On the Key Leakage from Linear Transformations

Seungkwang Lee, Nam-su Jho and Myungchul Kim

Information Security Research Division, ETRI
skwang@etri.re.kr

Abstract. Linear transformations are often applied to the table-based cryptographic implementation including white-box cryptography in order to prevent key-dependent intermediate values from being analyzed. However, it has been shown that there still exists a correlation before and after the linear transformations, and thus this is not enough to protect the key against gray-box attacks such as power analysis. So far, the Hamming weight of rows in the invertible matrix has been considered the main cause of the key leakage from the linear transformation. In this study, we provide an in-depth analysis of the cause with the characteristics of block invertible binary matrices and the distribution of intermediate values. Our experimental results and proof show that the balanced distribution of the key-dependent intermediate value is the main cause of key leakage.

Keywords: Power analysis, linear transformations, key leakage.

1 Introduction

From a secret key point of view, a block cipher can be seen as a secret bijection between a plaintext set and a ciphertext set. One of the easy ways to implement this bijection is a lookup table of mapping a plaintext to its corresponding ciphertext. Since implementing a block cipher as one lookup table is impractical because of its huge size, it is usually implemented as a series of lookup tables like in the case of white-box cryptography. The important point over here is that white-box cryptography generates key-instantiated lookup tables and protects each table with linear and nonlinear transformations in order to prevent a key leakage from lookup values.

There are various techniques to extract the key hidden in white-box cryptographic implementations of standard block ciphers such as DES and AES. First, a number of practical cryptanalysis techniques [4, 13, 20, 25–27, 38] on the white-box DES (WB-DES) and AES (WB-AES) and their variants [7, 16, 19, 21, 40] have been introduced. Second, Differential Fault Analysis (DFA) [32] on white-box cryptography was also demonstrated, where an attacker is able to inject a fault at a desired location in memory. Here, those white-box attacks rely on an in-depth understanding of a target implementation so that an attacker is able to access precise internal states during the execution. Thus commercial white-box cryptography [3, 11, 14, 37] focuses on making a barrier to the full control of an attacker and is often combined with additional protection techniques including obfuscation, enveloping, hardware ID binding, and anti-debug protections.

In contrast to above white-box attacks, gray-box attacks using non-invasive information such as power consumption of a target device can be mounted. Differential Power Analysis (DPA) [18], one of the most well-known techniques to reveal the secret key imbedded in IC cards, is based on the fact that power consumption of a device is proportional or inversely proportional to the Hamming weight (HW) of data it processes. In detail, a power analysis attacker collects a number of power traces with random plaintexts and finds a correct key that computes hypothetical values most highly correlated to the collected traces at a particular point. We focus on the fact that white-box cryptography can be easily broken by power analysis [5, 33] without detailed knowledge of the target implementation. This means linear and nonlinear transformations applied to lookup tables have no effect on hiding key-sensitive intermediate values. In case of linear transformations, it was recently reported in [2, 31] that if the invertible matrix used for the linear transformation has at least one row of HW 1 then power analysis will succeed with overwhelming probability. Otherwise, the correct key is expected to be indistinguishable from the wrong key hypothesis correlation and power analysis fails with high probability.

In this study, our analysis explains that the key leakage from the linear transformation is likely to happen even if there is no such row of HW 1 in the matrix. We also show that there will be no such matrix containing any row of HW 1 if a block invertible binary matrix with 4×4 submatrices of full rank is chosen for carrying maximum information and maximizing information diffusion as recommended in [9]. In addition, a large invertible matrix can be efficiently generated using the technique introduced in [39]. Importantly, we find out that the key leakage after linear transformations is largely due to the balanced distribution of intermediate values, and we offer a demonstration and a simple proof using the Walsh transforms. To enhance our finding, we try to insert a random byte in the intermediate value before linear transformations making an unbalanced distribution and show a reduced correlation to the key.

The rest of this paper is organized as follows. Section 2 reviews some basic concepts including power analysis and revisits the key leakage issue on the linear transformation with the Walsh transforms. In Section 3, we provide our analysis of the main reason why the key is still revealed in the presence of linear transformation. Section 4 concludes this paper.

2 Background

In this section, we introduce the basic concept of power analysis and we use the Walsh transforms for demonstrating the key leakage in the presence of linear and nonlinear transformations.

2.1 Power Analysis

An explanation of successful power analysis on the white-box cryptographic implementation could be that attacker's correct hypothetical value will correlate to the target lookup value. Here, DPA or Correlation Power Analysis (CPA) [6] can be used as power analysis techniques. Note that Differential Computation Analysis [5] improves the efficiency of DPA and CPA attacks since there is no measurement noise in the software execution traces, unlike the power consumption traces.

After collecting the traces with random plaintexts, DPA and CPA perform statistical analysis in different ways. DPA uses the selection function D to split the collected traces into sets based on the attacker's hypothetical values. If the attacker's hypothetical key is correct (and therefore the hypothetical value is correct), then the trace separation by D is also accurate and there will be a peak in the differential trace.

In contrast, CPA uses a leakage model including the HW and the Hamming distance instead of the selection function D. When attacking a white-box implementation, the bit (mono-bit) model is appropriate because HW-based CPA attacks are unlikely to be successful due to the disturbed HW by linear and non-linear transformations. Given N power traces $V_{1..N}[1..\kappa]$ containing κ samples each, CPA will estimate the power consumption at each point of each trace using attacker's hypothetical intermediate value. For K different key candidates, let $\mathcal{E}_{n,k^*}$ $(1 \leq n \leq N,\ 0 \leq k^* < K)$ denote the power estimate in the n^{th} trace with the hypothetical key k^*. To measure a correlation between hypothetical power consumption and measured power traces, the estimator r is defined as follows [22]:

$$r_{k^*,j} = \frac{\sum_{n=1}^{N}(\mathcal{E}_{n,k^*} - \overline{\mathcal{E}_k^*}) \cdot (V_n[j] - \overline{V[j]})}{\sqrt{\sum_{n=1}^{N}(\mathcal{E}_{n,k^*} - \overline{\mathcal{E}_k^*})^2 \cdot \sum_{n=1}^{N}(V_n[j] - \overline{V[j]})^2}},$$

where $\overline{\mathcal{E}_k^*}$ and $\overline{V[j]}$ are sample means of $\mathcal{E}_k^*$ and $V[j]$, respectively. If there exists a correlation, a noticeable peak will be found in the correlation plot for the correct key.

Power analysis countermeasures can be categorized into masking and hiding, where masking breaks the correlation between power signals and the processed data while hiding reduces the signal to noise ratio. Maksing [1, 10, 12, 23, 28, 35] randomizes every key-dependent intermediate value by precomputing a new masked lookup table for each execution of encryption. To protect against higher-order DPA attacks [15, 24, 36], where an attacker exploits the joint key leakage from several intermediate values, higher-order DPA countermeasures have been studied [8, 17, 29, 30, 34]. One of the most used hiding techniques, on the other hand, is to induce random delay. When the target cryptographic operation occurs uniformly distributed across n time instants due to random delay, the number of power traces for a successful DPA grows in n^2 only if DPA is performed straightforwardly. Here we can see these countermeasures are strongly dependent on expensive run-time random source, and also result in slow execution of cryptographic algorithm.

