3] and Huanguo Zhang [1,2]

[1] *State Key Lab. of AIS & TC, Ministry of Education, Wuhan University, Wuhan 430072, China*
[2] *School of Computers, Wuhan University, Wuhan 430072, China*
[3] *State Grid Chip Design Center, Beijing 10000, China*

SUMMARY

The recent cryptanalysis on block ciphers has two major trends. Side channel analysis (SCA) has become a new threat to the hardware implementations of encryption algorithms. On the other hand, reverse engineering has been adopted to explore the unknown part of the encryption algorithms, which has become a new target of the cryptanalysis. Some drawbacks have been found in the existing methods of reverse engineering, which target on the special structures or utilize the flaws in the unknown parts. The major disadvantage is that the number of rounds to be analyzed is limited, and the complexity is high. The existing SCAs for reverse engineering depend on the leakage models in a large extent and mainly focus on the single component of the algorithms, whereas the other parts of the target algorithm are known. In this paper, we present a more general and feasible reverse analysis by combining the mathematical methods and the SCA methods. We use the strict avalanche criterion for the non-linear operations of block ciphers and apply the power analysis to reverse the structure parameters. We propose a new reverse analysis method to reduce the dependency on the leakage models, which can be combined with the structural cryptanalysis to reverse the internal parameters of the linear and non-linear operations. We finally achieve the reverse analysis on the unknown round function of block ciphers. Copyright © 2013 John Wiley & Sons, Ltd.

Received 28 December 2012; Revised 17 March 2013; Accepted 18 May 2013

KEY WORDS: block ciphers; SCA; reverse engineering; cryptanalysis

1. INTRODUCTION

Most of the existing cryptanalysis methods, including both the traditional mathematic analyses [1–4] and side channel analyses (SCAs) [5–10], have been built on the public algorithms and their specific implementations. However, there are many cryptographic algorithms whose implementation details are partly or totally unknown. These secret algorithms have been widely used in military and government. Some commercial cryptographic algorithms are unknown as well, such as Twofish [11], MARS [12], and A3/A8 [13]. Therefore, reverse engineering plays an important role on analyzing the security of the secret cryptographic algorithms.

1.1. Related work

The existing reverse engineering methods mainly include the mathematics-based reverse analyses [14–16] and SCA-based reverse analyses [17–22]. Invasive attack is another reverse analysis

*Correspondence to: Ming Tang, State Key Lab. of AIS & TC, Ministry of Education, Wuhan University, Wuhan 430072, China.
†E-mail: m.tang@126.com

which reconfigures the logic circuits through different pairs of input and output. However, this method needs complicated requirements and advanced techniques, so they will not be discussed in this paper.

Mathematics-based Reverse Engineering

In EUROCRYPT 2001, Biryukov and Shamir proposed the structure analysis on the cipher with SASAS structure [14] and succeed in recovering the unknown parameters of SPN (Sbox Permutation Network) with three rounds. Considering its high complexity, the structure analysis recovers the parameters of the unknown S-box after building the relationship between the input and output of the S-box. Moreover, they tried to find the equality of the unknown linear operation through the collision and can recover the two and half rounds of block cipher with SPN structure. Biryukov improved their method in 2010 [23]. However, the complexity of the structure analysis was too high to the real algorithms with higher rounds. Borghoff utilized this method on the PRESENT-like Maya algorithm in 2011 [15] to recover the unknown S-box, whose size is only 4×4.

Another type of mathematics-based reverse analysis is to recover the secret S-box in the C2 algorithm in CRYPTO 2009 [16]. This method utilized the design defaults of C2 where several subkeys or plaintexts do not go through the S-box.

The existing mathematical reverse analyses have demonstrated the feasibility to recover the unknown parts in encryption algorithms, although they all have several disadvantages such as high complexity, small number of rounds to be analyzed, and algorithm with special structure.

SCA-based Reverse Engineering

This method is also called as SCARE (Side Channel Analysis of Reverse Engineering). The existing SCAREs are mainly conducted through the power analysis. Roman Novak firstly proposed a power analysis to recover the unknown S-box in A3/A8 algorithm [21] , which is considered as the first work in SCARE. After several years, there have been more SCAREs against other algorithms at both hardware and software levels [17, 18, 20]. In 2010, Sylvain Guilley proposed a new SCA, which is claimed to reverse any kind of algorithms [19]. These SCAREs are all based on Correlation Power Analysis (CPA) [6], which is highly dependent on the knowledge of the leakage model.

In SCARE, adversaries are assumed to know the basic structure of the algorithm. The target of SCARE is the unknown operation of algorithm. The reverse analysis could be considered to recover the parameters of the unknown operation.

To improve the practicability of the reverse analysis, we propose a new general reverse analysis method on block ciphers.

1.2. Our contributions

We proposed a new reverse engineering analysis composed of two phases, structure analysis and parameter analysis.

Reverse analysis on the structure of block ciphers

The target of the reverse analysis proposed in this paper is the round function of block ciphers, which could be defined as the structure in Figure 1. Based on the strict avalanche criterion of S-box and power analysis, we propose the reverse analysis to recover the structure of the round. The complexity of our reverse analysis is $O(m)$, where m generally ranges from 1 to 16.

POLAR DPA-based reverse analysis on the parameters of unknown round function

We proposed a new reverse engineering method to recover the unknown round function based on the POLAR differential power analysis (DPA) technique. We found the different spikes of DPA have different biases, which could be used to recover the secret of the algorithm. Our method utilizes the bias of DPA to efficiently reverse engineering the unknown operations within the round. The complexity of this reverse engineering is $O(k \times (n-m))$ for the linear operation and $O(k \times 2^m)$ for the S-box, where n is the size of initial input, m is the size of input/output in the S-box, k is the number of S-boxes, $n = m \times k$.