2.2 Detecting Key Leakage by the Walsh Transforms

Given a table-based implementation of a block cipher which is protected by linear and nonlinear transformations (often we use the term *encoding*), we can detect the key leakage using the Walsh transforms. To understand how the Walsh transform can be used to quantify a correlation we use the following definitions from [33].

Definition 1. *Let* $x = \langle x_1, \ldots, x_n \rangle$*,* $\omega = \langle \omega_1, \ldots, \omega_n \rangle$ *be elements of* $\{0,1\}^n$ *and* $x \cdot \omega = x_1\omega_1 \oplus \ldots \oplus x_n\omega_n$*. Let* $f(x)$ *be a Boolean function of* n *variables. Then the Walsh transform of the function* $f(x)$ *is a real valued function over* $\{0,1\}^n$ *that can be defined as* $W_f(\omega) = \Sigma_{x \in \{0,1\}^n} (-1)^{f(x) \oplus x \cdot \omega}$*.*

Definition 2. *Iff the Walsh transform* W_f *of a Boolean function* $f(x_1, \ldots, x_n)$ *satisfies* $W_f(\omega) = 0$*, for* $0 \leq HW(\omega) \leq m$*, it is called a balanced* m^{th} *order correlation immune function or an* m*-resilient function.*

Then we know that $W_f(\omega)$ quantifies the imbalances in the encoding, and the large absolute value of $W_f(\omega)$ means the strong correlation between $f(x)$ and $x \cdot \omega$. By utilizing this property, we calculate the correlation between the table lookup values and hypothetical values.

Let's demonstrate the key leakage from the encoded lookup table generated by the composition of SubBytes and AddRoundKey in the first round of AES with a 128-bit key. We denote the initial round key by k $(= k_0 k_1 \ldots k_{15})$, and decompose the Mixcolumns operation with a column vector $[x_0\ x_1\ x_2\ x_3]^T$ of the state matrix as follows:

$$\begin{aligned}
&\begin{bmatrix} 02\ 03\ 01\ 01 \\ 01\ 02\ 03\ 01 \\ 01\ 01\ 02\ 03 \\ 03\ 01\ 01\ 02 \end{bmatrix} \begin{bmatrix} x_0 \\ x_1 \\ x_2 \\ x_3 \end{bmatrix} \\
&= x_0 \begin{bmatrix} 02 \\ 01 \\ 01 \\ 03 \end{bmatrix} \oplus x_1 \begin{bmatrix} 03 \\ 02 \\ 01 \\ 01 \end{bmatrix} \oplus x_2 \begin{bmatrix} 01 \\ 03 \\ 02 \\ 01 \end{bmatrix} \oplus x_3 \begin{bmatrix} 01 \\ 01 \\ 03 \\ 02 \end{bmatrix} \\
&= x_0 \cdot MC_0 \oplus x_1 \cdot MC_1 \oplus x_2 \cdot MC_2 \oplus x_3 \cdot MC_3,
\end{aligned}$$

where MC_i is the i^{th} column vector of the MixColumns matrix, and $y_i(x_i) = x_i \cdot MC_i$. Now we let

$$\begin{aligned}
x \quad &= S(p \oplus k_0) \\
y_0(x) &= \left[2 \cdot x\ x\ x\ 3 \cdot x \right]^T
\end{aligned}$$

where $p \in \mathrm{GF}(2^8)$ means the first subbyte of the plaintext in the state matrix, S represents SubBytes. Let $f(x)$ denote the lookup values of $y(x)$ encoded by linear and nonlinear transformations. We have 32 Boolean functions $f_{i\in\{1,\ldots,32\}}(x)$: $\{0,1\}^8 \rightarrow \{0,1\}$. To recover the target subkey $k_0 = \mathit{0x}88$, we calculate the Walsh transforms W_{f_i} and sum all the imbalances for each key candidate and ω such that $\mathrm{HW}(\omega) = 1$ as follows:

$$\Delta^{f}_{k\in\{0,1\}^8} = \sum_{\omega=1,2,4,\ldots,128} \sum_{i=1,\ldots,32} |W_{f_i}(\omega)|.$$

The reason why we only select ω of $\mathrm{HW}(\omega) = 1$ is that the HW-based key leakage model is not effective to detect the correlation before and after the both transformations.

The Walsh transforms and their sum of all imbalances are plotted in Fig. 1. As we can see in Fig. 1a, the Walsh transforms with $\omega = 4$ of the correct key ($\mathit{0x}88$) produce 0 except two points; the W_{f14} and W_{f16} of the correct key are -128, and their absolute value (128) is the most highest value. In contrast, the maximum and the average values of $|W_{f_i}(\omega)|$ of wrong key candidates are 56 and about 13.13 (the standard deviation is about 9.35), respectively. This gives us that $f_{14}(\cdot)$ and $f_{16}(\cdot)$ cause key leakages and thus power analysis using the 3^{rd} bit (the LSB is the 1^{st} bit) of attacker's hypothetical SubBytes output can be successful. Hereafter, we will utilize the Walsh transforms for various purposes including the calculation of correlation and our proof regarding the cause of key leakage.

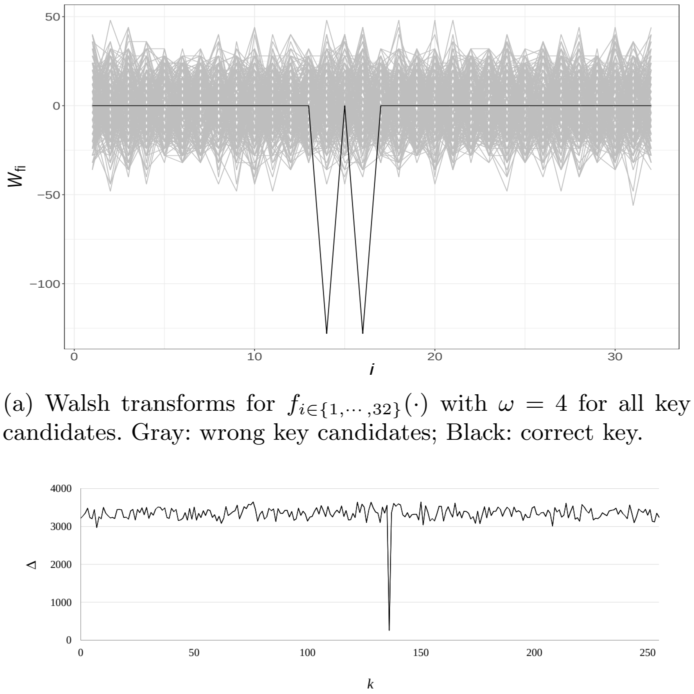
3 Analysis of Linear Transformations

As mentioned, previous studies [2, 31] on linear transformations point out that the existence of rows of HW 1 in the invertible matrix decides the key leakage. In addition, it is reportedly possible to recover the key in the presence of a matrix without identity row by calculating all possible linear combinations (2^8 times) of the bits in the target intermediate value [2]. However, we note that a 32×32 linear transformation is applied to the SubBytes output multiplied with MC_i in the typical WB-AES implementation [9], instead of applying an 8×8 linear transformation to the SubBytes output. Therefore, it becomes very complex, unlike their analysis, to carry out an attack on all possible combinations. In this section, we show that the main cause of key leakage lies in the distribution of the intermediate values rather than some characteristic of the matrix.

3.1 Key Leakage from Linear Transformations

To explain the key leakage from the linear transformation we now let:

$$\begin{aligned} x \quad &= S(p \oplus k_i) \\ f^i(x) &= M \cdot y_i(x)_{i\in\{0,1,2,3\}}, \end{aligned}$$



(a) Walsh transforms for $f_{i\in\{1,\cdots,32\}}(\cdot)$ with ω = 4 for all key candidates. Gray: wrong key candidates; Black: correct key.

(b) Sum of all imbalances for all key candidates. $\Delta^{f}_{k=0x88}$ = 256 (= |−128| + |−128|).

Fig. 1: Key leakage detection using the Walsh transforms.

where $p \in \mathrm{GF}(2^8)$, M is a 32×32 invertible binary matrix, and k_0 = $0x88$, k_1 = $0x99$, k_2 = $0xAA$ and k_3 = $0x$BB. In other words, f^i linearly transforms the SubBytes output x multiplied with MC_i, where x is connected to k_i. For f^i_j given to an attacker, $0 \leq i \leq 3$ and $1 \leq j \leq 32$, a mono-bit power analysis based on the SubBytes output in the first round can be simulated by the Walsh transforms, and in this case we compute 1,024 (= 4 × 32 × 8) Walsh transforms to recover the four subkeys. For simplicity, we abuse the notation by skipping the superscript of f^i by letting $f = f^i$ and use the subscript to f indicating its Boolean functions. Using the Walsh transforms, we can observe how the key leakage appears with respect to the four subkeys in the linear transformations by M. Importantly, we choose M as a block invertible binary matrix for the following reasons, as already explained. First, it is recommended by the author of [9] to choose a non-singular matrix with 4×4 submatrices of full rank, ensuring that the encoded components will carry maximum information and maximizing information diffusion. Second, a large block invertible matrix can be efficiently generated by using the technique explained in [39]. In the case of block invertible matrices, there will be no such row of HW 1. However, the Walsh transforms plotted in Fig. 2 show that there still exists a problematic probability of key leakage from linear transformations. Unlike in the case of Fig. 1b of a key leakage from the linear and nonlinear transformations, note that this shows the key leakage from linear transformations without nonlinear transformations. As a result, we can see that linear transformations with M hide three subkeys k_0, k_2 and k_3 (the Walsh transforms score 0), but expose $k_1(\mathit{0x}99)$ from $y_1(x)$ (the Walsh transforms score 256 in Fig. 2b). This gives us that linear transformations produce well-balanced outputs with an overwhelming probability, but this is not always guarantee a reliable protection on secret keys.

We repeated the above experiment using 1,000 randomly generated block invertible matrices. For $\mathrm{HW}(\omega) = 1$, we have $W_{f_i}(\omega) = 0$ with approximately 99.7% and $W_{f_i}(\omega) = 256$ with 0.3%; the average of $|W_{f_i}(\omega)|$ is approximately 0.7 as shown in Table 1 (we will prove later that $W_{f_i}(\omega) = 0$ or 256). The crucial observation here is that such a small probability of $W_{f_i}(\omega) = 256$ results in the key leakage. To be specific, 1,024 Walsh transforms probably produce 3 peaks of the correct key distinguishable from wrong key candidates, and the 3 peaks can reveal 1 to 3 subkeys. Fig. 3 shows our experimental result that 1 to 3 out of four subkeys are exposed in most cases. Only 51 of 1,000 matrices did not leak any of the four subkeys. Consequently, a linear transformation cannot guarantee the protection of key in white-box cryptography. From now on, we are going to analyze the problematic characteristic of the linear transformations producing W_{f_i} scores of 0 or 256. We first take a look at the block invertible matrix for this matter.

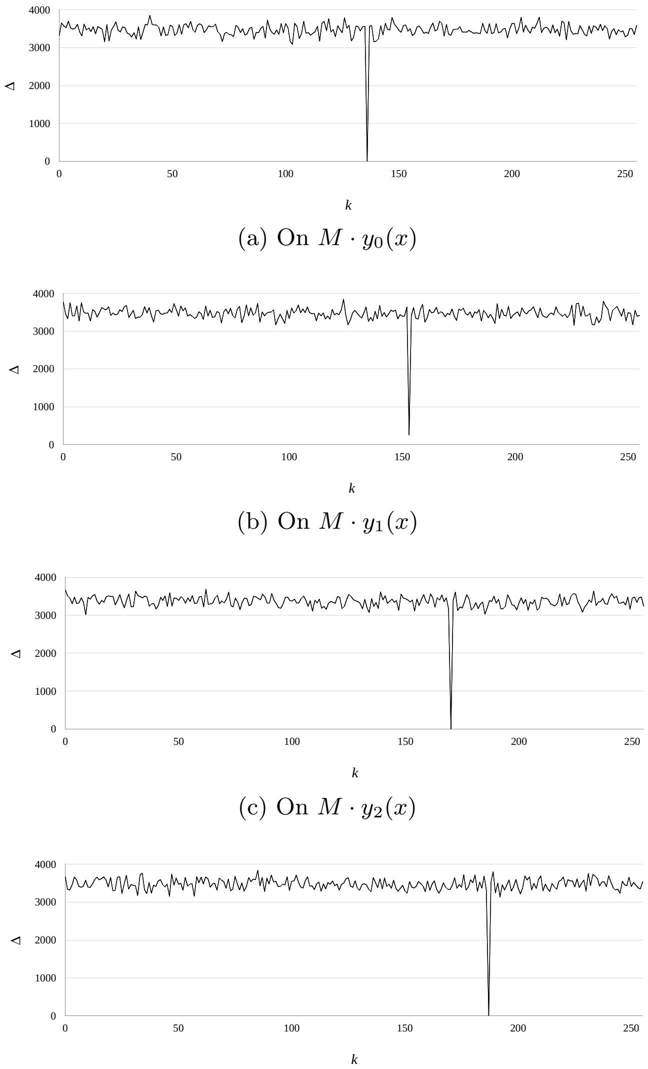
		To be linearly transformed			
		y_0	y_1	y_2	y_3
Number of	$W_{f\,i}(\omega) = 0$	255,206	255,205	255,309	255,203
	$W_{f\,i}(\omega) = 256$	794	795	691	797

Table 1: Statistic of $W_{f\,i}$ scores calculated with 1,000 randomly generated block invertible matrices.

3.2 Analysis of Block Invertible Square Matrix

In [9], the authors choose M as a non-singular matrix with submatrices of full rank for maximizing information diffusion. To begin with, we briefly review the definition of a block invertible square matrix [39].