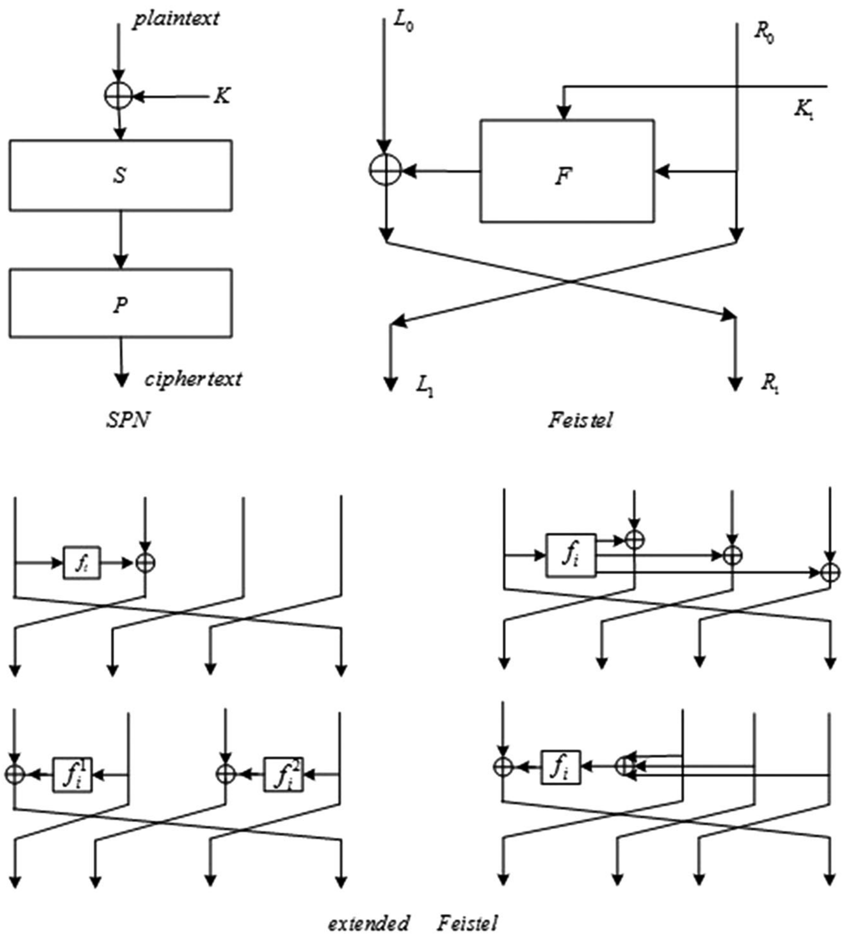

Figure 1. Common structures of block cipher.

1.3. Organization

This paper is organized as follows: Section 2 introduces the structure of block cipher and the POLAR DPA. Section 3 illustrates the reverse analysis both on the structure and inner operations. Section 4 gives the practical experiments of the reverse analysis. The final part is the conclusion.

2. PRELIMINARY

2.1. Definition of block cipher

We choose block ciphers as the target of reverse analysis. Most of the existing block ciphers have interactive structure, including SPN, Feistel, and extended Feistel structure as Figure 1.

In the common structures of block ciphers, such as Feistel and extended Feistel structure, the round function can be viewed as the combination of non-linear and linear operations. Figure 2 illustrates an example of the round function where S represents the non-linear operation and P is the linear one. The target of our reverse engineering analysis is to explore both S and P layer.

2.2. Differential power analysis

The SCAs have drawn a lot of attention since Paul Kocher proposed the DPA [9] in 1999. The basic idea of power analysis (PA) is to reveal the secret key with the additional power leakages from the cryptographic device. The existing PAs includes SPA, DPA, CPA, and MIA [6, 9, 24].

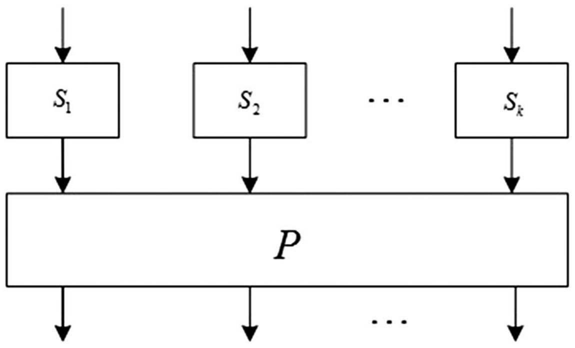

Figure 2. The structure of the round function.

For most of DPA attacks, the main ideas and principles originate from the model proposed by Kocher. Similar to other SCAs, DPA attacks are composed of two stages, data collection and statistical analysis. The basic DPA attack process is given below.

1. The attackers build the key-dependent selection function D denoted by $D(k_i, C_i)$, where k_i is the guessed key and C_i is the ciphertext, i refers to the ith data.
2. In the data collection stage, m power traces are collected when encryptions performed with random plaintexts and the fixed key. The set of power traces is denoted by P. For one guessed key, we can calculate the value of the selection function D by using both the guessed keys and the ciphertexts, denoted as $D\left(d_1^{k_i}, d_2^{k_i}, \cdots, d_M^{k_i}\right)^T$. The set P can be partitioned into two subsets according to D, denoted as P_1 and P_0, respectively.
3. In the statistical analysis stage, we exhaustively search k_i so as to find out the correct key. Each trace in the set P can be denoted by $T_i[j]$ (i refers to the ith trace, and j represents the jth sample point). Finally, the differential power trace can be calculated according to the equation (1).

$$\Delta D[j] = \overline{P_1} - \overline{P_0} = \frac{\sum_{i=1}^{M} D(k_i, C_i) T_i[j]}{\sum_{i=1}^{M} D(k_i, C_i)} - \frac{\sum_{i=1}^{M} (1 - D(k_i, C_i)) T_i[j]}{\sum_{i=1}^{M} (1 - D(k_i, C_i))} \tag{1}$$

The differential power traces show the significant spikes when the guessed key is correct. This conclusion has been proved by Kocher [9].

DPA is based on the fact that there is a high correlation between the power dissipation and the data that is calculated in some devices. If the key is correct, spikes will arise in regions where D is related to the processing data. In addition, we find that the selection function D is the most important factor in data analysis stage.

2.3. *Polar DPA*

Different from the traditional DPA, the POLAR DPA builds two selection functions,

$$D_1 = A \wedge C_1\ ,\ D_2 = A \wedge C_2 \tag{2}$$

D_1,D_2 represent the differential value and C_1,C_2 are the unknown intermediates of encryption algorithm we want to know. A is the known variable that is used to partition the traces. Similarly, to exploit bias signal in DPA, the POLAR DPA exploits the two DPA traces regarding the two selection functions to identify the value of $C_1 \wedge C_2$. We illustrate the POLAR DPA in Figure 3.

The upper half of Figure 3 indicates the situation when the bias of the differential power traces corresponding to D_1 and D_2 are identical, whereas the lower half represents the opposite situation, that is to say, the biases are opposite. Based on these, we derive **Proposition 1**.