Definition 3. *If all the blocks $B_{i,j}$ in a block matrix ${}^{n}_{m}M[{}^{p}B]$ are invertible, matrix M is called an (m, n, p) block invertible matrix. Furthermore, if $m = n$, and M is invertible then M is called an (m, p) block invertible square matrix, where ${}^{n}_{m}M[{}^{p}B]$ denotes an $n \times m$ matrix M with nm/p^2 blocks (submatrices), and $B_{i,j}$ denotes the block in row i and column j of blocks.*



(a) On $M \cdot y_0(x)$

(b) On $M \cdot y_1(x)$

(c) On $M \cdot y_2(x)$

(d) On $M \cdot y_3(x)$

Fig. 2: Sum of the imbalance of $W_{f_i}(\omega)$ for all subkey candidates on linearly transformed $y_{i\in\{0,1,2,3\}}(x)$.

Generating $(n,\ 2)$ block invertible square matrices begins with a (2, 2) block invertible square matrix and extends by (4, 2), (6, 2), $\ldots$, and repeats it $(n-2)/2$ times. The important point over here is that every 2×2 submatrix in a $(n,\ 2)$ block invertible square matrix should be invertible by the definition and all 2×2

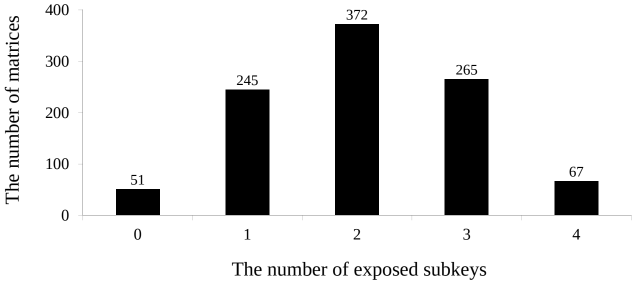


Fig. 3: The number of block invertible matrices (y-axis) vs. the number of exposed subkeys from $M \cdot y_{i\in\{0,1,2,3\}}$ for each block invertible matrix M (x-axis).

invertible matrices in GF(2) are as follows:

$$\begin{vmatrix}1 & 0\\0 & 1\end{vmatrix} \quad \begin{vmatrix}1 & 1\\1 & 0\end{vmatrix} \quad \begin{vmatrix}0 & 1\\1 & 1\end{vmatrix} \quad \begin{vmatrix}0 & 1\\1 & 0\end{vmatrix} \quad \begin{vmatrix}1 & 1\\0 & 1\end{vmatrix} \quad \begin{vmatrix}1 & 0\\1 & 1\end{vmatrix}$$

At a glance, the number of 1s in the 4 out of 6 matrices is greater than 0s. By the principle of constructing a block invertible square matrix, the HW of each row and column in an (n, 2) block invertible matrix will be greater than $n/2$. For example, let's assume that a (4, 2) matrix is initialized with

$$\begin{vmatrix}1 & 0\\0 & 1\end{vmatrix},$$

then its resulting matrix will be

$$\begin{vmatrix}1 & 0 & 1 & 0\\0 & 1 & 0 & 1\\1 & 0 & 1 & 1\\0 & 1 & 1 & 0\end{vmatrix}$$

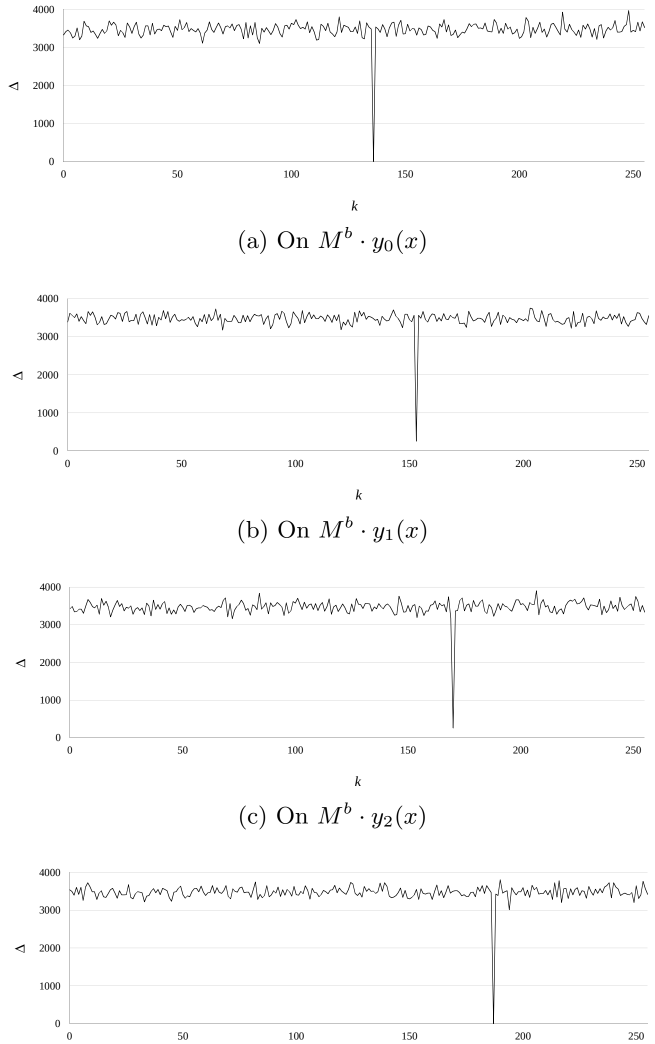
if the technique in [39] is used. Another case of an initialization with

$$\begin{vmatrix}0 & 1\\1 & 0\end{vmatrix}$$

will produce

$$\begin{vmatrix}0 & 1 & 0 & 1\\1 & 0 & 1 & 0\\0 & 1 & 1 & 0\\1 & 0 & 1 & 1\end{vmatrix}.$$

During the generation of a (32, 2) matrix through this process, 1s appear more frequently. We performed the following experiment to check if this overweight HW of the block invertible matrix is the main reason for key leakage. We randomly generated a balanced *non-invertible* (singular) 32×32 matrix M^b, such that $f(x) = M^b \cdot y_{i \in \{0,1,2,3\}}(x)$, where M^b has the HW of 16 for each row and column, and used it to compute the sum of imbalances. As shown in Fig. 4, there still exist key leakages from y_1 and y_2 with the Walsh transform score 256. For this reason, we can conclude that the heavy HW of matrices is not the cause of key leakages from linear transformations.



(a) On $M^b \cdot y_0(x)$

(b) On $M^b \cdot y_1(x)$

(c) On $M^b \cdot y_2(x)$

(d) On $M^b \cdot y_3(x)$

Fig. 4: Sum of the imbalance for all key candidates on each $y_{i \in \{0,1,2,3\}}(x)$ multiplied with a balanced matrix M^b.

3.3 Analysis of Key-dependent Intermediate Values

The next key-leakage point to be analyzed is y. From Definition 1 and 2, we know that a balanced correlation immune function is strongly dependent on the distribution of $f_i(x) \oplus x \cdot \omega$. Here recall that given a key-dependent value $x \in \mathrm{GF}(2^8)$ and 1,000 randomly generated block invertible matrices M, $W_{f_i}(\omega) = 0$ with approximately 99.7% while only 0.3% of $W_{f_i}(\omega) = 256$, where $\mathrm{HW}(\omega) = 1$. The following proof explains why the distribution of y leads to $W_{f_i}(\omega) = 256$ causing key leakages.

Lemma 1. *Assume that a 256×8 binary matrix* $\mathbf{H}$ *is defined as*

$$\mathbf{H} = \begin{bmatrix} h_{1,1} & h_{1,2} & \dots \\ \vdots & \ddots & \\ h_{256,1} & & h_{256,8} \end{bmatrix}$$

where i^{th} *row vector* $\mathbf{h}_{i,*} = \langle h_{i,1}, h_{i,2}, \dots, h_{i,8} \rangle$ *is an element of* $GF(2^8)$ *and* $\mathbf{h}_{i,*} \neq \mathbf{h}_{j,*}$ *for all* $i \neq j$*. Then the HW of XORs of arbitrary chosen column vectors from* H *is 0 or 128. In other words,* $HW(\mathbf{h}_{*,j_1} \oplus \mathbf{h}_{*,j_2} \oplus \cdots \oplus \mathbf{h}_{*,j_n}) =$ *0 or 128, where* n *is a random positive integer and* $j_i \in \{1, 2, \dots, 8\}$*.*

Proof : Let $\mathcal{J}$ be a set of randomly chosen indices from $\{1, 2, \dots, 8\}$. Note that for any duplicated indices α and α' in $\mathcal{J}$, i.e. $\alpha = \alpha'$, removing the duplicated indices from $\mathcal{J}$ makes no change to the result HW.

$$\oplus_{j \in \mathcal{J}} \mathbf{h}_{*,j} = \big(\oplus_{j \in \mathcal{J} - \{\alpha, \alpha'\}} \mathbf{h}_{*,j} \big) \oplus \mathbf{h}_{*,\alpha} \oplus \mathbf{h}_{*,\alpha'}$$

$$= \big(\oplus_{j \in \mathcal{J} - \{\alpha, \alpha'\}} \mathbf{h}_{*,j} \big) \oplus \mathbf{0} = \oplus_{j \in \mathcal{J} - \{\alpha, \alpha'\}} \mathbf{h}_{*,j}.$$

Therefore without loss of generality we can assume that $\mathcal{J}$ contains no duplicated indices and moreover $\big|\mathcal{J}\big| = n \leq 8$.

Now we can define following partitions of indices:

$$\mathcal{I}_{b_1, b_2, \dots, b_n} = \{\ell \in \mathcal{I} | h_{\ell, j_i} = b_i \text{ for all } j_i \in \mathcal{J}\},$$

where $\mathcal{I} = \{1, 2, \dots, 256\}$, and $b_i \in \{0, 1\}$. Here all $\mathcal{I}_{b_1, b_2, \dots, b_n}$ are disjoint to the others and $\cup \mathcal{I}_{b_1, b_2, \dots, b_n} = \mathcal{I}$. To complete the proof, we need that for any choice of b_i's, $\big|\mathcal{I}_{b_1, b_2, \dots, b_n}\big| = 256/2^n = 2^{8-n}$. This can be shown easily as followings. Suppose that $\big|\mathcal{I}_{b_1, b_2, \dots, b_n}\big| = t > 2^{8-n}$. It means that there are t row vectors in $\mathbf{H}$ satisfying the condition ${j_i}^{th}$ bit of the vector equals to b_i. In other words, n bits are determined by choice of b_i's and only $8 - n$ bits are remained free. From the condition of t is larger than 2^{8-n} and the pigeon hole principle in mathematics, there must exist at least two indices ℓ and ℓ' in $\mathcal{I}_{b_1, b_2, \dots, b_n}$, where all bits of $\mathbf{h}_{\ell,*}$ are completely same to the bits of $\mathbf{h}_{\ell',*}$. It contradicts to the assumption $\mathbf{h}_{i,*} \neq \mathbf{h}_{j,*}$ for any $i \neq j$.

From the definition of HW, we can deduce $HW(\oplus_{j\in\mathcal{J}}\mathbf{h}_{*,j})$ is summation of $\left|\mathcal{I}_{b_1,b_2,\ldots,b_n}\right|$ where $\oplus_{i=1,\ldots,n}b_i = 1$.

$$HW(\oplus_{j\in\mathcal{J}}\mathbf{h}_{*,j}) = \varSigma_{\oplus_{i=1,\ldots,n}b_i=1}\left|\mathcal{I}_{b_1,b_2,\ldots,b_n}\right|$$

$$= \varSigma_{\oplus_{i=1,\ldots,n}b_i=1}2^{8-n} = \varSigma_{2^{n-1}}2^{8-n}$$

$$= 2^{n-1}\cdot 2^{8-n} = 2^7 = 128.$$

Note that if $\mathcal{J}$ is empty after de-duplication then the final HW becomes 0. It concludes the proof of lemma.

Note that $W_{f_i}(w)$ is defined as $\sum_{x\in GF(2^8)}(-1)^{f_i(x)\oplus w\cdot x} = \sum_{x\in\{0,1\}^8}(-1)^{M_{i,*}\cdot y(x)\oplus w\cdot x}$, where $M_{i,*}$ is i^{th} row of the matrix M and $y(x)$ is one of $y_0(x)$ - $y_3(x)$ depending on the target subkey. For convenience, let $y(x) = y_0(x)$, a 32×1 matrix $[2\cdot x\;\;x\;\;x\;\;3\cdot x]^T$. If we define $\mathbf{Y}(x)$ as a 32×256 matrix $[2\cdot\mathbf{H}\;\;\mathbf{H}\;\;\mathbf{H}\;\;3\cdot\mathbf{H}]^T$, where the $\mathbf{H}$ is the matrix defined at the Lemma 1, it is easy to show that each column vector of $2\cdot\mathbf{H}$ or $3\cdot\mathbf{H}$ can be defined with XORs of some column vectors of $\mathbf{H}$ based on the property of $GF(2^8)$. Then the above equation can be re-written as

$$\sum_{j=\{1,2,\ldots 256\}}(-1)^{B_j(M_{i,*}\cdot\mathbf{Y}(x)\oplus(w\cdot\mathbf{H}^T))},$$

where $B_j(v)$ means the j^{th} bit of the vector v. Since the exponents of the equation can have only two values 0 or 1, the summation over $\{1,2,\ldots,256\}$ can be re-written with the number of exponents which are 1.

$$W_{f_i}(w) = 256 - (2\times HW(M_{i,*}\cdot\mathbf{Y}(x)\oplus(w\cdot\mathbf{H}^T)))$$

Note that all row vectors of the matrix $\mathbf{Y}(x)$ is represented by XORing of column vectors of $\mathbf{H}$. Therefore $M_{i,*}\cdot\mathbf{Y}(x)\oplus(w\cdot\mathbf{H}^T)$ can be also represented by XORing of column vectors of $\mathbf{H}$. From the lemma 1, it deduces that the HW of $M_{i,*}\cdot\mathbf{Y}(x)\oplus(w\cdot\mathbf{H}^T)$ is 0 or 128. Finally, $W_{f_i}(w) = 256 - (2\times HW(M_{i,*}\cdot\mathbf{Y}(x)\oplus(w\cdot\mathbf{H}^T)))$ becomes 256 or 0. What is remarkable point over here is that the probability of $W_{f_i}(w) = 256$ is very small but not zero. Specifically, it happens when all column indices of $\mathbf{H}$ are canceled each other when the summation is computed with the randomly chosen matrix M.