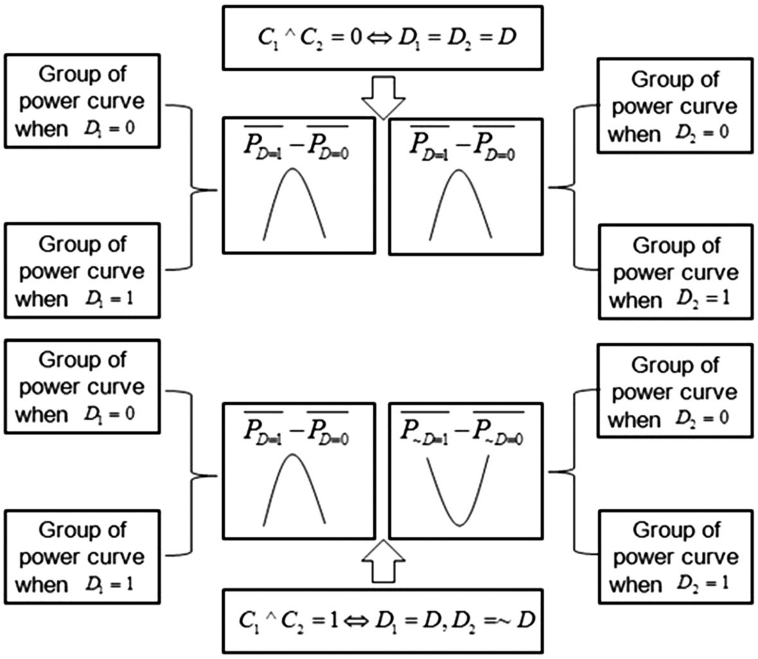
$C_1 \wedge C_2 = 0 \Leftrightarrow D_1 = D_2 = D$
Group of power curve when $D_1 = 0$
Group of power curve when $D_1 = 1$
$\overline{P_{D=1}} - \overline{P_{D=0}}$
$\overline{P_{D=1}} - \overline{P_{D=0}}$
Group of power curve when $D_2 = 0$
Group of power curve when $D_2 = 1$
Group of power curve when $D_1 = 0$
Group of power curve when $D_1 = 1$
$\overline{P_{D=1}} - \overline{P_{D=0}}$
$\overline{P_{\sim D=1}} - \overline{P_{\sim D=0}}$
Group of power curve when $D_2 = 0$
Group of power curve when $D_2 = 1$
$C_1 \wedge C_2 = 1 \Leftrightarrow D_1 = D, D_2 = \sim D$

Figure 3. The effectiveness of the POLAR DPA.

Proposition 1
For two selection functions in $GF(2)$, $D_1 = A \wedge C_1$, $D_2 = A \wedge C_2$, $A, C_1, C_2, D_1, D_2 \in GF(2)$, $\wedge$ represents the bitwise XOR operation in $GF(2)$, $\Delta\overline{P}_1, \Delta\overline{P}_2$ represent the differential biases of D_1 and D_2, respectively. We have $C_1 \wedge C_2 = \Delta\overline{P}_1 \wedge \Delta\overline{P}_2$.

Proof
If $C_1 \neq C_2 \Leftrightarrow C_1 \wedge C_2 = 1$:

$$\begin{aligned}
&\Rightarrow D_2 = \sim D_1 \\
&\Rightarrow \overline{P}_{D_1} = \overline{P}_{D_1=1} - \overline{P}_{D_1=0} \\
&\Rightarrow \overline{P}_{D_2} = \overline{P}_{D_2=1} - \overline{P}_{D_2=0} = \overline{P}_{\sim D_1=1} - \overline{P}_{\sim D_1=0} = \overline{P}_{D_1=0} - \overline{P}_{D_1=1} \\
&\Rightarrow \overline{P}_{D_1} = \sim \overline{P}_{D_2}
\end{aligned}$$

If $C_1 = C_2 \Leftrightarrow C_1 \wedge C_2 = 0$:

$$\begin{aligned}
&\Rightarrow D_2 = D_1 \\
&\Rightarrow \overline{P}_{D_1} = \overline{P}_{D_1=1} - \overline{P}_{D_1=0} \\
&\Rightarrow \overline{P}_{D_2} = \overline{P}_{D_2=1} - \overline{P}_{D_2=0} = \overline{P}_{D_1=1} - \overline{P}_{D_1=0} \\
&\Rightarrow \overline{P}_{D_1} = \overline{P}_{D_2}
\end{aligned}$$

□

Through the POLAR DPA, we could obtain a series of differential output values, which could be used to recover the inner parameters of unknown operations in the round function.

3. THE REVERSE ENGINEERING ON THE ROUND FUNCTION

3.1. Reverse engineering on the structure of round function

Based on the definition of the round function in Figure 2, the goal of the reverse analysis of the structure of round function is to explore the size of S-boxes. Inspired by the strict avalanche criterion of S-box [25], we proposed the structure reverser analysis.

Definition 1
Consider X and X_i, two n-bit, binary plaintext vectors, such that X and X_i differ only in bit i, $1 < i < n$. Let

$$V_i = Y \wedge Y_i,$$

where $Y = f(X)$, $Y_i = f(X_i)$, and f is the cryptographic transformation under consideration. If f is to meet the strict avalanche criterion, the probability that each bit in V_i is equal to 1 should be one half over the set of all possible plaintext vectors X and X_i.

Our reverse analysis is based on the PA, which consists of three parts: static power, dynamic power and noise as following:

$$P_{\texttt{total}} = P_{\texttt{dynamic}} + P_{\texttt{noise}} + P_{\texttt{static}} \tag{3}$$

$P_{\texttt{dynamic}}$ is the dominant part in the total power consumption, which is generated during the transformation of the output signals. When we input the same pair of plaintext and key, there is no transformation of the output signals in the second time, and the power value is close to the sum of $P_{\texttt{noise}}$ and $P_{\texttt{static}}$, which is named as $P'_{\texttt{total}}$. When we change the plaintext, we can obtain the $P_{\texttt{dynamic}}$ by subtracting $P'_{\texttt{total}}$ from the total consumption $P_{\texttt{total}}$. Therefore, we utilize the dynamic power consumption to achieve the reverse analysis.

We could combine the strict avalanche criterion and the PAs to recover the size of S-boxes.

Hypothesis
From 2.1, we list the hypotheses of the round function as the following:

- There are only S layer and P layer in the round function.
- The input size of S-box is numbered as m.
- The input size of a round equals to the output size and is numbered as n, and k is the number of S-boxes in one layer.
- Each S-box is satisfied with the strict avalanche criterion.
- Regardless of the bit location, the power consumption for any bit transformation is approximately the same, which is denoted as P_{1bit}.