As mentioned already, our experiment showed that $W_{f_i}(w) = 256$ with approximately 0.3% in the calculation with the correct key, while the wrong key candidates produced $|W_{f_i}(\omega)| = 56$ at maximum and 13.13 in average. For this reason, 1,024 calculations of $W_{f_i}(w)$ with respect to four subkeys given a matrix M are likely to cause key leakages with overwhelming probability. Based on these findings, we perform the following experiments to check if the unbalanced distribution of the intermediate values can reduce the correlation in the linear transformation.

3.4 Inserting A Random Byte in the Intermediate Values

To demonstrate the effect of unbalanced distribution of intermediate values, we insert random bytes in the intermediate values before linear transformations. We begin with an analysis of the inserting position.We will insert a random byte at a particular position in the four-byte intermediate value $y_{i\in\{0,1,2,3\}}(x)$ and then perform a linear transformation with a 40×40 block invertible matrix M^* to check if any key leakage occurs. Among the five inserting positions ρ_1 - ρ_5 of y_0, for example,

$$\left[\rho_1 \; 2\cdot x \; \rho_2 \; x \; \rho_3 \; x \; \rho_4 \; 3\cdot x \; \rho_5\right]^T$$

we select ρ_i, where $i \in [1, 5]$, and then insert different $\gamma \in_R \mathrm{GF}(2^8)$ at ρ_i for each $x \in \mathrm{GF}(2^8)$. Let $y_0^*(x)$ denote $y_0(x)$ after the random byte insertion, and let $f^*(x) = M^* \cdot y_0^*(x)$. Then we can define the Walsh transforms with respect to f^*:

$$W_{f_i^*}(\omega) = \varSigma_{x\in\{0,1\}^8}(-1)^{f_i^*(x)\oplus x\cdot\omega}$$

for 40 Boolean functions

$$f^*_{i\in\{1,\ldots,40\}}(x) : \{0,1\}^8 \to \{0,1\}.$$

With 1,000 randomly generated M^*, we computed $W_{f_i^*}(\omega)$. As a result, Table 2 gives us that the correct subkey results in $W_{f_i^*}(\omega) = 0$ with approximately 5% (the max and average $|W_{f_i^*}(\omega)|$ are about 72 and 12.7, respectively). Recall that, without the random byte insertion, $W_{f_i}(\omega) = 0$ with approximately 99.7% and the average of $|W_{f_i}(\omega)|$ is approximately 0.7. This implies that the encoding imbalance increases in the linear transformation with an unbalanced intermediate value by inserting a random byte.

For comparison, we conducted an additional experiment as follows.

1. Let $y^\gamma(x) = [\gamma 1 \; \gamma 2 \; \gamma 3 \; \gamma 4 \; \gamma 5]^T$ for each $x \in \mathrm{GF}(2^8)$. In other words, this is a five-byte random vector.
2. $f^\gamma(x) = M^* \cdot y^\gamma(x)$.
3. Repeat step (1) - (2) with 1,000 random M^* matrices, and accumulate the number of occurrences of each value of $W_{f_i^\gamma}(\omega)$.
4. Compute % of $W_{f_i^\gamma}(\omega) = 0$ and the average $|W_{f_i^\gamma}(\omega)|$.
5. Compute the cosine similarity between the distributions of $W_{f_i^\gamma}(\omega)$ and $W_{f_i^*}(\omega)$ for each ρ_i.

As a result, we have $W_{f_i^\gamma}(\omega) = 0$ with approximately 5% (the max and average $|W_{f_i^\gamma}(\omega)|$ are approximately 76 and 12.74, respectively) and the cosine similarity between their distributions is always larger than 0.999. The cosine similarity larger than 0.99 means they show very similar distribution. We note that the cosine similarity between the distributions of $W_{f_i^\gamma}(\omega)$ and $W_{f_i}(\omega)$ is about 0.25.

In order to visualize this effect of inserting a random byte, we select ρ_5 and calcalculate the sum of the imbalances of $W_{f_i^*}(\omega)$ for each key candidate with ω such that $\mathrm{HW}(\omega) = 1$ as follows:

$$\varDelta^{f^*}_{k\in\{0,1\}^8} = \sum_{\omega=1,2,\ldots,128}\;\sum_{i=1,\ldots,40} |W_{f_i^*}(\omega)|.$$

	ρ_1	ρ_2	ρ_3	ρ_4	ρ_5
% of $W_{f_i^*}(\omega) = 0$	5.05 (0.03)	5.06 (0.07)	4.93 (0.05)	5.0 (0.05)	5.04 (0.04)
Average of $\lvert W_{f_i^*}(\omega)\rvert$	12.73 (0.02)	12.75 (0.01)	12.76 (0.01)	12.73 (0.01)	12.76 (0.01)
Similarity with $W_{f_i^\gamma}$	> 0.999				

Table 2: $W_{f_i^*}$ after inserting a random byte at each inserting position (the standard deviation in parenthesis), and the cosine similarity of the distributions between $W_{f_i^*}$ and $W_{f_i^\gamma}$.

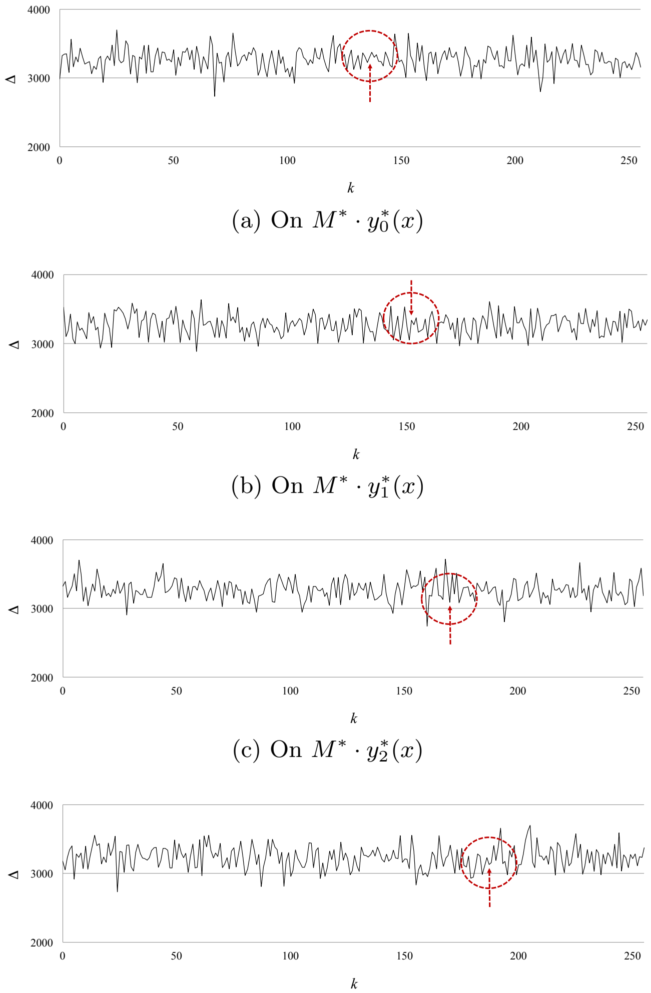
Fig. 5 shows $\Delta^{f^*}_{k\in\{0,1\}^8}$ and we can see that the correct subkeys *0x*88 - *0x*BB are no longer distinguishable from other candidates. In addition, it is noticeable that inserting more than one random byte in the intermediate values does not increase the imbalance; they show a similar level of the imbalance with the one-byte insertion.

4 Conclusion

Previous studies have shown that rows of HW 1 in the matrix are the main cause of key leakages from the linear transformation. Also, it has been suggested to recover the key in the presence of such a matrix without identity row by calculating all possible linear combinations of the bits in the target intermediate value. In this paper, we pointed out that there is no such row of HW 1 if we choose a block invertible matrix with 4×4 submatrices of full rank for maximizing information diffusion. Nevertheless, we showed that the key leakage is likely to happen from the linear transformation regardless of the HW of block invertible matrices. In addition, we pointed out that the SubBytes output is not protected by an 8×8 linear transformation but the SubBytes output multiplied with the decomposed MixColumns is supposed to be protected by a 32×32 one. Thus, it is complicated for an attacker to analyze all possible linear combinations. Our analysis explained that the balanced distribution of intermediate values cause the key leakage. We demonstrated that the unbalanced distribution of the intermediate values can be effective to reduce the probability of key leakage.

References

1. Akkar, M.L., Giraud, C.: An Implementation of DES and AES, Secure against Some Attacks. In: Proceedings of the Third International Workshop on Cryptographic Hardware and Embedded Systems. pp. 309–318. CHES '01, Springer-Verlag, London, UK, UK (2001)
2. Alpirez Bock, E., Brzuska, C., Michiels, W., Treff, A.: On the ineffectiveness of internal encodings - Revisiting the DCA attack on white-box cryptography. In: Applied Cryptography and Network Security - 16th International Conference, ACNS



(a) On $M^* \cdot y_0^*(x)$

(b) On $M^* \cdot y_1^*(x)$

(c) On $M^* \cdot y_2^*(x)$

(d) On $M^* \cdot y_3^*(x)$

Fig. 5: Sum of the imbalance of $W_{f_i^*}(\omega)$ for all key candidates. Red arrow: the correct key.

2018, Proceedings. pp. 103–120. Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics), Springer, Germany (1 2018)

3. Axsan white-box cryptographic solution.: `https://www.arxan.com/technology/white-box-cryptography/`

4. Billet, O., Gilbert, H., Ech-Chatbi, C.: Cryptanalysis of a White Box AES Implementation. In: Selected Areas in Cryptography, 11th International Workshop, SAC 2004, Waterloo, Canada, August 9-10, 2004, Revised Selected Papers. pp. 227–240 (2004)

5. Bos, J.W., Hubain, C., Michiels, W., Teuwen, P.: Differential Computation Analysis: Hiding Your White-Box Designs is Not Enough. In: Cryptographic Hardware and Embedded Systems - CHES 2016 - 18th International Conference, Santa Barbara, CA, USA, August 17-19, 2016, Proceedings. pp. 215–236 (2016), https://doi.org/10.1007/978-3-662-53140-2_11
6. Brier, E., Clavier, C., Olivier, F.: Correlation Power Analysis with a Leakage Model. In: Cryptographic Hardware and Embedded Systems - CHES 2004: 6th International Workshop Cambridge, MA, USA, August 11-13, 2004. Proceedings. Lecture Notes in Computer Science, vol. 3156, pp. 16–29. Springer (2004)
7. Bringer, J., Chabanne, H., Dottax, E.: White Box Cryptography: Another Attempt. IACR Cryptology ePrint Archive 2006, 468 (2006)
8. Carlet, C., Goubin, L., Prouff, E., Quisquater, M., Rivain, M.: Higher-Order Masking Schemes for S-Boxes. In: Fast Software Encryption - 19th International Workshop, FSE 2012, Washington, DC, USA, March 19-21, 2012. Revised Selected Papers. pp. 366–384 (2012), https://doi.org/10.1007/978-3-642-34047-5_21
9. Chow, S., Eisen, P., Johnson, H., Oorschot, P.C.V.: White-Box Cryptography and an AES Implementation. In: Proceedings of the Ninth Workshop on Selected Areas in Cryptography (SAC 2002). pp. 250–270. Springer-Verlag (2002)
10. Coron, J., Goubin, L.: On Boolean and Arithmetic Masking against Differential Power Analysis. In: Cryptographic Hardware and Embedded Systems - CHES 2000, Second International Workshop, Worcester, MA, USA, August 17-18, 2000, Proceedings. pp. 231–237 (2000), https://doi.org/10.1007/3-540-44499-8_18
11. Gemalto white-box cryptographic solution: https://sentinel.gemalto.com/software-monetization/white-box-cryptography/
12. Golic, J.D., Tymen, C.: Multiplicative Masking and Power Analysis of AES. In: Cryptographic Hardware and Embedded Systems - CHES 2002, 4th International Workshop, Redwood Shores, CA, USA, August 13-15, 2002, Revised Papers. pp. 198–212 (2002), https://doi.org/10.1007/3-540-36400-5_16
13. Goubin, L., Masereel, J., Quisquater, M.: Cryptanalysis of White Box DES Implementations. In: Selected Areas in Cryptography, 14th International Workshop, SAC 2007, Ottawa, Canada, August 16-17, 2007, Revised Selected Papers. pp. 278–295 (2007)
14. InsideSecure white-box cryptographic solution: https://www.insidesecure.com/Products/Application-Protection/Software-Protection/WhiteBox
15. Joye, M., Paillier, P., Schoenmakers, B.: On Second-Order Differential Power Analysis. In: Cryptographic Hardware and Embedded Systems - CHES 2005, 7th International Workshop, Edinburgh, UK, August 29 - September 1, 2005, Proceedings. pp. 293–308 (2005), https://doi.org/10.1007/11545262_22
16. Karroumi, M.: Protecting White-Box AES with Dual Ciphers. In: Information Security and Cryptology - ICISC 2010 - 13th International Conference, Seoul, Korea, December 1-3, 2010, Revised Selected Papers. pp. 278–291 (2010)
17. Kim, H., Hong, S., Lim, J.: A Fast and Provably Secure Higher-order Masking of AES S-box. In: Proceedings of the 13th international conference on Cryptographic hardware and embedded systems. pp. 95–107. CHES'11, Springer-Verlag, Berlin, Heidelberg (2011)
18. Kocher, P.C., Jaffe, J., Jun, B.: Differential Power Analysis. In: Advances in Cryptology - CRYPTO '99, 19th Annual International Cryptology Conference, Santa Barbara, California, USA, August 15-19, 1999, Proceedings. pp. 388–397 (1999)
19. Lee, S., Choi, D., Choi, Y.J.: Conditional Re-encoding Method for Cryptanalysis-Resistant White-Box AES. vol. 5. Electronics and Telecommunications Research Institute (Oct 2015), http://dx.doi.org/10.4218/etrij.15.0114.0025