Algorithm 1
Based on the strict avalanche criterion, half of the output of the S-box would be changed when one-bit of the S-box input changed, which generated the transformation power consumption of $\frac{m}{2} P_{1bit}$. If there are two-bit changed, it could generate at least $\frac{m}{2} + 1$ bits to be changed, which is the reason why we need to alter at most $\frac{m}{2} + 1$ bits to guarantee all bits of S-box changed. The steps of **Algorithm 1** are listed as follows:

Step1: Guess the size of S-box, which is defined as gm.
Step2: Change the first bit of S-box input, and the power consumption of transformation is represented as P_{d1}.
Step3: Change $\frac{gm}{2} + 1$ input bits of S-box from high to low order, and power consumption of transformation is denoted as P_{d2}. If P_{d2}/P_{d1} is not equal to 2, it ought to be a wrong guessed size; otherwise, the flow goes to Step4.

Step4: Change $gm + 1$ input bits of S-box from high to low order, and the dynamic power consumption is P_{d3}. If P_{d3}/P_{d1} is close to 3, it can be regarded as a correct guess; otherwise, the guess should be a wrong one and the flow is back to Step1 to find another guessed size of S-box.

The proof of effectiveness

To guarantee the effectiveness of **Algorithm 1**, we give the proof of it.

- If $gm = m$, to change an input bit, the dynamic power consumption is $\frac{m}{2} P_{1bit}$. The dynamic power consumption is mP_{1bit} when $\frac{gm}{2} + 1$ input bits are changed and all output bits of S-box are all switching. The dynamic power consumption is $\frac{3m}{2} P_{1bit}$ with the $gm + 1$ changing bits. Therefore, when the guessed size is correct, P_{d2}/P_{d1} equals to 2 and P_{d3}/P_{d1} equals to 3.
- If $gm < m$ and $\frac{gm}{2} + 1$ input bits are changed, it is possible for all the output of S-box switching, or not all the output switching. P_{d2}/P_{d1} maybe 2 or less than 2. If P_{d2}/P_{d1} is 2, it needs to change $gm + 1$ input bits. Because $gm + 1 \leqslant m$ and the corresponding dynamic power consumption are less than mP_{1bit}, the value of P_{d3}/P_{d1} must be less than 2.
- If $gm > m$, we change $\frac{gm}{2} + 1$ input bits; all of the output bits of the first S-box must be switching, and the changing bits may influence the second S-box to generate another part of dynamic power consumption. Therefore, the value of P_{d2}/P_{d1} may be 2 or more than 2. If the P_{d2}/P_{d1} is equal to 2, it is needed to alter $gm+1$ input bits. As $gm+1 > m+1$, the dynamic power consumption is more than $\frac{3m}{2} P_{1bit}$, P_{d3}/P_{d1} should be more than 3.

Complexity of Algorithm 1

To analyze the complexity of **Algorithm 1**, we give the pseudocode as follows.

```
Algorithm 1: GetSboxSize
Input: $n$    //generally ranging from 1 to 16
Output: $m$

Begin
To change the first bit of $S$-box to obtain the dynamic power consumption $P_{d1}$
For( i = 2; i ⩽ n; i++)
  { To change $\frac{i}{2} + 1$ input bits of $S$-box to obtain the dynamic power consumption $P_{d2}$
    If ($P_{d1}/P_{d2}$ == 2)
    { To change $i + 1$ input bits of $S$-box to obtain the dynamic power consumption $P_{d3}$
      If($P_{d3}/P_{d1}$ == 3)  then $m = i$; return $m$;
      Else continue; }
    Else continue;
  }
  End
```

We have analyzed the complexity of **Algorithm 1**. It is necessary to make $(m-1)$ times of guess and $(2m-1)$ times of power samples to acquire the correct size of S-box. The complexity of the structure reverse analysis is $O(m)$.

3.2. Reverse engineering on the parameters of inner operations

Combing the structure analysis and the POLAR DPA, we propose the reverse analysis to recover the P operation and to acquire the parameters of S-box with the POLAR DPA. The details of the block cipher are given in Figure 4. Supposing that the P is a surjection operation.

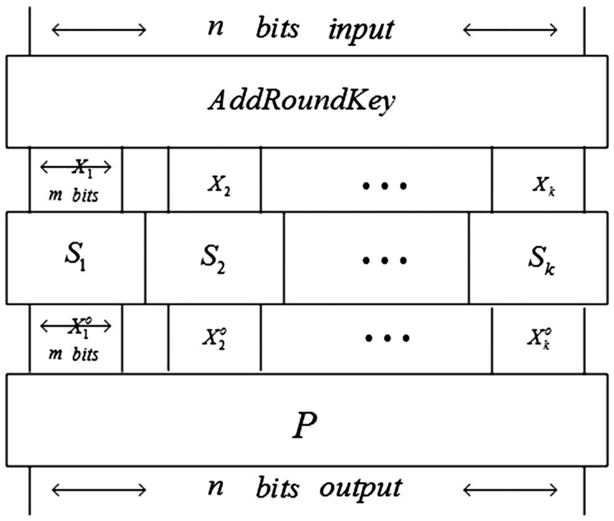

Figure 4. The detailed structure of the round function.

The Reverse Engineering on the parameters Layer P

As the layer of P, if the size of input and output is n, P could be represented as $n \times n$ matrix with $0/1$ elements. The reverse engineering of P equals to obtain the matrix, and P could be represented as the following equation:

$$Y = A \bullet X^0 = \begin{pmatrix} a_{11} & a_{12} & \dots & a_{1n} \\ a_{21} & a_{22} & \dots & a_{2n} \\ \dots & \dots & \dots & \dots \\ a_{n1} & a_{n2} & \dots & a_{nn} \end{pmatrix} \bullet \left(x_1^0, x_2^0, \dots, x_n^0\right)^T$$

Y is a vector with n dimensions, and to recover layer P means to extract the matrix A. To decrease the complexity, we could divide the A into several sub-blocks based on the column and divide the X based on the row. We could obtain the following equation:

$$Y = [A_1 \mid A_2 \mid \dots \mid A_k] \bullet \left(X_1^0 \mid X_2^0 \mid \dots \mid X_k^0\right)^T$$

$$Where:\ A_i = \begin{pmatrix} a_{1((i-1)m+1)} \cdots a_{1(im)} \\ \dots\dots\dots \\ a_{n((i-1)m+1)} \cdots a_{n(im)} \end{pmatrix}, X_i^0 = \begin{pmatrix} x^0_{(i-1)m+1} \\ \dots \\ x^0_{im} \end{pmatrix}, i = 1, \dots, k.$$

A_i could be represented as an $n \times m$ matrix with $0/1$ elements, where A could be acquired by solving each A_i. As introduced in 3.1, we could obtain the size of S-box as m. If we set the input of the ith S-box as constant, which is denoted as X_i, others are random, we could acquire several different output vectors Y of layer P through several pairs of input sequences. We define the jth output vector as $Y_{i,j}$ and choose a pair of output $(Y_{i,0}, Y_{i,1})$ to analyze. For a pair of inputs have the same byte as X_i, the corresponding output of S-box should be the same, and the differential of these outputs is 0. We could obtain the following equation:

$$A_i^{-1} \cdot Y_{i,0} \wedge A_i^{-1} \cdot Y_{i,1} = 0 \Leftrightarrow A_i^{-1} \cdot (Y_{i,0} \wedge Y_{i,1}) = 0. \quad (4)$$

A_i^{-1} could be represented as the $n \times m$ matrix with $0/1$ elements, $Y_{i,0} \wedge Y_{i,1}$ is an n-dimensional vector that is named as ΔY_i. With the characters of the matrix rank, as for the matrix A_{m*n} and B_{n*s}, if $A_{m*n} * B_{n*s}$ $(m < n)$, we could obtain $r(A_{m*n}) + r(B_{n*s}) \leqslant n$, where $r(A_{m*n})$,$r(B_{n*s})$ represent the rank of A_{m*n} and B_{n*s}, respectively. If A_{m*n} has full rank, $r(A_{m*n}) = m$ and $r(B_{n*s}) \leqslant n - m$. As a result, we only need to find $(n - m)$ groups of linear independent vectors ΔY_i and could solve the corresponding matrix A_i^{-1}.

To repeat the previous steps, we could obtain each $A_i^{-1}(i = 1, 2, ..., k)$ and acquire the complete matrix A^{-1}. For the layer P is a surjection operation, we could acquire A through A^{-1} and recover the P operation.

We can recover the value of ΔY_i through the POLAR DPA and reverse the layer P by **Algorithm 2**. The pseudocode is given below.

```
Algorithm 2: GetPLayer
Input: $m, k$
Output: $A$

Begin
For( i = 1; $i \leqslant k$; i++)
{
For(j = 1; $j \leqslant (n - m)$; j++)
{
To acquire $\Delta Y_i$ through the POLAR DPA.
}
Solve the linear equations to obtain $A_i^{-1}$;
}
$A^{-1} = \lfloor A_1^{-1} \mid A_2^{-1} \mid \ldots \mid A_k^{-1} \rfloor$, to acquire $A$
return $A$; //$A$ is unique for a specific algorithm
End
```

Analyzing the complexity, it needs $(n-m)$ times of the POLAR DPA to solve A_i^{-1} and to acquire the matrix A after k times computation. The total complexity is $O(k * (n - m))$.

The Reverse Engineering on the parameters of S-box

To choose a pair of input for the ith S-box, named as X_1 and X_2, the corresponding differential output of P could be represented as

$$P(S_i(X_1) \wedge S_i(X_2)) = \delta. \tag{5}$$

Furthermore, we could obtain the differential values of $S_i(X_1)$ and $S_i(X_2)$ and build the following equations through exhaustive search of S-box

$$S_i(X_0) \wedge S_i(X_1) = \delta_1\ ,\ S_i(X_0) \wedge S_i(X_2) = \delta_2\ ,\ \ldots,\ S_i(X_0) \wedge S_i(X_{2^m-1}) = \delta_{2^m-1}. \tag{6}$$

We could exhaust 2^m groups of S-box value and certificate the correct S-box through one time encryption. If every S-box of one layer is distinguished, we should compute k times to recover the parameters of all different S-boxes, and the complexity is $O(k * 2^m)$.

If only the S-box is unknown in the algorithm, with the same idea, we also could achieve the reverse analysis on S-box. To solve Equation (6), we can also use the POLAR DPA to obtain δ_i, and the specific algorithm is given below.

```
Algorithm 3: GetSbox
Input: m,k
Output: S

Begin
For( i = 1; $i \leqslant k$; i++)
{
For(j = 1; $j < 2^m$; j++)
{
Execute the POLAR DPA to obtain $\delta_i$
}
For(j = 0; $j < 2^m$; j++)
{
$S_i(0) = j$, which is substituted into the equations (6) to obtain the parameters of $S_i$
If(the parameters of $S_i$ is correct) continue;
}
}
$S = S_1, S_2, \dots, S_k$;
return $S$;
End
```

One time of reverse analysis needs 2^m times of the POLAR DPA and 2^m times of exhaustive searches, and the complexity to recover all S-boxes in one layer is $O(k * 2^m)$, when there are different S-boxes in one S layer.

By using the aforementioned method, we can first reverse the size of the round function and then reverse the parameters of the inner operations. The attack on algorithm MIBS is given in the Section 4.

4. EXPERIMENTS AND ANALYSIS

To verify the effectiveness of the reverse analysis, we conduct the experiments on MIBS algorithm [26], which is supposed to have the unknown round function. Our experiments are based on the simulation platform, which include Design Compiler, Modelsim, and Synopsys PrimePower.

4.1. *Experiments of structure reverse engineering*

First, we introduce the implementation of MIBS algorithm, whose S-boxes are implemented with LUTs. We use the structure reverse analysis illustrated in 3.1 to acquire the size of S-box.

We attacked the first round function of MIBS with the chosen plaintext analysis, which uses the static key and chooses different plaintexts to control the input bit of the S-box in the first round from high bit to low bit. Complying with the method in 3.1, it needed three groups of dynamic power consumptions to verify different guessed sizes of the S-box. We define $Cbit_i (i = 1, 2, 3)$ to represent the number of changed bits in the ith group of dynamic power consumption. Part of guessed-size schemes of the S-box are listed in Table I.

We use PrimePower to collect power samples. As introduced in 3.1, we obtain the $P'_{\texttt{total}}$ with a pair of the same plaintexts. When we change the plaintexts, we can obtain the the dynamic power consumption by subtracting $P'_{\texttt{total}}$ from the total consumption. The traces of the dynamic power consumption for the aforementioned schemes are illustrated in Figure 5.

For one guessed size of the S-box, it needs three times of power sampling. We use P_{d1}, P_{d2} and P_{d3} to represent the dynamic power of the first , second, and third sampling, respectively. The ratios of the dynamic power are concluded in Table II.