20. Lepoint, T., Rivain, M., Mulder, Y.D., Roelse, P., Preneel, B.: Two Attacks on a White-Box AES Implementation. In: Selected Areas in Cryptography - SAC 2013 - 20th International Conference, Burnaby, BC, Canada, August 14-16, 2013, Revised Selected Papers. pp. 265–285 (2013)
21. Link, H.E., Neumann, W.D.: Clarifying Obfuscation: Improving the Security of White-box DES. In: International Conference on Information Technology: Coding and Computing (ITCC'05) - Volume II. vol. 1, pp. 679–684 Vol. 1 (2005)
22. Mangard, S., Oswald, E., Popp, T.: Power Analysis Attacks: Revealing the Secrets of Smart Cards (Advances in Information Security) (2007)
23. Messerges, T.S.: Securing the AES Finalists Against Power Analysis Attacks. In: Fast Software Encryption, 7th International Workshop, FSE 2000, New York, NY, USA, April 10-12, 2000, Proceedings. pp. 150–164 (2000), https://doi.org/10.1007/3-540-44706-7_11
24. Messerges, T.S.: Using Second-Order Power Analysis to Attack DPA Resistant Software. In: Cryptographic Hardware and Embedded Systems - CHES 2000, Second International Workshop, Worcester, MA, USA, August 17-18, 2000, Proceedings. pp. 238–251 (2000), https://doi.org/10.1007/3-540-44499-8_19
25. Michiels, W., Gorissen, P., Hollmann, H.D.L.: Cryptanalysis of a Generic Class of White-Box Implementations. In: Selected Areas in Cryptography, 15th International Workshop, SAC 2008, Sackville, New Brunswick, Canada, August 14-15, Revised Selected Papers. pp. 414–428 (2008)
26. Mulder, Y.D., Roelse, P., Preneel, B.: Cryptanalysis of the Xiao - Lai White-Box AES Implementation. In: Selected Areas in Cryptography, 19th International Conference, SAC 2012, Windsor, ON, Canada, August 15-16, 2012, Revised Selected Papers. pp. 34–49 (2012)
27. Mulder, Y.D., Wyseur, B., Preneel, B.: Cryptanalysis of a Perturbated White-Box AES Implementation. In: Progress in Cryptology - INDOCRYPT 2010 - 11th International Conference on Cryptology in India, Hyderabad, India, December 12-15, 2010. Proceedings. pp. 292–310 (2010)
28. Oswald, E., Mangard, S., Pramstaller, N., Rijmen, V.: A Side-channel Analysis Resistant Description of the AES S-box. In: Proceedings of the 12th international conference on Fast Software Encryption. pp. 413–423. FSE'05, Springer-Verlag, Berlin, Heidelberg (2005)
29. Prouff, E., Rivain, M.: Masking against Side-Channel Attacks: A Formal Security Proof. In: Advances in Cryptology - EUROCRYPT 2013, 32nd Annual International Conference on the Theory and Applications of Cryptographic Techniques, Athens, Greece, May 26-30, 2013. Proceedings. pp. 142–159 (2013), https://doi.org/10.1007/978-3-642-38348-9_9
30. Rivain, M., Prouff, E.: Provably Secure Higher-order Masking of AES. In: Proceedings of the 12th International Conference on Cryptographic Hardware and Embedded Systems. pp. 413–427. CHES'10, Springer-Verlag, Berlin, Heidelberg (2010)
31. Rivain, M., Wang, J.: Analysis and Improvement of Differential Computation Attacks against Internally-Encoded White-Box Implementations. IACR Transactions on Cryptographic Hardware and Embedded Systems 2019(2), 225–255 (Feb 2019), https://tches.iacr.org/index.php/TCHES/article/view/7391
32. Sanfelix, E., Mune, C., de Haas, J.: Unboxing the White-Box: Practical Attacks against Obfuscated Ciphers. In: Presented at BlackHat Europe 2015 (2015), https://www.blackhat.com/eu-15/briefings.html

33. Sasdrich, P., Moradi, A., Güneysu, T.: White-Box Cryptography in the Gray Box - - A Hardware Implementation and its Side Channels -. In: Fast Software Encryption - 23rd International Conference, FSE 2016, Bochum, Germany, March 20-23, 2016, Revised Selected Papers. pp. 185–203 (2016)
34. Schramm, K., Paar, C.: Higher Order Masking of the AES. In: Proceedings of the 2006 The Cryptographers' Track at the RSA conference on Topics in Cryptology. pp. 208–225. CT-RSA'06, Springer-Verlag, Berlin, Heidelberg (2006)
35. Trichina, E., Seta, D.D., Germani, L.: Simplified Adaptive Multiplicative Masking for AES. In: Cryptographic Hardware and Embedded Systems - CHES 2002, 4th International Workshop, Redwood Shores, CA, USA, August 13-15, 2002, Revised Papers. pp. 187–197 (2002), `https://doi.org/10.1007/3-540-36400-5\_15`
36. Waddle, J., Wagner, D.A.: Towards Efficient Second-Order Power Analysis. In: Cryptographic Hardware and Embedded Systems - CHES 2004: 6th International Workshop Cambridge, MA, USA, August 11-13, 2004. Proceedings. pp. 1–15 (2004), `https://doi.org/10.1007/978-3-540-28632-5\_1`
37. WhiteboxCRYPTO: `https://www.microsemi.com/document-portal/doc_view/135631-whiteboxcrypto-product-overview-rev4`
38. Wyseur, B., Michiels, W., Gorissen, P., Preneel, B.: Cryptanalysis of White-Box DES Implementations with Arbitrary External Encodings. In: Selected Areas in Cryptography, 14th International Workshop, SAC 2007, Ottawa, Canada, August 16-17, 2007, Revised Selected Papers. pp. 264–277 (2007)
39. Xiao, J., Zhou, Y.: Generating Large Non-Singular Matrices over an Arbitrary Field with Blocks of Full Rank (2002), `http://eprint.iacr.org/2002/096`
40. Xiao, Y., Lai, X.: A Secure Implementation of White-box AES. In: The Second Internationial Conference on Computer Science and Its Applications - CSA 2009. vol. 2009, pp. 1–6 (2009)