Illustration: Based on Algorithm 1, if the guessed size of the S-box is correct, the values of P_{d2}/P_{d1} and P_{d3}/P_{d1} are close to 2 and 3, respectively. Therefore, we could find the correct size of S-box is 4×4 in Table II. The result is correct according to the algorithm MIBS.

4.2. *Experiments of reverse engineering on inner operations*

The structure of MIBS algorithm is listed as Figure 6.

Table I. Some guessed-size schemes of S-box.

Guessed size of S-box	$Cbits_1$	$Cbits_2$	$Cbits_3$
2×2	1	2	3
4×4	1	3	5
8×8	1	5	9

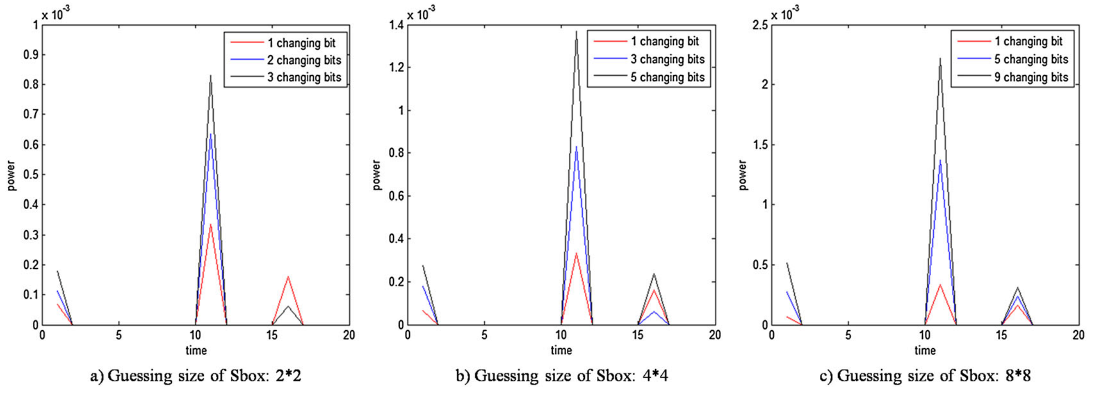

Figure 5. Dynamic power for different guessed sizes of S-box.

Table II. The ratio of dynamic power for different guessed size of S-box.

Guessed size of S-box	P_{d2}/P_{d1}	P_{d3}/P_{d1}	True/False
2×2	1.8250313946954	2.12470380505636	F
4×4	2.12470380505636	3.19299074362408	T
8×8	3.19299074362408	5.08413723302119	F

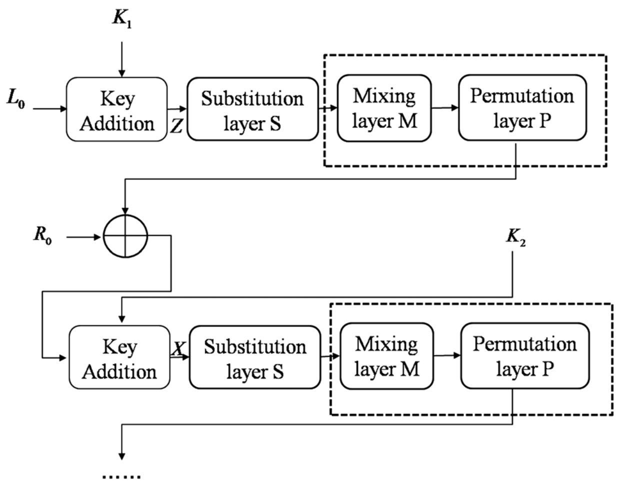

Figure 6. The structure of MIBS algorithm.

The dashed box in Figure 6 illustrates the P layer. The parameters of S and P are both unknown. If the input of the S-box in the second round could be named as X, we could obtain the equation (7).

$$X = P(S(Z)) \wedge R_0 \wedge K_2 \tag{7}$$

- Z: the input of S-box in the first round;
- R_0: the right part of plaintext;
- K_2: the subkey in the second round.

Equation (7) satisfied the policy of the POLAR DPA; we could recover the parameters of P. In the POLAR DPA, we need to do two DPA experiments to obtain one-bit differential value. For each DPA, the attacker could set the key and the left of plaintext L_0 to be static and choose the right of plaintext R_0 randomly. R_0 is the partition part of the POLAR DPA. We have obtained the size of

S-box as 4×4 in Section 4.1. According to **Algorithm 2**, for each DPA, Z is the set as the value of $0x31234567$ and $0x39abcdef$, respectively, where the input of the first S-box is fixed. We could obtain the corresponding DPA traces of the highest bit of Z with 10,000 power traces as illustrated in Figure 7.

From Figure 7, we can see that the polarities of two DPA traces are opposite and the correlation between two curves is -0.985464567, so the differential value should be 1. For each bit, we repeat the aforementioned process and we can obtain the differential value of each bit, which is illustrated in Table III.

According to Table III, for a pair of $Z(0x31234567, 0x39abcdef)$, the differential value after layer S and layer P is $0x9f4676c4$. To verify the differential value, we execute the encryption of MIBS for the pair of Z; the result is shown in Table IV.

We can learn that the differential value of our experiment is correct. As the input of the first S-box have been fixed in Z, thus we could obtain the equation (8).

$$P_1^{-1}(0x9f4676c4) = 0 \tag{8}$$

Repeat the previous steps, we could obtain a series of linear equations. To solve the P_1^{-1}, it is needed 28 groups of linear independent equations. And the other seven pairs of $P_i^{-1}(i = 2, \ldots, 8)$

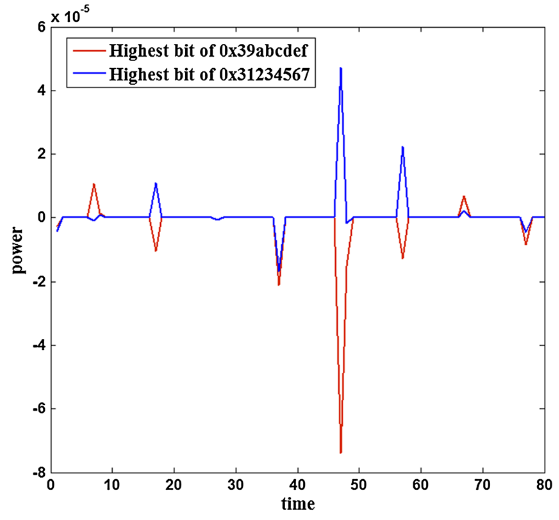

Figure 7. DPA curve of the highest bit.

Table III. The ratio of dynamic power for different guessed size of S-box.

bit0	bit1	bit2	bit3
0.912354785	0.985422113	−0.885453331	0.864855444
bit4	bit5	bit6	bit7
0.875424321	0.889513212	−0.984522313	−0.943232122
bit8	bit9	bit10	bit11
0.874546353	−0.949843211	−0.822896354	0.687654455
bit12	bit13	bit14	bit15
−0.985545242	−0.789512021	−0.854321232	0.921045523
bit16	bit17	bit18	bit19
0.987502313	−0.935468431	−0.821237896	0.785245422
bit20	bit21	bit22	bit23
0.785421138	0.875533221	−0.879464533	0.987135432
bit24	bit25	bit26	bit27
−0.823456861	−0.7122546663	−0.885765423	−0.985465312
bit28	bit29	bit30	bit31
−0.864565689	0978545633	0.894645669	−0.985464567
Differential Value	10011111010001100111011011000100(0x9f4676c4)		

Table IV. The intermediate result and differential value for the first round of MIBS.

Z	0x31234567	0x39abcdef
Output of S-layer	0x8f38dac0	0x857e2619
Output of P-layer	0xa6caef73	0x378c99b7
Differential value	0x9f4676c4	

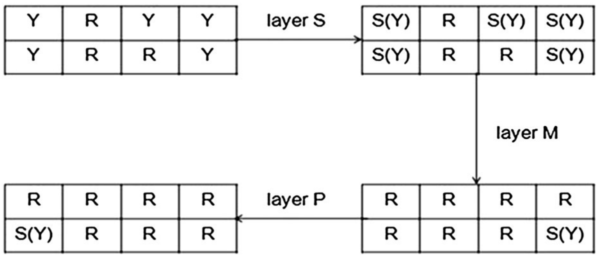

Figure 8. One round of MIBS Algorithm.

Table V. The intermediate result and differential value for the first round of MIBS.

bit3	bit2	bit1	bit0
0.388760987180	0.939511072952	−0.638112502011	0.887657305118

could be also recovered with the same method. We could recover P with P^{-1}, and as the introduction in Section 3.2, we can obtain the differential output of the S-box and build the equations as (6) so as to recover the parameter of the S-box.

When there are only S-boxes unknown, for example MIBS-like algorithm, we can hold the $0th$, $2th$, $3th$, $4th$, and $7th$ bits of Z as nibble Y; other bits are random. The flow of one round is as Figure 8.

According to the MIBS algorithm, we can obtain the following equation.

$$X_{[4]} = S(Y) \wedge R_{0[4]} \wedge K_{2[4]}. \tag{9}$$

- $X_{[4]}$ is the 4th nibble of the second round;
- $R_{0[4]}$ is the 4th nibble of R_0;
- $K_{2[4]}$ is the 4th nibble input of K_2.

The equation (9) satisfied the policy of the POLAR DPA, and we can use **Algorithm 3** to reverse the parameters of S-box. For example, we can set Y as $0x0$ and $0xd$, respectively, and $R_{0[4]}$ will be used as the partition part. We execute DPA twice; therefore, we can obtain the one-bit differential value of $S(0x0) \wedge S(0xd)$.

The correlations of each bit in the differential power curves when $Y = 0x0$ and $Y = 0xd$ are listed in Table V.

From the result of Table V, we could obtain $S(0x0) \wedge S(0xd) = 0x2$, which corresponds the correct differential value for $S(0x0) = 0x4$ and $S(0xd) = 0x6$ as Figure 9. Repeating the previous steps, we could obtain the differential values of the S-box as shown in Table VI.

We could recover the parameters of the S-box after exhaustively searching of $S(0x0)$ and verifying the effectiveness of the parameter by one time encryption. The results proof that the reverse analysis proposed in this paper could recover the unknown round function of the block ciphers. Furthermore, we focus on the general structure of the round function in block cipher, and the new method in this paper can be more universal. By contrast, the first reverse analysis of block ciphers using the SCA approach proposed by Roman Novak other existing SCAREs [14–16] mostly concentrated on the specific operation, such as S-box. The mathematical analysis usually operated with

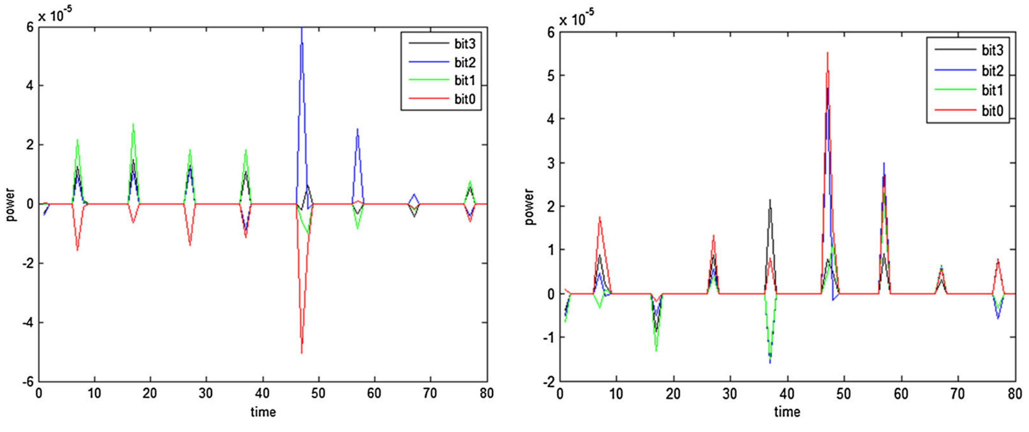

Figure 9. DPA curves for Y = 0x0 and Y = 0xd.

Table VI. The Differential value of S-box in MIBS.

Y	1	2	3	4	5	6	7	8	9	a	b	c	d	e	f
S(0x0) $\wedge$ S(Y)	b	7	c	9	e	8	4	f	1	3	a	6	2	5	d

the reduced round cipher, such as structure analysis proposed by Biryukov and Shamir to recover the SASAS round of the block ciphers with SPN. These structure analyses [17–22] mostly perform a successful attack under the condition that the target cipher is the reduced round cipher. The logic of the linear operations, the sizes of the linear operations, and the non-linear operations should be known by the attackers.

5. CONCLUSION

The unknown parts of a secret block cipher could make a cryptanalysis more difficult. To check if a block cipher meets the security requirement, it is necessary to evaluate it by cryptanalysis especially by the reverse engineering analysis. The existing SCAs for reverse engineering mainly focus on the single unknown non-linear operation. The complexity is very high and might be highly dependent on the leakage model to fulfill the reverse analysis. We proposed a new reverse engineering analysis on the unknown round function of the block ciphers. This method could recover both the structure of the round function and the parameters of inner operations efficiently.

We improve the traditional DPA to build the reverse analysis based on the polarity of power curves, which is named as the POLAR DPA. First, we used the strict avalanche criterion to recover the size of the S-box. Second, we acquired the parameters of nonlinear and linear operations in the round function through combination of the POLAR DPA and the structure analysis. The complexity of the structure reverse analysis is $O(m)$, and the complexity of inner reverse method is $O(k * (n - m))$ for the linear operation and $O(k * 2^m)$ for the non-linear operation, where n is the size of input, m is the size of input/output in the S-box, k is the number of S-boxes, $n = m * k$.

To prove the effectiveness of our reverse analysis, we conduct experiments on MIBS algorithm to recover the unknown rounds. The results of experiments have shown the feasibility and effectiveness of the reverse analysis, which could be regarded as a generic reverse engineering method for most of the block ciphers, and an important evaluation for the secret block ciphers.

ACKNOWLEDGEMENTS

This work was sponsored by National Natural Science Foundation of China (grant numbers 61202386, 60970166, 60970115, 61003267).

REFERENCES

1. Biham E, Shamir A. Differential cryptanalysis of DES-like cryptosystems. *Journal of Cryptology* 1991; **4**:3–72.
2. Biham E, Shamir A. *Differential Cryptanalysis of the Data Encryption Standard*. Springer-Verlag: Berlin Heidelberg, 1993.
3. Courtois NT, Pieprzyk J. Cryptanalysis of block ciphers with overdefined systems of equations. In *ASIACRYPT 2002*, Vol. 2501, LNCS. Springer-Verlag: Berlin Heidelberg, 2002; 267–287.
4. Matsui M. Linear cryptanalysis method for DES cipher. In *EUROCRYPT 1993*, Vol. 765, LNCS. Springer-Verlag: Berlin Heidelberg, 1994; 386–397.
5. Biham E, Shamir A. Differential fault analysis of secret key cryptosystems. In *CRYPT0 1997*, Vol. 1294, LNCS. Springer-Verlag: Berlin Heidelberg, 1997; 513–525.
6. Brier E, Clavier C, Olivier F. Correlation power analysis with a leakage model. In *CHES 2004*, Vol. 3156, LNCS. Springer-Verlag: Berlin Heidelberg, 2004; 16–29.
7. Chari S, Rao JR, Rohatgi P. Template attacks. In *CHES 2002*, Vol. 2523, LNCS. Springer-Verlag: Berlin Heidelberg, 2003; 13–28.
8. Kocher P. Timing attacks on implementations of Diffie–Hellmann, RSA, DSS, and other systems. In *CRYPTO 1996*, Vol. 1109, LNCS. Springer-Verlag: London, 1996; 104–113.
9. Kocher P, Jaffe J, Jun B. Differential power analysis. In *CRYPTO'99*, Vol. 1666, LNCS. Springer-Verlag: Berlin Heidelberg, 1999; 388–397.
10. Quisquater JJ, Samyde D. Electromagnetic analysis (EMA): measures and counter-measures for smart cards. In *ACM 2001*, Vol. 2140, LNCS. Springer-Verlag: Berlin Heidelberg, 2001; 200–210.
11. (Available from: http://www.schneier.com/paper-twofish-paper.html) [Accessed on June 1998].
12. (Available from: http://www.encryptfiles.net/encryption/algorithm/mars.php) [Accessed on August 1999].
13. (Available from: http://web.archive.org/web/20090318143444/http://www.scard.org/gsm/a3a8.txt). 1998.
14. Biryukov A, Shamir A. Structural cryptanalysis of SASAS. In *EUROCRYPT 2001*, Vol. 2045, LNCS. Springer-Verlag: Berlin Heidelberg, 2001; 394–405.
15. Borgho J, Knudsen LR, Leander G, Thomsen S. Cryptoanalysis of present-like cipher with secret S-boxes. In *FSE 2011*, Vol. 6733, LNCS. Springer-Verlag: Berlin Heidelberg, 2011; 270–289.
16. Borghoff J, Knudsen LR, Leander G, Matusiewicz K. Cryptanalysis of c2. In *CRYPTO 2009*, Vol. 5677, LNCS. Springer-Verlag: Berlin Heidelberg, 2009; 250–266.
17. Clavier C. An improved scare cryptanalysis against a secret A3/A8 GSM algorithm. In *ICISS 2007*, Vol. 4812, LNCS. Springer-Verlag: Berlin Heidelberg, 2007; 143–155.
18. Daudigny R, Ledig H, Muller F, Valette F. SCARE of the DES. In *ACNS 2005*, Vol. 3531, LNCS. Springer-Verlag: Berlin Heidelberg, 2005; 393–406.
19. Guilley S, Sauvage L, Micolod J. Defeating any secret cryptography with SCARE attacks. In *LATINCRYPT 2010*, Vol. 6212, LNCS. Springer-Verlag: Berlin Heidelberg, 2010; 273–293.
20. Novak R. Side-channel attack on substitution blocks. In *ACNS 2003*, Vol. 2846, LNCS. Springer-Verlag: Berlin Heidelberg, 2003; 307–318.
21. Novak R. Side-channel based reverse engineering of secret algorithms. In *ERK 2003*. IEEE: Ljubljana, Slovenia, 2003; 445–448.
22. Réal D, Dubois V, Guilloux A-M, Valette F, Drissi M. SCARE of an unknown hardware feistel implementation. In *CARDIS 2008*, Vol. 5189, LNCS. Springer-Verlag: Berlin Heidelberg, 2008; 218–227.
23. Biryukov A. Structural cryptanalysis of SASAS. *Journal of Cryptology* 2010; **23**:505–518.
24. Gierlichs B, Batina L, Tuyls P, Preneel B. Mutual information analysis. In *CHES 2008*, Vol. 5154, LNCS. Springer-Verlag: Berlin Heidelberg, 2008; 426–442.
25. Webster AF, Tavares SE. On the design of S-boxes. In *CRYPTO 1985*, Vol. 218, LNCS. Springer-Verlag: Berlin Heidelberg, 1986; 523–534.
26. Izadi M, Sadeghiyan B, Sadeghian SS, Khanooki HA. MIBS: A new lightweight block cipher. In *CANS 2009*, Vol. 5888, LNCS. Springer-Verlag: Berlin Heidelberg, 2009; 334–348.